Robust Deep Learning as Optimal Control: Insights and Convergence Guarantees

Jacob H. Seidman SEIDJ@SAS.UPENN.EDU
Mahyar Fazlyab MAHYARFA@SEAS.UPENN.EDU
Victor M. Preciado PRECIADO@SEAS.UPENN.EDU
George J. Pappas PAPPASG@SEAS.UPENN.EDU

Editors: A. Bayen, A. Jadbabaie, G. J. Pappas, P. Parrilo, B. Recht, C. Tomlin, M.Zeilinger

Abstract

The fragility of deep neural networks to adversarially-chosen inputs has motivated the need to revisit deep learning algorithms. Including adversarial examples during training is a popular defense mechanism against adversarial attacks. This mechanism can be formulated as a min-max optimization problem, where the adversary seeks to maximize the loss function using an iterative first-order algorithm while the learner attempts to minimize it. However, finding adversarial examples in this way causes excessive computational overhead during training. By interpreting the min-max problem as an optimal control problem, it has recently been shown that one can exploit the compositional structure of neural networks in the optimization problem to improve the training time significantly. In this paper, we provide the first convergence analysis of this adversarial training algorithm by combining techniques from robust optimal control and inexact oracle methods in optimization. Our analysis sheds light on how the hyperparameters of the algorithm affect the its stability and convergence. We support our insights with experiments on a robust classification problem.

Keywords: Adversarial training, optimal control, maximum principle, robust optimization

1. Introduction

Deep neural networks have repeatedly demonstrated their capacity to achieve state of the art performance on benchmark machine learning problems LeCun et al. (2015). However, their performance can be significantly affected by small input perturbations that can drastically change the network's output Szegedy et al. (2013). In safety-critical applications the cost of such errors is prohibitive. Therefore, an important line of work has emerged to train deep neural networks to be robust to adversarially-chosen perturbations.

Among the most empirically successful methods is an optimization-based approach, where adversarial training is formulated as a min-max non-convex optimization problem Madry et al. (2018). To solve this problem, the adversary seeks to maximize the loss over sets of admissible perturbations, typically using an iterative method such as Projected Gradient Descent (PGD) Madry et al. (2018), the Fast Gradient Sign method (FGSM) Goodfellow et al. (2015), or other methods Carlini and Wagner (2017). The learner's goal is then to minimize the worst-case loss, as computed by the adversary, over the parameters of the neural network. In practice, however, the adversary can only approximate the worst-case loss. Additionally, each iteration of the adversary requires one

backpropagation through the network. This results in a multiplicative factor increase in the number of backpropagations needed for training, which can significantly increase the total training time.

Nevertheless, it was shown in Zhang et al. (2019) that the computational cost for the adversary can be significantly reduced by exploiting the inherent compositional structure of deep neural networks. In particular, by viewing a T-layer neural network as a discrete-time dynamical system with time horizon T, the min-max robust optimization problem can be seen as a finite-horizon robust optimal control problem. In this interpretation, the adversary is finding the worst-case additive perturbation to the initial condition of the system (this is a special case of the H_∞ control problem Başar and Bernhard (2008)). The learner then minimizes the worst-case cost function over the parameters of the network. Deriving the necessary conditions for this robust control problem from the Pontryagin Maximum Principle (PMP) leads to algorithm proposed in Zhang et al. (2019). While the algorithm is empirically very successful, its convergence analysis has not yet been addressed.

In this paper, we give the first convergence proof of this optimal control inspired robust training algorithm. By viewing the adversary updates as being derived from the costate process of the deep network dynamics, we bound the error from the adversary's updates to its true gradients. This allows us to appeal to results on first order methods with inexact oracles and prove a convergence result for this algorithm which explicitly shows the dependence on the algorithm parameters. The argument we construct provides an outline for future results on the convergence of computationally efficient robust training algorithms. Our result further suggest that for a fixed number of backpropagations, increasing the number of adversary updates past a certain point can have a negative effect on performance. This insight is supported by experiments on a robust classification problem.

Preliminaries and Notation: We denote by $\mathbb{R}^d$ the set of d-dimensional vectors with real valued components. The inner product is denoted $\langle \cdot, \cdot \rangle$ and the 2-norm is denoted $\| \cdot \|$. We say a function f is L-smooth if it has L-Lipschitz gradients. For $\mu > 0$, a differentiable function f is μ-strongly concave if for all x, y, $f(x) \leq f(y) + \langle f(y), x - y \rangle - (\mu/2)\|x - y\|^2$. If a function f is μ-strongly concave and L-smooth, then for all x, y, $\|x - x^\star\| \leq 1/\mu \|\nabla f(x)\| \leq (1/\mu)\sqrt{2L(f(x^\star) - f(x))}$, where $x^\star = \operatorname{argmax}_x f(x)$ Boyd and Vandenberghe (2004). For a compact set $\mathcal{X}$ we define its diameter as $D(\mathcal{X}) := \max_{x,x' \in \mathcal{X}} \|x - x'\|$.

2. Robust Training Problem Formulation

Consider a T-layer deep neural network with hidden dimensions $n_1, \ldots, n_T$ described by $F(x, \theta)$: $\mathbb{R}^{d_x} \times \mathbb{R}^{d_\theta} \to \mathbb{R}^{d_y}$, where x is the input and θ are the trainable parameters. We overload notation slightly and let the "0-th" layer have dimension $n_0 = d_x$ and the output layer have dimension $n_T = d_y$. Given a norm-based perturbation ball $\mathcal{X}$ and a training dataset $\mathcal{S} = \{(x_{0,1}, y_1) \ldots (x_{0,S}, y_S)\}$ of size S, the robust training problem can be formulated as (Madry et al., 2018)

$$\underset{\theta}{\text{minimize}} \sum_{i=1}^{S} \max_{\eta_i \in \mathcal{X}} \Phi(F(x_{0,i} + \eta_i, \theta), y_i), \tag{1}$$

where $\Phi : \mathbb{R}^{n_T} \times \mathbb{R}^{d_y} \to \mathbb{R}$ is a convex surrogate loss function penalizing the difference between the predicted and true labels. Throughout we will reserve i as the data index.

3. An Optimal Control Inspired Algorithm

Due to their compositional structure, feed-forward deep neural networks can be viewed as dynamical systems. This approach has been taken recently in a number of papers which explore these dynamics and use the interpretation to suggest new training algorithms Weinan (2017); Li et al. (2017); Li and Hao (2018); Weinan et al. (2019); Zhang et al. (2019). Explicitly, we can describe a T-layer deep neural network $F(x,\theta)$ by the recursion $x_{t+1} = f_t(x_t,\theta_t)$, $t = 0,\cdots,T-1$, where $x_t \in \mathbb{R}^{n_t}$ are the states (the output of the t-th layer), $f_t\colon \mathbb{R}^{n_t}\times\mathbb{R}^{m_t}\to\mathbb{R}^{n_{t+1}}$ is the state transition map, $\theta_t \in \mathbb{R}^{m_t}$ are the trainable control parameters, θ is the concatenation of $(\theta_i)_{0\le T-1}$[1], and the initial conditions are given by the inputs to the network, $x_{0,i}$. Expressing the neural network as a dynamical system allows us to rewrite problem (1) as the following optimal control problem:

$$\begin{aligned} \underset{\theta_1,\ldots,\theta_T}{\text{minimize}}\ \underset{\eta_1,\ldots\eta_S}{\text{maximize}} \quad & \sum_{i=1}^{S}\Phi(x_{T,i},y_i)+\sum_{i=1}^{S}\sum_{t=0}^{T-1}R_t(x_{t,i},\theta_t) \\ \text{subject to} \quad & x_{t+1,i} = f_t(x_{t,i},\theta_t), \quad i=1,\ldots,S, \quad t=1,\ldots,T-1 \\ & x_{1,i} = f_0(x_{0,i}+\eta_i,\theta_0). \quad i=1,\ldots,S \end{aligned} \tag{2}$$

where R_t is a potential regularizer on the states and controls for the t-th layer. The two-player Pontryagin Maximum principle, proved in Zhang et al. (2019) gives necessary conditions for an optimal setting of the parameters $\theta^\star$, perturbations $\eta_1^\star,\ldots,\eta_S^\star$, and corresponding trajectories $\{x_{t,i}^\star\}$. Define the Hamiltonians

$$H_t(x,p,\theta) := p^\top f_t(x,\theta) - R_t(x,\theta), \quad t=1,\ldots,T-1 \tag{3}$$

$$H_0(x,p,\theta,\eta) := p^\top f_0(x+\eta,\theta) - R_0(x,\theta). \tag{4}$$

The two player maximum principle says in this case that if Φ, f_t, and R_t are twice continuously differentiable, with respect to x, uniformly bounded in x and t along with their partial derivatives, and the image sets $\{f_t(x,\theta)\mid\theta\in\mathbb{R}^{d_\theta}\}$ and $\{R_t(x,\theta)\mid\theta\in\mathbb{R}^{d_\theta}\}$ are convex for all x and t, then there exists an optimal costate trajectory $p_{t,i}^\star$ such that the following dynamics are satisfied

$$x_{t+1,i}^\star = \nabla_p H_t(x_{t,i}^\star,p_{t+1,i}^\star,\theta_t^\star), \quad x_{1,i}^\star = \nabla_p H_0(x_{0,i},p_{1,i}^\star,\theta_0^\star,\eta_i^\star) \tag{5}$$

$$p_{t,i}^\star = \nabla_x H_t(x_{t,i}^\star,p_{t+1,i}^\star,\theta_t^\star), \quad p_{T,i}^\star = -\nabla_x\Phi(x_{T,i}^\star,y_i), \tag{6}$$

and the following Hamiltonian condition for all $\theta_t\in\mathbb{R}^{d_{\theta_t}}$ and $\eta_i\in\mathcal{X}$

$$\sum_{i\in\mathcal{S}} H_t(x_{t,i}^\star,p_{t+1,i}^\star,\theta_t) \le \sum_{i\in\mathcal{S}} H_t(x_{t,i}^\star,p_{t+1,i}^\star,\theta_t^\star), \quad t=1,\ldots,T-1 \tag{7}$$

$$\sum_{i\in\mathcal{S}} H_0(x_{t,i}^\star,p_{t+1,i}^\star,\theta_t,\eta_i^\star) \le \sum_{i\in\mathcal{S}} H_0(x_{t,i}^\star,p_{t+1,i}^\star,\theta_t^\star,\eta_i^\star) \le \sum_{i\in\mathcal{S}} H_0(x_{t,i}^\star,p_{t+1,i}^\star,\theta_t^\star,\eta_i). \tag{8}$$

These necessary optimality conditions can be used to design an iterative algorithm of the following form. For each data point $i\in\{1,\cdots,S\}$,

1. With this representation, the input-output map of the neural network is $F(x,\theta) = f_{T-1}(f_{T-2}(\cdots f_0(x,\theta_0)\cdots),\theta_{T-2})\theta_{T-1})$.

1. Compute the state and costate trajectories $\{x_{i,t}\}$ and $\{p_{i,t}\}$ from (9), keeping θ_t and η_i fixed:

$$x^{\eta}_{t+1,i} = \nabla_p H_t(x^{\eta}_{t,i}, p^{\eta}_{t+1,i}, \theta_t), \quad x^{\eta}_{1,i} = \nabla_p H_0(x_{0,i}, p^{\eta}_{1,i}, \theta_0, \eta) \tag{9}$$

$$p^{\eta}_{t,i} = \nabla_x H_t(x^{\eta}_{t,i}, p^{\eta}_{t+1,i}, \theta_t), \quad p^{\eta}_{T,i} = -\nabla_x \Phi(x^{\eta}_{T,i}, y_i). \tag{10}$$

2. Minimize the Hamiltonian $H_0(x_{t,i}, p_{t+1,i}, \theta_t, \eta_i)$ with respect to η_i.

3. Maximize the sum of Hamiltonians $\sum_{i\in\mathcal{S}} H_t(x_{t,i}, p_{t+1,i}, \theta_t)$ with respect to θ_t for all t.

As was noticed as early as LeCun et al. (1988), it can be seen from the chain rule that the backward costate dynamics in (10) are equivalent to backpropagation through the network. With this interpretation, the gradient of the total loss for the i-th data point with respect to the adversary η_i can be written as $\nabla_\eta f_0(x_{0,i} + \eta_i, \theta_0)^\top p^{\eta_i}_{1,i}$. For a fixed value of θ_0, performing gradient descent on H_0 to find a worst-case adversarial perturbation can be expressed as the following updates, where $\alpha > 0$ is a step size and we have for the moment dropped the dependence on the data index i.

$$\eta^{\ell+1} = \eta^{\ell} - \alpha \nabla_\eta f_0(x_0 + \eta^{\ell}, \theta_0)^\top p^{\eta^{\ell}}_1. \tag{11}$$

An important observation made in Zhang et al. (2019) is that the adversary is only present in the first layer Hamiltonian condition and this function can be minimized by computing gradients only with respect to the first layer of the network. More explicitly, instead of using $p^{\eta^{\ell}}_1$, as in the updates in (11), we could instead use $p^{\eta^{0}}_1$ and the updates

$$\eta^{\ell+1} = \eta^{\ell} - \alpha \nabla_\eta f_0(x_0 + \eta^{\ell}, \theta_0)^\top p^{\eta^{0}}_1. \tag{12}$$

This removes the need to do a full backpropagation to recompute the costate $p^{\eta^{\ell}}_1$ for every update of η^{ℓ}, at the cost of now being an approximate gradient. In other words, we work with "frozen gradients" of the later layers. This inspires the "YOPO-m-n" (You Only Propogate Once) algorithm in Zhang et al. (2019), where the adversary is updated with m full backpropagations, after each of which n updates of the form (12) are performed. A modified version of this method is written in pseudocode with the Hamiltonian framework in mind in Algorithm 1. While in Zhang et al. (2019), Algorithm 1 was shown to have very promising empirical results, in this paper we provide a rigorous convergence analysis of its behavior.

4. Convergence Analysis of Adversarial Training

To prove convergence we interpret Algorithm 1 as consisting of two nested gradient methods with inexact gradient oracles. The inner method finds an adversarial perturbation by performing gradient descent on the Hamiltonian H_0 with frozen gradients at layers 2 through T, or equivalently, with a frozen costate p_1. Not updating this costate at every iteration is what creates the oracle error for the adversary's problem. By bounding the difference of the frozen costate to the actual costate, we are able to bound the oracle error and appeal to known inexact oracle convergence results for the adversary's problem.

The outer method then makes a parameter update to the network based on the perturbation found by the inner method. If the inner method found the true worst case perturbation, this would result in an exact gradient update. However, since in general the adversary's inner method will not converge

Algorithm 1: You Only Propagate Once (YOPO-m-n) Robust Traning Algorithm

Initialize θ^0 randomly;
for $k = 1, 2, \ldots$ **do**
  Randomly select mini-batch $\mathcal{B}$;
  Randomly initialize $\eta_i^{0,0} \in \mathcal{X}$, $i \in \{1, \ldots, B\}$;
  for $j = 0, \ldots, m-1$ **do**
    $x_{1,i} \leftarrow \nabla_p H_0(x_{0,i}, p_{1,i}, \theta_0, \eta_i^{j,0}), \quad i \in \{1, \ldots, B\}$;
    for $t = 1, \ldots, T-1$ **do**
      $x_{t+1,i} \leftarrow \nabla_p H_t(x_{t,i}, p_{t+1,i}, \theta_t)$;
    end
    $p_{i,T} \leftarrow -\frac{1}{B}\nabla\Phi(x_{T,i}, y_i), \quad i \in \{1, \ldots, B\}$;
    for $t = T-1, \ldots, 1$ **do**
      $p_{t,i} \leftarrow \nabla_x H_t(x_{t,i}, p_{t+1,i}, \theta_t), \quad i \in \{1, \ldots, B\}$;
    end
    for $\ell = 0, \ldots, n-1$ **do**
      $\eta_i^{j,\ell+1} \leftarrow \Pi_{\mathcal{X}}\left[\eta_i^{j,\ell} - \alpha\nabla_\eta H_0(x_{0,i}, p_{1,i}, \theta_0, \eta_i^{j,\ell})\right], \quad i \in \{1, \ldots, B\}$;
    end
  end
  $\theta^{k+1} \leftarrow \theta^k - \gamma_t \frac{1}{B}\sum_{i=1}^{B} \nabla_\theta \Phi(F(x_{i,0} + \eta_i^{m,n}, \theta^k), y_i)$;
end

to the true optimal point in finitely many iterations, and is an inexact method itself, the update for the network parameters can also be seen as coming from an inexact gradient oracle. Using the convergence result for the adversary then lets us bound the oracle error for the outer method, and we can then complete the proof with known techniques for convergence of gradient descent on non-convex functions with an inexact oracle. All proofs are deferred to the appendix of the extended version of this paper Seidman et al. (2019).

We first set up some notation. For a given data point i, let $\mathcal{A}_i(\eta, \theta) := \Phi(F(x_{0,i} + \eta, \theta), y_i)$. Let $\eta_i^\star(\theta) := \operatorname{argmax}_\eta \mathcal{A}_i(\eta, \theta)$. We define the robust loss function $\mathcal{R}(\theta) := (1/S)\sum_{i=1}^{S} \mathcal{A}_i(\eta_i^\star(\theta), \theta)$. Let $\mathcal{B}$ indicate a sampled mini-batch of the data of size B. Let $g_{\mathcal{B}}(\theta) = (1/B)\sum_{i\in\mathcal{B}} \nabla_\theta \mathcal{A}(\eta_i^\star(\theta), \theta)$ denote the corresponding stochastic gradient of the robust loss. Note that $\mathbb{E}[g_{\mathcal{B}}(\theta)] = \nabla_\theta \mathcal{R}(\theta)$ where the expectation is taken over the randomness of the mini-batch sampling.

We now present the assumptions that will be in place for the theoretical results of this paper.

Assumption 1 *There exists a constant $K > 0$ such that for all $t \in 1, \ldots, T$, the functions f_t, Φ, $\nabla_x f_t$, and $\nabla_x R_t$ are K-Lipschitz in x, uniformly in θ. For all $i = 1, \ldots, S$, the functions $\nabla_\theta \mathcal{A}_i$ and $\nabla_\eta \mathcal{A}_i$ satisfy the following Lipschitz conditions,*

$$\|\nabla_\theta \mathcal{A}_i(\eta, \theta^1) - \nabla_\theta \mathcal{A}_i(\eta, \theta^2)\| \leq L_{\theta\theta}\|\theta^1 - \theta^2\| \tag{13}$$

$$\|\nabla_\theta \mathcal{A}_i(\eta^1, \theta) - \nabla_\theta \mathcal{A}_i(\eta^2, \theta)\| \leq L_{\theta\eta}\|\eta^1 - \eta^2\| \tag{14}$$

$$\|\nabla_\eta \mathcal{A}_i(\eta, \theta^1) - \nabla_\eta \mathcal{A}_i(\eta, \theta^2)\| \leq L_{\eta\theta}\|\theta^1 - \theta^2\| \tag{15}$$

$$\|\nabla_\eta \mathcal{A}_i(\eta^1, \theta) - \nabla_\eta \mathcal{A}_i(\eta^2, \theta)\| \leq L_{\eta\eta}\|\eta^1 - \eta^2\| \tag{16}$$

Such Lipschitz assumptions are standard in the optimization literature. Note that the assumption of the existence of the gradients in x and η of functions of the network restricts the potential activation functions of the network to not include the ReLU function, though it does allow for sigmoid, tanh, and ELU activations. Leveraging these smoothness assumptions will be essential in proving the rate of convergence that follows.

Assumption 2 *$\mathcal{A}_i(\eta, \theta)$ is locally μ-strongly concave for $\eta \in \mathcal{X}$, that is for any θ and $\eta^1, \eta^2 \in \mathcal{X}$,*

$$\mathcal{A}_i(\eta^1, \theta) \leq \mathcal{A}(\eta^2, \theta) + \langle \mathcal{A}_i(\eta^2, \theta), \eta^1 - \eta^2 \rangle - \frac{\mu}{2}\|\eta^1 - \eta^2\|^2. \tag{17}$$

This assumption was made in previous results on convergence of robust training Wang et al. (2019) and is justified through the reformulation of robust training as distributionally robust optimization Sinha et al. (2018); Lee and Raginsky (2018). Perturbing each data point in the ℓ^p norm by ϵ results in perturbing the empirical distribution in the p-Wasserstein distance by at most ϵ.

Assumption 3 *The stochastic gradients satisfy $\mathbb{E}\left[\|g_{\mathcal{B}}(\theta) - \nabla\mathcal{R}(\theta)\|^2\right] \leq \sigma^2$, with $\sigma \geq 0$.*

This assumption is standard in convergence results for optimization algorithms with noisy gradients. It was shown in Sinha et al. (2018) that under these assumptions the robust loss function has Lipschitz gradients and the following relation holds. This will allow us to use techniques for convergence of gradient descent on non-convex functions.

Proposition 1 (Sinha et al. (2018)) *Under Assumptions 1 and 2, the robust loss function $\mathcal{R}(\theta)$ is L-smooth, where $L = L_{\theta\theta} + (L_{\theta\eta}L_{\eta\theta}/\mu)$, and the following inequality holds for all θ_1, θ_2,*

$$\mathcal{R}(\theta_1) \leq \mathcal{R}(\theta_2) + \langle \nabla\mathcal{R}(\theta_2), \theta_1 - \theta_2 \rangle + \frac{L}{2}\|\theta_1 - \theta_2\|^2. \tag{18}$$

We next derive the following three results used to prove our main theorem. The first result bounds the difference between the costate used for the adversary's update, as in (12), and the costate that would result in a true gradient update, as in (11). The proof shows that the costates are Lipschitz as a function of the initial condition of the system, and then uses a bound on successive values of the perturbation η from the adversary's updates.

Lemma 2 *There exists a constant C' dependent on T and K such that for all $\ell \in \{0, \ldots, n-1\}$, $j \in \{0, \ldots, m\}$, and $i \in \{1, \ldots, S\}$*

$$\|p_{1,i}^{\eta_i^{j,0}} - p_{1,i}^{\eta_i^{j,\ell}}\| \leq C'\alpha(n-1). \tag{19}$$

Hence, we are able to bound the error incurred from the frozen costates of the adversary's updates to the true gradients. In doing so, we can appeal to convergence results for inexact oracles and prove the next theorem on convergence of the adversary to a worst case perturbation.

Theorem 3 *Under Assumptions 1, 2, and 3, for a fixed value of θ and fixed data point i, let*

$$\hat{\eta}_i = \underset{\substack{j=1,\ldots,m \\ \ell=1,\ldots,n}}{\operatorname{argmin}} \|\nabla_\eta \mathcal{A}_i(\eta_i^{j,\ell}, \theta)\|. \tag{20}$$

Then, if we define $C = KC'$ and set $\alpha < 1/L_{\eta\eta}$, then

$$\|\nabla_\eta \mathcal{A}_i(\hat{\eta}_i, \theta)\|^2 \leq D(\mathcal{X})L_{\eta\eta}^2\left(1 - \frac{\mu}{L_{\eta\eta}}\right)^{mn+1} + \frac{2C^2}{L_{\eta\eta}}(n-1)^2\left(\frac{2}{\mu} + \frac{1}{2L_{\eta\eta}}\right), \tag{21}$$

The last intermediate result we use to prove our theorem relates how the suboptimality of the chosen adversarial perturbation bounds the error for the computed gradients of the robust loss. To prove our main theorem we will apply the bound from the previous result to the following lemma.

Lemma 4 *Under Assumptions 1 and 2, if η_i are such that $(1/B)\sum_{i\in\mathcal{B}}\|\nabla_\eta \mathcal{A}_i(\eta_i,\theta)\|^2 \leq \delta$ then*

$$\left\| \frac{1}{B}\sum_{i\in\mathcal{B}} \nabla_\theta \mathcal{A}_{\mathcal{B}}(\eta_i,\theta) - g_{\mathcal{B}}(\theta) \right\| \leq \frac{L_{\theta\eta}\delta}{\mu}. \tag{22}$$

Combining the previous three results allows us to prove the main theorem, stated below.

Theorem 5 (Convergence Analysis of Adversarial Training) *Under Assumptions 1, 2, and 3, if the the step sizes γ_t satisfy $\gamma_t = \gamma = \min\{1/L, \sqrt{\Delta/(L\sigma^2 N)}\}$ where $\Delta = \mathcal{R}(\theta^0) - \inf_\theta \mathcal{R}(\theta)$, $\alpha < 1/L_{\eta\eta}$, and the parameters are updated with the perturbations $\eta_i = \underset{j\in\{1,\dots,m\},\ell\in\{1,\dots,n\}}{\operatorname{argmin}} \|\nabla_\eta \mathcal{A}_i(\eta_i^{j,\ell},\theta)\|$, then there exists a constant C depending on T and K such that the iterates of YOPO-m-n satisfy*

$$\frac{1}{N}\sum_{k=1}^{N}\mathbb{E}\left[\|\nabla\mathcal{R}(\theta^k)\|^2\right] \leq 4\sigma\sqrt{\frac{L\Delta}{N}} + \frac{5L_{\theta\eta}^2}{\mu}\left(D(\mathcal{X})L_{\eta\eta}^2\left(1-\frac{\mu}{L_{\eta\eta}}\right)^{mn+1} + \frac{2C^2}{L_{\eta\eta}}\left(\frac{2}{\mu}+\frac{1}{2L_{\eta\eta}}\right)(n-1)^2\right). \tag{23}$$

The first term on the right side is typical for convergence of first order optimization algorithms on smooth non-convex functions, and is the same as the first term that appears in the convergence result of Wang et al. (2019). The second term represents the errors from both inexact oracles that accumulate over the algorithm. The expression

$$\mathcal{E}(m,n) := D(\mathcal{X})L_{\eta\eta}^2\left(1-\frac{\mu}{L_{\eta\eta}}\right)^{mn+1} + \frac{2C^2}{L_{\eta\eta}}\left(\frac{2}{\mu}+\frac{1}{2L_{\eta\eta}}\right)(n-1)^2, \tag{24}$$

shows how the solution to the adversary's problem contributes to the gradient oracle error for the parameter updates. The first term represents the approximate nature of the adversary's solution, as it has finitely many iterations to maximize the loss function. The second term shows the accumulation of gradient oracle errors for the adversary due to freezing the costate in between backpropagations.

Using this bound we can investigate the dependence of the algorithm on the number of back-propagations for the adversary, m, and the number of gradient steps taken with each frozen gradient, n. We see that $\mathcal{E}$ monotonically decreases in m, implying that a practitioner should set m to be as large as can be tolerated according to their computational budget. Thus, we will focus on the dependence of $\mathcal{E}$ on the number of adversary updates per backpropagation, n.

First we note that $\mathcal{E}$ is convex in n, as can be confirmed by computing its second partial derivative and observing $\partial^2\mathcal{E}/\partial n^2 \geq 0$. As $\partial\mathcal{E}/\partial n$ is monotonically increasing, we should only increase n up to just before the point where $\partial\mathcal{E}/\partial n$ becomes positive. This happens when

$$-\log\left(1-\frac{\mu}{L_{\eta\eta}}\right)D(\mathcal{X})L_{\eta\eta}^2 m\left(1-\frac{\mu}{L_{\eta\eta}}\right)^{mn+1} \leq \frac{4C^2}{L_{\eta\eta}}\left(\frac{2}{\mu}+\frac{1}{2L_{\eta\eta}}\right)(n-1), \tag{25}$$

that is, when the exponentially decaying factor in n on the left side overtakes the linearly growing factor in n on the right side. Therefore, our bound suggests that when n is too large we will

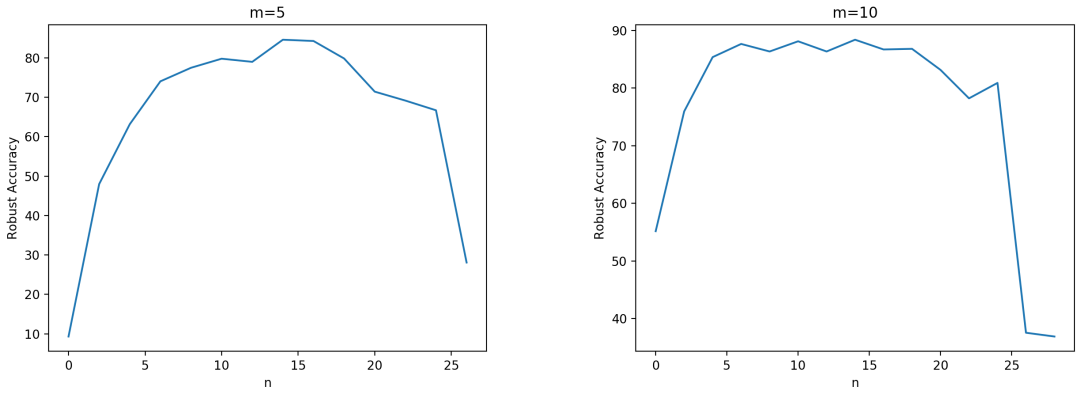

Figure 1: Robust Accuracy after training with YOPO-m-n after 10 epochs. Figure on the left shows how accuracy changes with $m = 5$ fixed and varying n, figure on the right is the same for $m = 10$ and varying n. In both figures we see that performance degrades quickly in n after a certain point, as predicted by Theorem 5.

obtain lower robust accuracy, even though the adversary is given more updates to find a worst case perturbation. We demonstrate this phenomenon with a robust classification experiment on the MNIST dataset, as shown in Figure 4.

This observation is reminiscent of results in the literature on the Method of Successive Approximations (MSA) for finding controls and trajectories which satisfy the PMP. These methods alternate between computing state and costate trajectories and maximize the Hamiltonian to update the control. We can interpret the adversary's updates as a MSA variant for the adversary's Hamiltonian minimization condition. It has been shown that if the new controls result in trajectories that deviate too far from the trajectories used in the Hamiltonian (in our case resulting in a larger oracle error) these methods will not converge Chernousko and Lyubushin (1982). This is consistent with the interpretation of our result.

5. Conclusion

We give the first convergence analysis for a recently proposed robust training algorithm for deep neural networks. By using methods from optimal control theory and results from inexact oracle methods in optimization, we shed light on the behavior of the algorithm as a function of its hyperparamters. It is likely that the interpretation of PMP-based algorithms as inexact oracle methods can be used to prove convergence for other learning algorithms inspired by optimal control, such as the MSA variants proposed in Li and Hao (2018); this is left for future work. Another avenue to explore is the behavior of approximate adversary updates in the overparameterized regime, as inspired by recent convergence results of overparameterized adversarial training with vanilla PGD Gao et al. (2019).

References

Tamer Başar and Pierre Bernhard. *H-infinity optimal control and related minimax design problems: a dynamic game approach*. Springer Science & Business Media, 2008.

Stephen Boyd and Lieven Vandenberghe. *Convex optimization*. Cambridge university press, 2004.

Nicholas Carlini and David Wagner. Towards evaluating the robustness of neural networks. In *2017 IEEE Symposium on Security and Privacy (SP)*, pages 39–57. IEEE, 2017.

FL Chernousko and AA Lyubushin. Method of successive approximations for solution of optimal control problems. *Optimal Control Applications and Methods*, 3(2):101–114, 1982.

Ruiqi Gao, Tianle Cai, Haochuan Li, Cho-Jui Hsieh, Liwei Wang, and Jason D Lee. Convergence of adversarial training in overparametrized neural networks. In *Advances in Neural Information Processing Systems*, pages 13009–13020, 2019.

Ian Goodfellow, Jonathon Shlens, and Christian Szegedy. Explaining and harnessing adversarial examples. In *International Conference on Learning Representations*, 2015.

Yann LeCun, D Touresky, G Hinton, and T Sejnowski. A theoretical framework for back-propagation. In *Proceedings of the 1988 connectionist models summer school*, volume 1, pages 21–28. CMU, Pittsburgh, Pa: Morgan Kaufmann, 1988.

Yann LeCun, Yoshua Bengio, and Geoffrey Hinton. Deep learning. *nature*, 521(7553):436–444, 2015.

Jaeho Lee and Maxim Raginsky. Minimax statistical learning with wasserstein distances. In *Advances in Neural Information Processing Systems*, pages 2687–2696, 2018.

Qianxiao Li and Shuji Hao. An optimal control approach to deep learning and applications to discrete-weight neural networks. In *International Conference on Machine Learning*, pages 2991–3000, 2018.

Qianxiao Li, Long Chen, Cheng Tai, and E Weinan. Maximum principle based algorithms for deep learning. *The Journal of Machine Learning Research*, 18(1):5998–6026, 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.

Jacob H Seidman, Mahyar Fazlyab, Victor M. Preciado, and George J. Pappas. Robust deep learning as optimal control: Insights and convergence guarantees, 2019. URL https://www.seas.upenn.edu/~mahyarfa/files/L4DC_SFPP.pdf.

Aman Sinha, Hongseok Namkoong, and John Duchi. Certifiable distributional robustness with principled adversarial training. In *International Conference on Learning Representations*, 2018.

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.

Yisen Wang, Xingjun Ma, James Bailey, Jinfeng Yi, Bowen Zhou, and Quanquan Gu. On the convergence and robustness of adversarial training. In *International Conference on Machine Learning*, pages 6586–6595, 2019.

E Weinan. A proposal on machine learning via dynamical systems. *Communications in Mathematics and Statistics*, 5(1):1–11, 2017.

E Weinan, Jiequn Han, and Qianxiao Li. A mean-field optimal control formulation of deep learning. *Research in the Mathematical Sciences*, 6(1):10, 2019.

Dinghuai Zhang, Tianyuan Zhang, Yiping Lu, Zhanxing Zhu, and Bin Dong. You only propagate once: Accelerating adversarial training via maximal principle. *arXiv preprint arXiv:1905.00877*, 2019.