10-15-2021

Improving Neural Network Robustness via Persistency of Excitation

Kaustubh Sridhar
University of Pennsylvania, ksridhar@seas.upenn.edu

Oleg Sokolsky
University of Pennsylvania

Insup Lee
University of Pennsylvania

James Weimer
University of Pennsylvania

Recommended Citation (OVERRIDE)

Improving Neural Network Robustness via Persistency of Excitation

Abstract

Improving adversarial robustness of neural networks remains a major challenge. Fundamentally, training a neural network via gradient descent is a parameter estimation problem. In adaptive control, maintaining persistency of excitation (PoE) is integral to ensuring convergence of parameter estimates in dynamical systems to their true values. We show that parameter estimation with gradient descent can be modeled as a sampling of an adaptive linear time-varying continuous system. Leveraging this model, and with inspiration from Model-Reference Adaptive Control (MRAC), we prove a sufficient condition to constrain gradient descent updates to reference persistently excited trajectories converging to the true parameters. The sufficient condition is achieved when the learning rate is less than the inverse of the Lipschitz constant of the gradient of loss function. We provide an efficient technique for estimating the corresponding Lipschitz constant in practice using extreme value theory. Our experimental results in both standard and adversarial training illustrate that networks trained with the PoE-motivated learning rate schedule have similar clean accuracy but are significantly more robust to adversarial attacks than models trained using current state-of-the-art heuristics.

Keywords

Adversarial Robustness of Deep Neural Networks, Persistency of Excitation, Adaptive Control Theory, Robust Parameter Estimation

Disciplines

Computer Engineering | Computer Sciences

Comments

Improving Neural Network Robustness via Persistency of Excitation

Kaustubh Sridhar Oleg Sokolsky Insup Lee James Weimer

***Abstract*— Improving adversarial robustness of neural networks remains a major challenge. Fundamentally, training a neural network via gradient descent is a parameter estimation problem. In adaptive control, maintaining persistency of excitation (PoE) is integral to ensuring convergence of parameter estimates in dynamical systems to their true values. We show that parameter estimation with gradient descent can be modeled as a sampling of an adaptive linear time-varying continuous system. Leveraging this model, and with inspiration from Model-Reference Adaptive Control (MRAC), we prove a sufficient condition to constrain gradient descent updates to reference persistently excited trajectories converging to the true parameters. The sufficient condition is achieved when the learning rate is less than the inverse of the Lipschitz constant of the gradient of loss function. We provide an efficient technique for estimating the corresponding Lipschitz constant in practice using extreme value theory. Our experimental results in both standard and adversarial training illustrate that networks trained with the PoE-motivated learning rate schedule have similar clean accuracy but are significantly more robust to adversarial attacks than models trained using current state-of-the-art heuristics.**

I. INTRODUCTION

Neural networks are vulnerable to adversarial examples [1] and most existing defenses are still highly susceptible to white box attacks [2], [3] (where the adversary has full access to the network and its defense mechanism).

We believe that adversarial robustness can be improved by leveraging the fact that every neural network training process (standard or robust) is a parameter estimation problem [4], where the goal is to find the true parameters of a model. A model[1] with its true parameters, *i.e.*, the parameters of the true mapping from its input space to output space, always maps similar inputs to similar outputs [4]. We posit (and empirically demonstrate) that this implies increased adversarial robustness for neural networks with their true parameters.

In system identification and adaptive control, Persistency of Excitation (PoE) conditions [5] are integral to robust estimation of true parameters. They restrict parameter estimation dynamics to exponentially-stable trajectories that ensure robust convergence to true values. Further, recent work [4] analyzed neural network training and identified the lack of PoE in gradient descent (GD) as a major roadblock on the path to robustness. Thus, the main challenge addressed by this work is ensuring neural network training dynamics, and specifically gradient descent dynamics, is persistently excited and converges to the network's true parameters.

Earlier attempts to characterize PoE for GD were either impeded by a neural network's inherent nonlinearities [6], [7] or limited to simple two layer networks and specific loss functions [4]. In this work, we overcome the nonlinearity and complexity trap faced by [4], [6], [7] with the insight of modeling GD as a discretization of an adaptive continuous-time (CT) linear time-varying (LTV) system. We take inspiration from Model-Reference Adaptive Control (MRAC) [5], where adaptive control laws are chosen such that the system's dynamics emulate a reference system's dynamics, to propose the following two-step approach.

First, we choose a reference family of persistently excited systems with a globally exponentially stable (GES) equilibrium at the unknown true parameters of the network. Then we prove sufficient conditions for consecutive updates of the discrete-time (DT) GD dynamics to lie on the exact discretization of a system from our reference family. Our novel two part approach theoretically guarantees convergence to the unknown true parameters of any model trained by minimizing a smooth loss with GD and empirically demonstrates increased robustness to adversarial attacks in stochastic gradient descent (SGD) based standard and adversarial training.

Our proven sufficient condition is equivalent to scaling a baseline learning rate schedule where the initial value is a function of the inverse of Lipschitz constant of the loss gradient. To ensure a rigorous evaluation with minimal increase in model training time, we estimate this second-order Lipschitz constant with an inexpensive addition to the baseline model training procedure via extreme value theory [8], [9]. To observe the utility of our persistently exciting learning schedule, we apply it to standard training on MNIST [10], CIFAR10, CIFAR100 [11] datasets, and adversarial training on CIFAR10 dataset. We see an increase in adversarial accuracy of up to 15 points against a 20-step PGD adversary [12] with perturbation budget $\epsilon = \frac{1}{255}$ in standard training and an increase up to 0.7 points in adversarial training on the competitive Autoattack benchmark [13] (with $\epsilon = \frac{8}{255}$) composed of both white-box and black-box attacks.

A. Related Work

PoE in Control Theory and Deep Learning. PoE has been thoroughly explored for CT LTV systems and is essential to robust parameter estimation in guaranteeing GES of parameter error dynamics which ensures convergence of estimated parameters to the true values [5], [14]. For learning-based system identification, early work [15], [16] found PoE

The authors would like to acknowledge the support of the Army Research Office award number W911NF2010080.

The authors are with the PRECISE Center in the Departments of Electrical & Systems Engineering and Computer & Information Science, University of Pennsylvania, Philadelphia, PA, USA; {ksridhar, sokolsky, lee, weimerj}@seas.upenn.edu.

Code: github.com/kaustubhsridhar/PoE-robustness

[1] Model and neural network are used interchangeably.

conditions for Radial Basis Functions but emphasizes the difficulty in characterizing PoE conditions for general neural networks because of the nonlinearities in the models [6], [7].

Recent seminal work in [4] aims to tackle this challenging problem. Based on the premise of robust neural networks having bounded Lipschitz constants [1], the authors derive sufficient richness conditions on the inputs to a two-layer network with ReLU activation functions trained with GD. However, their results are specific to a two-layer network initialized close to its true optima, particular loss functions and dependent on the gradient update rule. Moreover, these conditions do not scale to modern deep neural networks. To scale, they are forced to adopt an optimization trick to force noise (for PoE) into each layer of a network. This trick can also be found in other robust learning approaches [17], [18].

To avoid the issues faced in forward analysis by [4], we flip the problem around: we start with a well-characterized CT LTV family of persistently excited dynamics and then find sufficient conditions for GD updates to fit on the trajectories in this family. Our approach is only dependent on the gradient update rule and in practice, generalizes to all loss functions and scales to models that converge in training.

Techniques for Robust Learning. Adversarial training (AT), first introduced in [19], is currently, the most effective defense to white-box attacks. AT requires solving a min-max optimization problem. The inner maximization problem is approximately solved with the PGD attack in PGD-AT [12]. A variant that modifies the inner maximization problem to tradeoff clean accuracy for robust accuracy was proposed in TRADES [20]. Further improvement with additional unlabelled data (RST) [21] has increased robustness of models on the competitive AutoAttack benchmark, *a.k.a.* RobustBench (an ensemble of four white-box and black-box attacks with a single hyperparameter - perturbation budget $\epsilon = \frac{8}{255}$) [13]. Employing our PoE-motivated learning rate schedule further increases the robustness of models trained with the above state-of-the-art (SOTA) AT frameworks, thereby proving its importance as a force multiplier for any training algorithm.

Estimating Lipschitz Constant of Loss Gradient. Several works have studied neural network Lipschitz constant estimation (*e.g.* [22], [9]) but here, we are concerned with the Lipschitz constant of loss gradient (denoted $\mathcal{L}$). In [23], approximate upper bounds were derived for $\mathcal{L}$ but to our best knowledge, no efficient estimation method has been previously proposed and implemented in practical SGD. In this work, we apply an extreme value theory approach [8], [9] and estimate $\mathcal{L}$ in both standard and adversarial training.

B. Contributions

1) We propose extending PoE, with inspiration from MRAC, to neural network training to obtain sufficient conditions for convergence of GD dynamics for any model to its true parameters. Our insight into modeling GD as a sampling of an adaptive CT LTV system is vital to generalizing beyond the simple 2-layer network and certain loss functions in [4].

2) We present an efficient implementation strategy in practical SGD training, based on extreme value theory, to obtain an estimate of the initial learning rate in a learning schedule for PoE. We also detail a simple heuristic to tune batch size to satisfy a principal assumption in our derivation.

3) We demonstrate the effectiveness of our approach in standard training with SGD on MNIST, CIFAR10, and CIFAR100 (up to 15 points accuracy increase on 20-step, $\epsilon = 1/255$ PGD attack) & with various SOTA adversarial trained CIFAR10 models on the competitive Autoattack benchmark (universal improvements of up to 0.7 points with $\epsilon = 8/255$).

II. PROBLEM FORMULATION

In this Section, we mention some preliminaries from adaptive control & GD and then formally state our problem. **Adaptive Control Preliminaries:** For a continuous-time (CT) linear time-varying (LTV) system given by,

$$\dot{z}(t) = -\Phi(t)\Phi(t)^T z(t), \quad t \geq 0 \tag{1}$$

with $z(t) \in \mathbb{R}^d$, $\Phi(t) \in \mathbb{R}^{d\times p}$, PoE is defined as follows.

Definition 1 (PoE [5]). *The signal $\Phi(t) : \mathbb{R}^{\geq 0} \to \mathbb{R}^{d\times p}$ is persistently exciting if there exists $\mu_1, \mu_2, T_0 > 0$ such that,*

$$\mu_2 \mathbf{I} \geq \int_t^{t+T_0} \Phi(s)\Phi(s)^\top ds \geq \mu_1 \mathbf{I} \tag{2}$$

where $\mathbf{I}$ is the $d \times d$ Identity matrix.

PoE and GES are connected via the following lemma.

Lemma 1 (PoE and GES [5, Theorem 2.5.1]). *If $\Phi(t)$ is piece-wise continuous and persistently exciting, then system (1) is GES.*

Lemma 1 ensures GES convergence of states on (1) to their equilibrium and informs our definition of the persistently exciting reference family for GD updates to track.

GD Preliminaries. We represent feature space with $\mathcal{X}$, label space with $\mathcal{Y}$ and model with parameters $\Theta \in \mathcal{P}$ given by $h_\Theta : \mathcal{X} \to \mathcal{Y}$. In the theoretical part of this work, we focus on model training with GD wherein, we represent the training data with $(X, Y) \in \mathcal{X} \times \mathcal{Y}$ and the loss function minimized with $L : \mathcal{Y} \times \mathcal{Y} \to \mathbb{R}$. We denote the vectorized version of parameters as $\theta = \text{vec}(\Theta) \in \mathbb{R}^d$, vectorized loss gradients as $\nabla l(\theta) = \text{vec}\left(\nabla L(h_\Theta(X), Y)\right) \in \mathbb{R}^d$, and learning rate as η. Now, we write the vectorized GD update step below.

Definition 2 (Vectorized GD Update). *The vectorized form of the kth update step in GD for training a model h_Θ on training data (X, Y) by minimizing loss $L(h_\Theta(X), Y)$ with learning rate η^k is given by*

$$\theta^{k+1} = \theta^k - \eta^k \nabla l(\theta^k), \quad k = 1, 2, \ldots. \tag{3}$$

The above definition casts GD into a DT nonlinear time-varying (NLTV) system. Analyzing this system for a particular loss function l and model architecture h_θ is intractable for increasingly larger models trained by minimizing custom loss functions. We propose a bottom-up solution for this problem.

First, we choose a family of persistently excited CT dynamics such that all of its members converge exponentially-fast to the unknown true parameters (denoted θ^*) of a model. In this work, we conjecture (and empirically demonstrate in Sections IV, V) that the true parameters coincide with the maximal ϵ-robust optimum (where an ϵ-robust optimum provides a model with the same output for all inputs in a ball of perturbation size ϵ around any input in domain $\mathcal{X}$).

Second, we find sufficient conditions to constrain consecutive states of the DT NLTV system in (3) to lie on discretized trajectories from the aforementioned family. Through these two steps we obtain sufficient conditions for convergence of GD updates to the true parameters of the model. We tackle the first of the two steps below by choosing the following family of dynamics with each member of the family, for $k \geq 1$, starting at the kth GD update (θ^k).

Definition 3 (**Reference Family of Persistently Exciting CT Systems**). *The reference family of persistently exciting CT systems, with GES equilibrium at θ^*, that governs the evolution of a state vector $\Gamma(t) \in \mathbb{R}^d$, $t \geq k$ with initial value $\Gamma(k) = \theta^k$, is given by*

$$\dot{\Gamma}(t) = -\Phi(t)\Phi(t)^\top(\Gamma(t) - \theta^*) \tag{4}$$

where $\Phi(t) = \Phi^k \;\forall\; t \in [k, k+1)$ is piece-wise constant matrix $\in \mathbb{R}^{d\times p}$ and $\Phi^k\Phi^{k^\top}$ is full rank $\forall\, k$.

Equation (4) is a form of the CT LTV system in (1) with an equilibrium at θ^*. Our choice of $\Phi(t)$ in the above definition ensures that $\Phi(t)$ is persistently exciting and consequently that the system in (1) has GES equilibrium (Proved in Section III). Finally, with the requisite family defined, we formally state the problem considered by this paper below.

Problem Statement 1 (**PoE of GD**). *We aim to find sufficient conditions for every pair of consecutive kth-step GD updates θ^k, θ^{k+1} (Definition 2) to lie on a discretized trajectory from the reference persistently excited CT family in Definition 3.*

We remark that any mention of 'PoE of GD' in this work denotes the above desired property of the GD dynamics.

III. Main Theoretical Results on Learning Rates for PoE of GD

In this section, we present our main theoretical result in Theorem 1 which proposes an upper bound on learning rates η^k, $k \geq 1$ to accomplish our problem statement. Also, we discuss the proven sufficient condition and its connections to convex optimization & constant learning rate training. Before stating Theorem 1, we present our assumptions on the $\mathcal{L}-$smoothness of loss and the acuteness of descent directions below.

Assumption 1 (**$\mathcal{L}-$Smooth Loss Function**). *The loss function is $\mathcal{L}-$smooth if its gradient $\nabla l : \mathbb{R}^d \rightarrow \mathbb{R}^d$ is $\mathcal{L}-$Lipschitz, i.e. there exists a constant $\mathcal{L} > 0$ such that*

$$\forall\, \theta_1, \theta_2 \in \mathbb{R}^d, \;\; \|\nabla l(\theta_2) - \nabla l(\theta_1)\|_2 \leq \mathcal{L}\|\theta_2 - \theta_1\|_2. \tag{5}$$

$\mathcal{L}$ is also called the second-order Lipschitz constant. $\mathcal{L}$-smoothness is a commonly recurring assumption in optimization theory [24], [25]. In practice, standard and adversarial losses are smooth for certain models [26], [27] and not others. Yet, in Section V, our approach results in increased robustness for a wide variety of architectures.

Assumption 2 (**Acuteness of descent directions**). *The angle between the kth descent direction and the next true descent direction (from $(k+1)$th update to true parameters) is acute.*

$$\textit{i.e.}\;\; (\theta^k - \theta^{k+1})^\top(\theta^{k+1} - \theta^*) \geq 0 \tag{6}$$

Assumption 2 states that the local gradient and true gradient are acute, an intuitive property of GD with training data that is representative of the population. Further, we monitor this assumption in our experiments in Section IV and observe that it is indeed satisfied throughout model training with large batch SGD and with GD (full-batch SGD). Thus, leveraging Assumptions 1, 2, we state the main theorem below.

Theorem 1 (**Sufficient Conditions for PoE of GD**). *Consider a model trained via GD with a learning rate schedule given by $\left(\eta^k\right)_{k\geq 1}$ by minimizing a $\mathcal{L}-$smooth loss (Assumption 1) and satisfying Assumption 2. We have PoE of GD and hence convergence of GD updates to the model's true parameters if $\eta^k < 1/\mathcal{L}$ for all k.*

Proof. We begin with our sufficient condition: $\eta^k < \frac{1}{\mathcal{L}}$ which can be rewritten as $\mathcal{L} < \frac{1}{\eta^k}$ such that,

$$\frac{\|\nabla l(\theta^k) - \nabla l(\theta^*)\|_2}{\|\theta^k - \theta^*\|_2} < \frac{1}{\eta^k} \tag{7}$$

Observing the gradient at the optima is zero in (7), i.e., $\nabla l(\theta^*) = 0$, we have, $\frac{\|\nabla l(\theta^k) - 0\|_2}{\|\theta^k - \theta^*\|_2} < \frac{1}{\eta^k} \iff \eta^k\|\nabla l(\theta^k)\|_2 < \|\theta^k - \theta^*\|_2$. By substituting (3), we have,

$$\|\theta^k - \theta^{k+1}\|_2 < \|\theta^k - \theta^*\|_2$$
$$\iff \|U^\top(\theta^k - \theta^{k+1})\|_2 < \|U^\top(\theta^k - \theta^*)\|_2 \;\forall\, U \in SO(d)$$

where $SO(d)$ is the set of orthonormal rotation matrices in d-dimensions (since rotated vectors maintain their magnitudes). Now, choosing $U = [v_1, v_2, v_3, ..., v_d]$, $v_i^\top v_j = 0$, $\|v_i\|_2 = 1$ where (for infitesimally small $\delta > 0$),

$$v_1 = \frac{(\theta^k - \theta^{k+1}) - \delta(\theta^k - \theta^*)}{\|(\theta^k - \theta^{k+1}) - \delta(\theta^k - \theta^*)\|_2} \tag{8}$$

$$v_2^\top v_1 = 0 \implies v_2^\top(\theta^k - \theta^{k+1}) - \delta v_2^\top(\theta^k - \theta^*) = 0 \tag{9}$$

then we can write, $(\theta^k - \theta^{k+1}) = a_1 v_1 + a_2 v_2$ and $(\theta^k - \theta^*) = b_1 v_1 + b_2 v_2$ for constants a_1, a_2, b_1, b_2 as follows.

$$\begin{aligned} a_1 &= v_1^\top(\theta^k - \theta^{k+1}) \\ &= \frac{\|\theta^k - \theta^{k+1}\|_2^2 - \delta(\theta^k - \theta^{k+1})^\top(\theta^k - \theta^*)}{\|(\theta^k - \theta^{k+1}) - \delta(\theta^k - \theta^*)\|_2} \end{aligned} \tag{10}$$

$$\begin{aligned} b_1 &= v_1^\top(\theta^k - \theta^*) \\ &= \frac{(\theta^k - \theta^{k+1})^\top(\theta^k - \theta^*) - \delta\|\theta^k - \theta^*\|_2^2}{\|(\theta^k - \theta^{k+1}) - \delta(\theta^k - \theta^*)\|_2} \end{aligned} \tag{11}$$

$$a_2 = v_2^\top(\theta^k - \theta^{k+1}) = \delta v_2^\top(\theta^k - \theta^*) \;\; (\text{via } (9)) \tag{12}$$

$$b_2 = v_2^\top(\theta^k - \theta^*) \tag{13}$$

From Assumption 2, we have

$$\begin{aligned} \frac{(\theta^k - \theta^{k+1})^\top(\theta^k - \theta^*)}{\|\theta^k - \theta^{k+1}\|_2^2} &= 1 + \frac{(\theta^k - \theta^{k+1})^\top(\theta^{k+1} - \theta^*)}{\|\theta^k - \theta^{k+1}\|_2^2} \\ &\geq 1 \quad (\text{from } (6)) \\ \implies (\theta^k - \theta^{k+1})^\top(\theta^k - \theta^*) &\geq \|\theta^k - \theta^{k+1}\|_2^2. \end{aligned} \tag{14}$$

Therefore, in the limit of $\delta \to 0$, we have $b_1 \geq a_1 > 0$ (from (10), (11), (14)) and $b_2 > a_2 \to 0^+$. The latter is because as $\delta \to 0$, we have v_1 lying along $(\theta^k - \theta^{k+1})$ which means, from (14), v_1 and $(\theta^k - \theta^*)$ are acute. This in turn implies v_2 and $(\theta^k - \theta^*)$ are acute and hence from (13), $b_2 > 0$ and from (12), $a_2 \to 0$ from the positive side.

Thus, for the above choice of U, in the limit of $\delta \to 0$, we have, $U^\top(\theta^k - \theta^{k+1}) = [v_1^\top, v_2^\top, \dots, v_d^\top]^\top(a_1 v_1 + a_2 v_2) = [a_1, a_2, 0, \dots, 0]^\top \leq [b_1, b_2, 0, \dots, 0]^\top = [v_1^\top, v_2^\top, \dots, v_d^\top]^\top(b_1 v_1 + b_2 v_2) = U^\top(\theta^k - \theta^*)$.

Continuing in the limit of $\delta \to 0$ and choosing $\Sigma = \text{diag}\left(\frac{a_1}{b_1}, \frac{a_2}{b_2}, c_3, \dots, c_d\right)$ where $0 < c_i < 1$ for $i = 3, \dots, d$ (note $0 < \Sigma < \mathbf{I}$), we can scale down the right hand side vector above to match the left hand side vector as follows,

$$
\begin{aligned}
& U^\top(\theta^k - \theta^{k+1}) = \Sigma U^\top(\theta^k - \theta^*) \\
\iff\ & (\theta^k - \theta^{k+1}) = U \Sigma U^\top(\theta^k - \theta^*) \\
\iff\ & (\theta^k - \theta^{k+1}) = (\mathbf{I} - e^{-\Phi^k {\Phi^k}^\top})(\theta^k - \theta^*) \qquad (15)
\end{aligned}
$$

Since $\Phi^k {\Phi^k}^\top$ is full rank and we observe $(\mathbf{I} - e^{-\Phi^k {\Phi^k}^\top}) = VDV^\top$ with $V \in SO(d)$ and diagonal $0 < D < \mathbf{I}$, we can choose Φ^k such that $V = U$ and $D = \Sigma$. Rewriting (15),

$$
\theta^{k+1} - \theta^* = e^{-\Phi^k {\Phi^k}^\top}(\theta^k - \theta^*). \qquad (16)
$$

Since (16) is equivalent to the discretization of the CT dynamics of system (4) in time interval $[k, k+1]$ with $\Gamma(k+1) = \theta^{k+1}$ and initial value $\Gamma(k) = \theta^k$, we have proven that θ^k, θ^{k+1} lie on the discretized trajectory of said system from the reference family of Definition 3.

Finally, since $\Phi^k {\Phi^k}^\top$ is full rank and $\Phi^k {\Phi^k}^\top$ is positive definite, the system from (4) in time interval $[k, k+1]$ given by $\dot{\Gamma}(t) = -\Phi^k {\Phi^k}^\top(\Gamma(t) - \theta^*), t \geq k$ is persistently excited (since for any $T > 0$, we have $\int_t^{t+T} \Phi^k {\Phi^k}^\top ds = \Phi^k {\Phi^k}^\top T$ and $0 < \lambda_{\min} T \leq \Phi^k {\Phi^k}^\top T \leq \lambda_{\max} T$ where $\lambda_{\min}, \lambda_{\max}$ are the minimum and maximum eigenvalues of positive definite $\Phi^k {\Phi^k}^\top$) and has GES equilibrium at true optimum θ^*. □

Next, we provide some discussions on the main result.

Remark 1. On sufficient conditions for PoE and the conservative upper bound of $1/\mathcal{L}$. It is worth noting that the proof has two sufficient conditions: full rankness of $\Phi^k {\Phi^k}^\top$ which is sufficient but not necessary for PoE and $\eta^k < 1/\mathcal{L}$ which is sufficient but not necessary for Inequality (7) to hold. With these 2 sufficient conditions, our upper bound $1/\mathcal{L}$ is conservative and values greater than it may also ensure PoE. In fact, inspired by empirical successes in Sections IV, V we later conjecture that an upper bound of $2/\mathcal{L}$ may ensure PoE. Further, a necessary & sufficient condition for PoE instead of the first sufficient condition above may provide a larger upper bound and is an interesting open problem.

Remark 2. On the connection to convex optimization. In GD on a $\mathcal{L}-$smooth and convex loss function, a learning rate choice of $\eta^k \leq \frac{1}{\mathcal{L}}$ guarantees monotonic progress to the minima [24]. Our similar result is expected because every persistently exciting trajectory, on which states converge exponentially fast to minima θ^*, is a convex shortcut from θ^k to θ^* through θ^{k+1} that may/may not lie on the loss surface. This relationship provides an interesting intuition for our approach and strengthens its validity.

Remark 3. On training with a constant learning rate and GD vs SGD. In GD on any $\mathcal{L}-$smooth differentiable loss function with a fixed learning rate η, the algorithm converges to local minima if $\eta < \frac{2}{\mathcal{L}}$ [25]. Thus, if a model converges via GD with a fixed learning rate, halving it should be adequate for PoE. Unfortunately the simplicity of dividing by 2 does not always work in practice because modern neural network training algorithms, faced with GPU constraints, use SGD rather than GD and learning schedules rather than a constant learning rate for which this holds. We discuss this gap ahead.

IV. Implementation in SGD Training

In practice, models are trained with SGD and its variants [24] rather than GD where a slowly-decaying learning rate schedule (such as a sequence obeying $\sum_k \eta^k = \infty$, $\sum_k {\eta^k}^2 < \infty$ [28]) is often necessary for training to converge in the first place. Thus, in this Section, we present an implementation strategy for SGD based training that satisfies Theorem 1, Assumption 2 and actually converges.

A. PoE-motivated learning rate schedules for SGD

We assume a baseline learning rate schedule, $(\gamma^k)_{k \geq 1}$ with $\gamma^k \leq \gamma^1\ \forall\ k$, that ensures convergence to a local optima. Our *PoE-motivated learning rate schedule* starts at $\eta^1 = \frac{1}{\mathcal{L}_{\text{est}}}$ and is subsequently scaled in the same way as the baseline schedule, *i.e.* $\eta^k = \eta^1 \frac{\gamma^k}{\gamma^1}\ \forall\ k \geq 2$. Our algorithm for obtaining $\mathcal{L}_{\text{est}}$ (see Section IV-B) always provides an estimate larger than the true value [8], [9] ensuring that $\eta^1 = \frac{1}{\mathcal{L}_{\text{est}}} < \frac{1}{\mathcal{L}_{\text{true}}}$ and since $\eta^k \leq \eta^1\ \forall\ k \geq 2$, Theorem 1 is satisfied. Further, by following a similar annealing cycle as the baseline schedule, we have convergence in practice. Lastly, following Remark 1, we analyze another schedule, *a.k.a. largest convergent schedule*, where $\eta^1 = 2/\mathcal{L}_{\text{est}}$ and we similarly scale subsequent values $\eta^k = \eta^1 \frac{\gamma^k}{\gamma^1}\ \forall\ k \geq 2$. This too ensures convergence in practice and we later conjecture in Section VI that it also leads to PoE. Figure 1 shows an example of these schedules for typical annealing strategies.

Our choice of initial learn rate $\eta^1 = \frac{1}{\mathcal{L}_{\text{est}}}$ which is close to Theorem 1's upper bound stems from an experimental analysis of adversarial accuracy versus learning rate. We trained LeNet5 models [10] (with ReLU/Tanh activations) with various constant learning rates for 10 epochs via SGD on MNIST. Each trained model is evaluated against a 40-step PGD attack with $\epsilon = 0.3$. Plotting the clean and PGD attack accuracy in Figure 2, we see that PGD attack accuracy peaks near the largest learning rate at which the model converges to a local optima (*i.e.* when clean accuracy is close to 100%), theoretically given by $\frac{2}{\mathcal{L}}$ [25]. Also, at half this point (our upper bound of $\frac{1}{\mathcal{L}}$), we notice PGD attack accuracy is still high but drops immediately on the left. This drop can be explained by the ill-conditioning of $||\nabla l(\theta^k)||/||\theta^k - \theta^*||$ term in (7) where the denominator is small for small learning rates. Moreover, since it is hard to predict the exact drop point, we stay close to the upper bound and use $\frac{1}{\mathcal{L}_{\text{est}}}$ & $\frac{2}{\mathcal{L}_{\text{est}}}$.

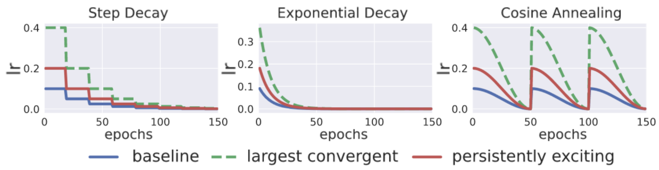


Fig. 1. An *e.g.* of baseline ($\eta^1 = 0.1$), largest convergent ($\eta^1 = 2/\mathcal{L}$) & PoE-motivated ($\eta^1 = 1/\mathcal{L}$) learning rate (lr) schedules for step decay, exponential decay & cosine annealing strategies.

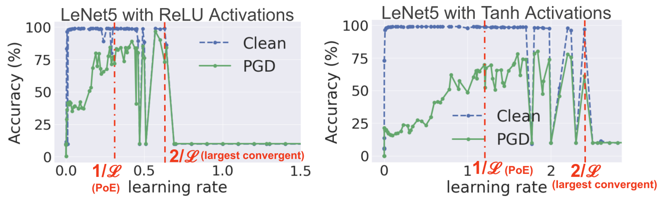
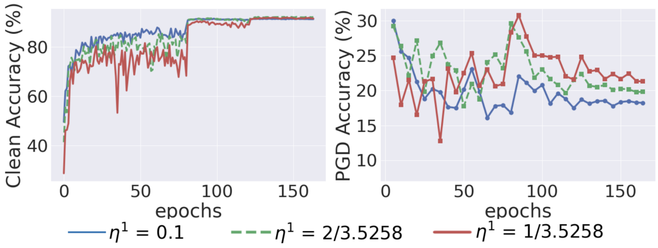


Fig. 2. Accuracy on Clean and PGD attacked MNIST validation set for a LeNet5 model (with ReLU [left] and Tanh [right] activations) vs constant learning rate (lr) used in training. The largest convergent and the PoE-motivated lr's have higher PGD accuracy than baseline lr = 0.1.

B. Estimation of Certified Lipschitz Constant $\mathcal{L}_{est}$

Inspired by [8], [9], we estimate $\mathcal{L}$ with three steps:
(1) We collect average loss gradient and model parameters after each epoch in baseline training (*i.e.* with the baseline schedule). They're denoted by $(\nabla l(\theta^i),\, \theta^i)_{1,\ldots,N_{\text{epochs}}}$.
(2) We estimate a Lipschitz constant by sampling N points, computing $N/2$ slopes between consecutive pairs as $s_i = \frac{||\nabla l(\theta^{i+1}) - \nabla l(\theta^i)||_2}{||\theta^{i+1}-\theta^i||_2}$, $i = 1, 3, 5, \ldots, N$ and finding the maximum, $l = \max\{s_1, s_3, s_5, \ldots, s_N\}$. We repeat this M times and applying the Fisher–Tippett–Gnedenko theorem [8], fit a 3 parameter (shape, location, scale) reverse Weibull distribution to $\{l_1, \ldots, l_M\}$ given an initial shape value. The fitted scale parameter is the desired estimate of Lipschitz constant.
(3) We certify our estimated Lipschitz constant by iterating between Step (2) and a Kolmogorov–Smirnov (K-S) test to test that our samples $\{l_1, \ldots, l_M\}$ are drawn from a reverse Weibull distribution with the Step (2)'s fitted parameters. Out of various p-values obtained, we choose the Lipschitz constant (*i.e.* scale parameter) with the highest p-value. However, a question remains, how are hyperparameters M, N tuned?
A heuristic for hyper-parameter (M, N) tuning. We repeat steps (2), (3) for different M, N and choose the Lipschitz constant from the case when atleast one p-value is both larger and smaller than a mid-to-large significance value $\alpha = 0.4$ to 0.6. This heuristic works well in practice. Using the above, we estimate $\mathcal{L}_{\text{est}}$ for ResNet20 standard training (*i.e.* minimizing cross-entropy loss on clean images) with $\alpha = 0.55, M = 200, N = 100$. Plotting clean & PGD accuracy in Figure 3, we observe that models trained with the PoE-motivated and largest convergent schedules are more robust than the baseline while matching its clean accuracy.
Comparison to grid search: By having to train a baseline model to estimate $\mathcal{L}$ before training with a PoE schedule, we have a 2× increase in training time. Grid search, on the other hand, is not theoretically motivated and is unlikely to obtain an optimal learning rate schedule in just 2 training rounds. Thus, our PoE-motivated approach is the clear winner.



(a) Clean accuracy vs epochs (b) PGD attack accuracy vs epochs

Fig. 3. Lipschitz constant estimated with extreme value theory for ResNet 20 standard training is 3.5258. Training with PoE-motivated ($\eta^1 = 1/\mathcal{L}_{\text{est}}$) and largest convergent ($\eta^1 = 2/\mathcal{L}_{\text{est}}$) schedules consistently increases PGD attack accuracy while matching clean accuracy of baseline (epochs 80-end).

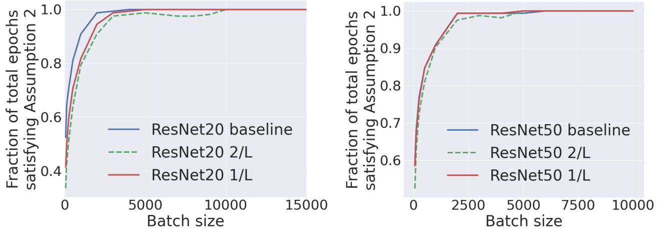


Fig. 4. Fraction of total epochs (164) where Assumption 2 holds vs batch size for ResNet20 [left], ResNet50 [right] standard training on CIFAR10 with all three schedules. The Assumption always holds for large batch sizes.

C. Batch size selection for Assumption 2

Following the discussion post Assumption 2 and with the parameters saved every epoch in Section IV-B, we monitor Inequality (6) in ResNet20, 50 standard training on CIFAR10 with θ^* set to the final parameters. Plotting the fraction of total epochs in which it holds against batch size in Figure 4, we see that it is indeed satisfied for all epochs when trained with large batch sizes but faces GPU memory constraints & clean accuracy degradation [29] with said large batch sizes. For the tradeoff, we find that a simple heuristic of starting with the smallest batch size at which Assumption 2 holds & decreasing until a clean accuracy threshold (here, 0.8 on CIFAR10, 0.6 on CIFAR100) is reached, works in practice.

V. Experimental Results

Following implementation details in Sections IV-B, IV-C, we perform standard training and AT with baseline, PoE-motivated and largest convergent schedules. The clean & adversarial accuracy (across 5 random seeds) & relevant parameters for standard training is presented in Table I; for AT in Table II. For standard training, we analyze ResNet20, 50, 110 [30] & DenseNet 40 [31] on CIFAR10 & the latter 3 on CIFAR100. The baseline training starts at $\eta^1 = 0.1$. We test each trained model on a 20-step PGD adversary with $\epsilon = \frac{1}{255}$ & step-size $\frac{0.1}{255}$. We also use the PyTorch baseline's standard weight decay 1e-4, momentum 0.9, and a step schedule with learning rate scaled down by 10 at epochs $81, 122$ for ResNets & $150, 225$ for DenseNet. For CIFAR10 AT, we train ResNet50 in PGD-AT framework [12]; WideResNet (WRN) 34-10 [32] in TRADES [20]; WRN 28-10 in RST [21]. PGD-AT & TRADES decay η by 10 every 50 epochs & once at the 75th epoch respectively. RST uses 500K additional unlabelled images from the TinyImages dataset [21] & a cosine annealing schedule. We follow the SOTA

TABLE I

STANDARD TRAINED MODELS ON CIFAR10 AND CIFAR100 EVALUATED ON 20-STEP PGD ATTACK WITH $\epsilon = 1/255$. BOLD AND UNDERLINED NUMBERS DENOTE THE BEST AND 2ND BEST PGD ATTACK ACCURACY IN EACH ROW.

Dataset	Model	baseline η^1		PoE-motivated $\eta^1 = 1/\mathcal{L}_{\text{est}}$ (Ours)		Largest convergent $\eta^1 = 2/\mathcal{L}_{\text{est}}$ (Ours)		Section IV-B, IV-C Parameters	
		Clean	PGD Attack	Clean	PGD Attack	Clean	PGD Attack	$\mathcal{L}_{\text{est}}$, (M, N), epochs	Batch
CIFAR 10	ResNet20	81.55 ± 2.6	24.69 ± 2.3	83.03 ± 0.2	28.72 ± 1.3	81.92 ± 1.3	**31.9 ± 3.1**	3.526, (200, 100), 164	5000
	ResNet50	84.44 ± 2.0	24.77 ± 2.5	84.29 ± 2.0	27.35 ± 2.4	84.24 ± 2.7	**35.82 ± 2.8**	10.79, (200, 164), 164	2000
	ResNet110	83.77 ± 0.5	28.81 ± 2.5	82.63 ± 0.9	**39.08 ± 2.5**	84.62 ± 0.4	34.22 ± 2.8	11.85, (200, 164), 164	1000
	DenseNet40	82.97 ± 2.8	12.11 ± 1.1	85.75 ± 2.4	14.81 ± 0.8	87.92 ± 2.1	**15.97 ± 1.6**	5.429, (100, 100), 300	2000
CIFAR 100	ResNet50	63.46 ± 4.4	6.9 ± 1.1	63.59 ± 4.1	7.56 ± 1.1	65.01 ± 4.0	**8.51 ± 1.4**	8.75, (200, 164), 164	256
	ResNet110	62.47 ± 4.2	9.43 ± 1.6	60.94 ± 4.1	12.52 ± 1.9	62.42 ± 4.8	**13.48 ± 3.6**	14.48, (200, 164), 164	256
	DenseNet40	60.0 ± 0.2	1.82 ± 0.1	60.0 ± 0.5	**2.11 ± 0.1**	61.97 ± 0.6	2.0 ± 0.1	10.47, (100, 100), 300	256

TABLE II

ADVERSARIAL TRAINED MODEL ON CIFAR10 EVALUATED ON AUTOATTACK WITH $\epsilon = 8/255$. (*) INDICATES EXTRA UNLABELED DATA USED. BOLD AND UNDERLINED NUMBERS DENOTE THE BEST AND 2ND BEST AUTOATTACK ACCURACY IN EACH ROW.

Approach; Model	Current SOTA		PoE-motivated $\eta^1 = 1/\mathcal{L}_{\text{est}}$ (Ours)		Largest convergent $\eta^1 = 2/\mathcal{L}_{\text{est}}$ (Ours)		Section IV-B, IV-C Parameters	
	Clean	Autoattack	Clean	Autoattack	Clean	Autoattack	$\mathcal{L}_{\text{est}}$, (M, N), epochs	Batch
TRADES; WRN34-10	84.81 ± .29	52.12 ± .09	84.51 ± .19	**52.56 ± .26**	83.44 ± .15	52.27 ± .04	7.497, (99, 25), 75	128
PGD-AT; ResNet-50	85.98 ± .09	42.66 ± .08	86.21 ± .32	**43.03 ± .13**	86.35 ± .11	42.98 ± .16	10.40, (55, 150), 150	256
RST(*); WRN28-10	89.48 ± .05	59.38 ± .14	89.5 ± .15	59.6 ± .11	89.48 ± .07	**59.7 ± .08**	13.46, (160, 200), 200	256

code of all three AT methods for other hyperparameters & test on Autoattack with $\epsilon = 8/255$.

VI. DISCUSSION AND FUTURE WORK

Table I shows a clear improvement in PGD attack accuracy over baseline (while maintaining similar clean accuracy) with both the largest convergent and PoE-motivated schedules. This demonstrates that our approach is promising in practical SGD. In Table II, across various AT frameworks/models evaluated on Autoattack (where small improvements are considered noteworthy), we have consistent increase in Autoattack accuracy over SOTA and continued clean accuracy similarity with our schedules. Thus, we note that, even with the varied dynamics of AT frameworks, our approach acts as a 'force-multiplier' for robustness.

Lastly, based on the success of the largest convergent schedule & our conservative upper bound, we conjecture that starting with a learning rate just below $\frac{2}{\mathcal{L}}$ also guarantees PoE of GD. An immediate next step includes proving the conjecture. In addition, future work can focus on extending our sufficient condition proof (for PoE of GD) to PoE of SGD and its variants.

REFERENCES

[1] C. Szegedy *et al.*, "Intriguing properties of neural networks," *arXiv preprint arXiv:1312.6199*, 2013.

[2] N. Carlini and D. Wagner, "Adversarial examples are not easily detected: Bypassing ten detection methods," in *Proceedings of the 10th ACM Workshop on Artificial Intelligence and Security*, 2017, pp. 3–14.

[3] A. Athalye, N. Carlini, and D. Wagner, "Obfuscated gradients give a false sense of security: Circumventing defenses to adversarial examples," in *International Conference on Machine Learning*, 2018.

[4] K. Nar and S. S. Sastry, "Persistency of excitation for robustness of neural networks," *arXiv preprint arXiv:1911.01043*, 2019.

[5] S. Sastry and M. Bodson, *Adaptive control: stability, convergence and robustness*. Courier Corporation, 2011.

[6] S. Lu and T. Basar, "Robust nonlinear system identification using neural-network models," *IEEE Transactions on Neural networks*, 1998.

[7] M. M. Polycarpou and P. A. Ioannou, *Identification and control of nonlinear systems using neural network models: Design and stability analysis*. Citeseer, 1991.

[8] G. Wood and B. Zhang, "Estimation of the lipschitz constant of a function," *Journal of Global Optimization*, vol. 8, pp. 91–103, 1996.

[9] T.-W. Weng *et al.*, "Evaluating the robustness of neural networks: An extreme value theory approach," *arXiv:1801.10578*, 2018.

[10] Y. LeCun *et al.*, "Gradient-based learning applied to document recognition," *Proceedings of the IEEE*, vol. 86, no. 11, 1998.

[11] A. Krizhevsky, G. Hinton, *et al.*, "Learning multiple layers of features from tiny images," 2009.

[12] A. Madry *et al.*, "Towards deep learning models resistant to adversarial attacks," *arXiv preprint arXiv:1706.06083*, 2017.

[13] F. Croce and M. Hein, "Reliable evaluation of adversarial robustness with an ensemble of diverse parameter-free attacks," in *ICML*, 2020.

[14] S. Srikant and M. Akella, "Persistence filter-based control for systems with time-varying control gains," *Systems and Control Letters*, 2009.

[15] D. Gorinevsky, "On the persistency of excitation in radial basis function network identification of nonlinear systems," *IEEE Transactions on Neural Networks*, vol. 6, no. 5, pp. 1237–1244, 1995.

[16] A. Kurdila, F. J. Narcowich, and J. D. Ward, "Persistency of excitation in identification using radial basis function approximants," *SIAM journal on control and optimization*, vol. 33, no. 2, pp. 625–642, 1995.

[17] M. Lecuyer *et al.*, "Certified robustness to adversarial examples with differential privacy," in *IEEE Symposium on Security & Privacy 2019*.

[18] J. Cohen *et al.*, "Certified adversarial robustness via randomized smoothing," in *International Conference on Machine Learning*, '19.

[19] I. J. Goodfellow, J. Shlens, and C. Szegedy, "Explaining and harnessing adversarial examples," *arXiv preprint arXiv:1412.6572*, 2014.

[20] H. Zhang *et al.*, "Theoretically principled trade-off between robustness and accuracy," in *International Conference on Machine Learning*, '19.

[21] Y. Carmon *et al.*, "Unlabeled data improves adversarial robustness," *arXiv preprint arXiv:1905.13736*, 2019.

[22] M. Fazlyab *et al.*, "Efficient and accurate estimation of lipschitz constants for deep neural networks," *arXiv:1906.04893*, 2019.

[23] C. Herrera *et al.*, "Estimating full lipschitz constants of deep neural networks," *arXiv preprint arXiv:2004.13135*, 2020.

[24] Y. Nesterov *et al.*, *Lectures on convex optimization*. Springer, 2018.

[25] D. P. Bertsekas, "Nonlinear programming," *Journal of the Operational Research Society*, vol. 48, no. 3, pp. 334–334, 1997.

[26] H. Li, Z. Xu, G. Taylor, C. Studer, and T. Goldstein, "Visualizing the loss landscape of neural nets," *arXiv preprint arXiv:1712.09913*, 2017.

[27] D. Wu *et al.*, "Adversarial weight perturbation helps robust generalization," *Advances in Neural Information Processing Systems*, 2020.

[28] L. Bottou, F. E. Curtis, and J. Nocedal, "Optimization methods for large-scale machine learning," *Siam Review*, 2018.

[29] N. S. o. Keskar, "On large-batch training for deep learning: Generalization gap and sharp minima," *arXiv:1609.04836*, 2016.

[30] K. He *et al.*, "Deep residual learning for image recognition," in *IEEE Conference on Computer Vision and Pattern Recognition*, 2016.

[31] G. Huang *et al.*, "Densely connected convolutional networks," in *IEEE Conference on Computer Vision and Pattern Recognition*, 2017.

[32] S. Zagoruyko and N. Komodakis, "Wide residual networks," *arXiv preprint arXiv:1605.07146*, 2016.

APPENDIX

Algorithm 1: Estimation of Certified Lipschitz Constant

```
Input  : All saved gradients and parameters
         $(\nabla l(\theta^i), \theta^i)$, M, N, intial shape choices
Output: $\mathcal{L}_{\text{est}}$
for shape$_0$ in initial shape choices do
    for j = 1, …, M do
        Sample $N$ points: $(\nabla l(\theta^i), \theta^i)_{i=1,\dots,N}$.
        Compute $N/2$ slopes between consecutive pairs of points:
            $s_i = \frac{||\nabla l(\theta^{i+1}) - \nabla l(\theta^i)||_2}{||\theta^{i+1} - \theta^i||_2}, \quad i = 1, 3, 5, ..., N.$
        Compute maximum of the $N/2$ slopes: $l_j = \max\{s_1, s_3, s_5, ..., s_N\}$.
    end for
    (shape, location, scale) $\leftarrow$ Fit three parameter reverse Weibull distribution to $\{l_1, \dots, l_M\}$ given initial shape value = shape$_0$.
    p-value $\leftarrow$ Kolmogrov-Smirnov goodness-of-fit test.
end for
return scale corresponding to largest p-value
```

VII. Estimation of Certified Lipschitz Constant $\mathcal{L}$ via Extreme Value Theory

We detail the estimation algorithm previously discussed in Section IV-B in Algorithm 1. We depict our heuristic for (M, N) tuning in Figure 5 for a ResNet-20 model [30] and a significance value of $\alpha = 0.55$. The row corresponding to $(M, N) = (200, 100)$ in Figure 5(a) is the only one that has both, p-values larger & lesser than α (satisfying our heuristic) and the largest of these corresponds to a Lipschitz constant of 3.5258 (see red box on Figure 5(b)). Thus, our estimated Lipschitz constant is $\mathcal{L}_{\text{est}} = 3.5258$. Please find the complete extended version of Figure 5(a), *i.e.* a complete heat map of all (M,N) tuples vs p-values in Figure 6.

Analysis of time complexity and memory overhead. Algorithm 1 maintains a $O(mn)$ time complexity which is negligible in comparison to the model training time (m, n are the number of values of hyperparamters M, N tried in Algorithm 1). Our primary increase in training time is a consequence of having to train a baseline model and then another model with a persistently exciting schedule. This results in a 2× increase in time complexity. We hope future work can help boost performance (for example, by adapting learning rates online to satisfy PoE conditions). We also note that there is a small memory overhead in having to save gradients plus parameters after every epoch for use in Lipschitz estimation post training. This overhead is given by $O(n_{\text{epochs}}(n_{\text{params}} + n_{\text{pixels}}))$ where n_{epochs}, n_{params}, and n_{pixels} denote the number of epochs, model parameters and input image pixels respectively.

Limitations of the estimation algorithm. The estimation algorithm is inherently random because it depends on the gradients and parameters saved during the training process which can change with each run even when using the same random seed. Yet, the advantage of our results are that future work can introduce a better estimation algorithm (preferably with less inherent randomness) for $\mathcal{L}$ and use it in conjunction with our PoE-motivated or largest convergent learning rate schedule for increased adversarial robustness.

VIII. Details of Adversarial Training Frameworks and Autoattack

We describe the adversarial training frameworks analyzed in this work and the autoattack benchmark used to evaluate models trained in said frameworks below.

PGD-AT [12]: The general adversarial training min-max optimization problem is given by

$$\arg\min_{\theta} \underset{(X,Y)\in\mathcal{X}\times\mathcal{Y}}{\mathbb{E}} \left[\max_{\delta\in\mathbb{S}} L(h_\Theta(X+\delta), Y)\right]$$

where $\mathbb{S}_p = \{\delta \mid ||\delta||_p < \epsilon\}$, X, Y represent batch training data & labels and the rest of the notation is defined in Section II. We are primarily concerned with l_∞ perturbations in this work which is why we have $\mathbb{S} = \mathbb{S}_\infty$. The inner maximization is solved by projected gradient descent (PGD) on the negative loss function (for K steps with α step size) to get an adversarial example represented as $X^{(K)} = X + \delta^{(K)}$. The perturbed data point in the $(t+1)$-th step (*i.e.* $X^{(t+1)}$) is given by

$$X^{(t+1)} = \prod_{X+\mathbb{S}} (X^{(t)} + \alpha \operatorname{sgn}(\nabla_X L(h_\Theta(X^{(t)}), Y)))$$

with initialization $X^{(0)} = X + \delta^{(0)}$ where $\delta^{(0)}$ can be set to 0 or to any random point within $\mathbb{S}$. The latter case is called PGD with random initialization. The $\prod_{x+\mathbb{S}}$ denotes projecting perturbations of perturbed data points into the set $\mathbb{S}$.

TRADES [20]: In TRADES, a theoretically motivated surrogate loss that balances the trade-off between standard and robust accuracy is minimized. The TRADES loss function is given by,

$$L_\Theta^{\text{TRADES}} = L(h_\Theta(X), Y) + \beta \max_{\delta\in\mathbb{S}} D_{\text{KL}}(h_\Theta(X+\delta)||h_\Theta(X))$$

where D_{KL} represents Kullback–Leibler (KL) divergence and β is a hyperparameter that controls the aforementioned trade-off.

RST [21]: In RST, a separate standard model is trained over CIFAR10 and used to generate pseudo-labels for unlabelled images from the TinyImages dataset [21]. Then a robust model is trained over the unlabelled data and its pseudo-labels by minimizing the TRADES loss given above. By this self-supervised training process, an adversarial-trained robust classifier is obtained.

Autoattack [13]: Autoattack consists of 4 attacks – Auto-PGD on cross entropy loss (white-box), Auto-PGD on difference of logits ratio loss (also white-box), Fast adaptive

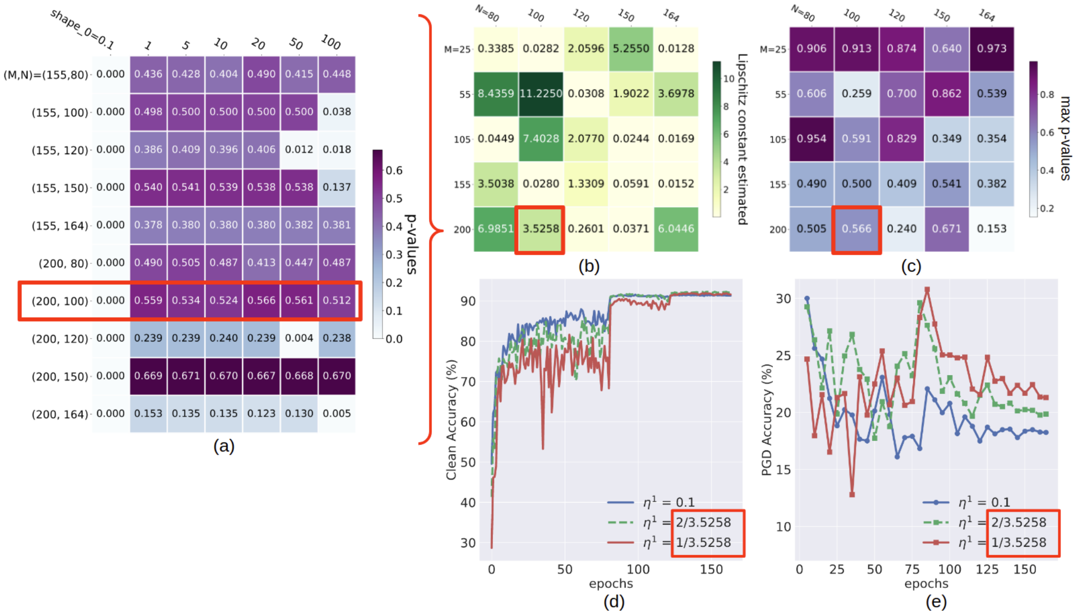


Fig. 5. Estimating $\mathcal{L}$ for ResNet-20 model in standard classifier training using initial shape choices = $\{0.1, 1, 5, 10, 20, 50, 100\}$, $M \in \{25, 55, 105, 155, 200\}$, $N \in \{80, 100, 120, 150, 164\}$, and significance value $\alpha = 0.55$. Here, (a) shows a heat map of p-values for some (M, N) tuples vs initial shape (shape_0) values; (b) shows Lipschitz constant estimates for all M, N values in a heat map; (c) depicts their corresponding max p-values (also in a heat map); (d) and (e) are reproductions of Figure 3 and depict the variation of clean and PGD attack accuracy as a function of epochs for all three schedules (baseline, PoE-motivated and largest convergent).

boundary attack (black-box) and Square attack (also black-box). Evaluation on autoattack has very little (0.01%) to no variance in different runs. Moreover, it has only one hyperparameter ϵ (usually set to $8/255$) while all others are fixed and abstracted away from the evaluation making comparison across models and frameworks easy.

Hyperparameters for Adversarial Training: We set momentum to 0.9 in all three frameworks and set weight decay to 5e-4 in PGD-AT & RST; 2e-4 in TRADES. The β parameter in the TRADES formulation of adversarial loss (which is also used in RST) is set to 0.6. It does not exist for the adversarial loss in PGD-AT. The same perturbation budget of $\epsilon = 8/255$, attack steps = 10, and attack step-size of 0.007 are used in all three methods. These hyperparameters are obtained from the current SOTA of the three frameworks as given in [12], [20], [21].

IX. Additional Details of Experiments

Data Augmentation for Standard and Adversarial Training: Following the common practice for CIFAR datasets (and following the SOTA implementations of all 4 adversarial training frameworks), training images are augmented with random crops (padding by 4 pixels and cropping to 32 × 32) and random horizontal flips.

Computation resources used in running experiments: We ran the experiments on either two Nvidia GeForce RTX 3090 GPUs (each with 24 GB of memory) or two Nvidia Quadro RTX 6000 GPUs (each with 24 GB of memory). The CPUs used were Intel Xeon Gold processors @ 3 GHz.

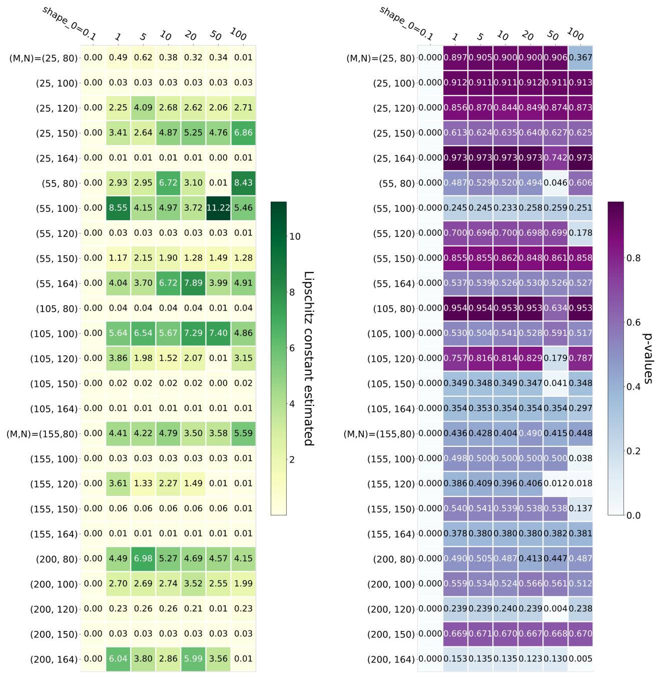


Fig. 6. [LEFT] heat map of Lipschitz constants estimated (*i.e.* fitted scale parameter) for various (M, N) tuples vs various initial shape parameters. [RIGHT] heat map of corresponding p-values for various (M, N) tuples vs various initial shape parameters.

Code bases utilized: The code for LeNet5 on MNIST is based on https://github.com/ChawDoe/LeNet5-MNIST-PyTorch (No license). Standard training in CIFAR10, CIFAR100 for all models is based on code from a repository of

PyTorch baselines at https://github.com/bearpaw/pytorch-classification (MIT license). We used the Advertorch python library at https://github.com/BorealisAI/advertorch (GNU general public license) for PGD implementation in standard training.

In adversarial training, the code for PGD-AT framework [12] is from https://github.com/MadryLab/robustness (MIT license), the code for TRADES [20] framework is from https://github.com/yaodongyu/TRADES (MIT license) and the code for RST [21] is from https://github.com/yaircarmon/semisup-adv (MIT license). Our Lipschitz constant estimation code is based on previous work by [9] and can be found at https://github.com/huanzhang12/CLEVER (Apache license).