You Only Propagate Once: Accelerating Adversarial Training via Maximal Principle

Dinghuai Zhang*, Tianyuan Zhang*, Yiping Lu*, Zhanxing Zhu, Bin Dong
{zhangdinghuai, 1600012888} @ pku.edu.cn, yplu@stanford.edu, zhanxing.zhu@pku.edu.cn, dongbin@math.pku.edu.cn
Peking University & Stanford University & Beijing Institute of Big Data Research

Dinghuai Zhang and Tianyuan Zhang are looking for PhD positions this year, please feel free to contact them if you find YOPO interesting!

Introduction

- Adversarial training suffers from extremely large computation costs.
- To solve this problem, we take an optimal control view to fully utilize network's architecture.
- We reformulate adversarial training as a differential game, and derive You Only Propagate Once (YOPO) algorithm based on Pontryagin's Maximal Principle (PMP) where the adversary control is only coupled with the first layer.
- Gradient based YOPO achieve $4 \sim 5$ times acceleration.
- Combining YOPO with TRADES[1], we achieve both higher clean and robust accuracy within less than half of the time TRADES need.

Background

Adversarial Examples Changing input with a perturbation in a human-imperceptible way can cause the neural network to output an incorrect prediction. These well-designed perturbed input samples are called adversarial examples.

Projected Gradient Descent (PGD) Attack PGD attack is one of the strongest attacks (approaches to generate adversarial examples). We denote PGD-r attack as doing

$$x^{t+1} = \Pi_{x+\mathcal{S}} \left(x^t + \alpha \operatorname{sign} \left(\nabla_x \ell(\theta, x, y) \right) \right)$$

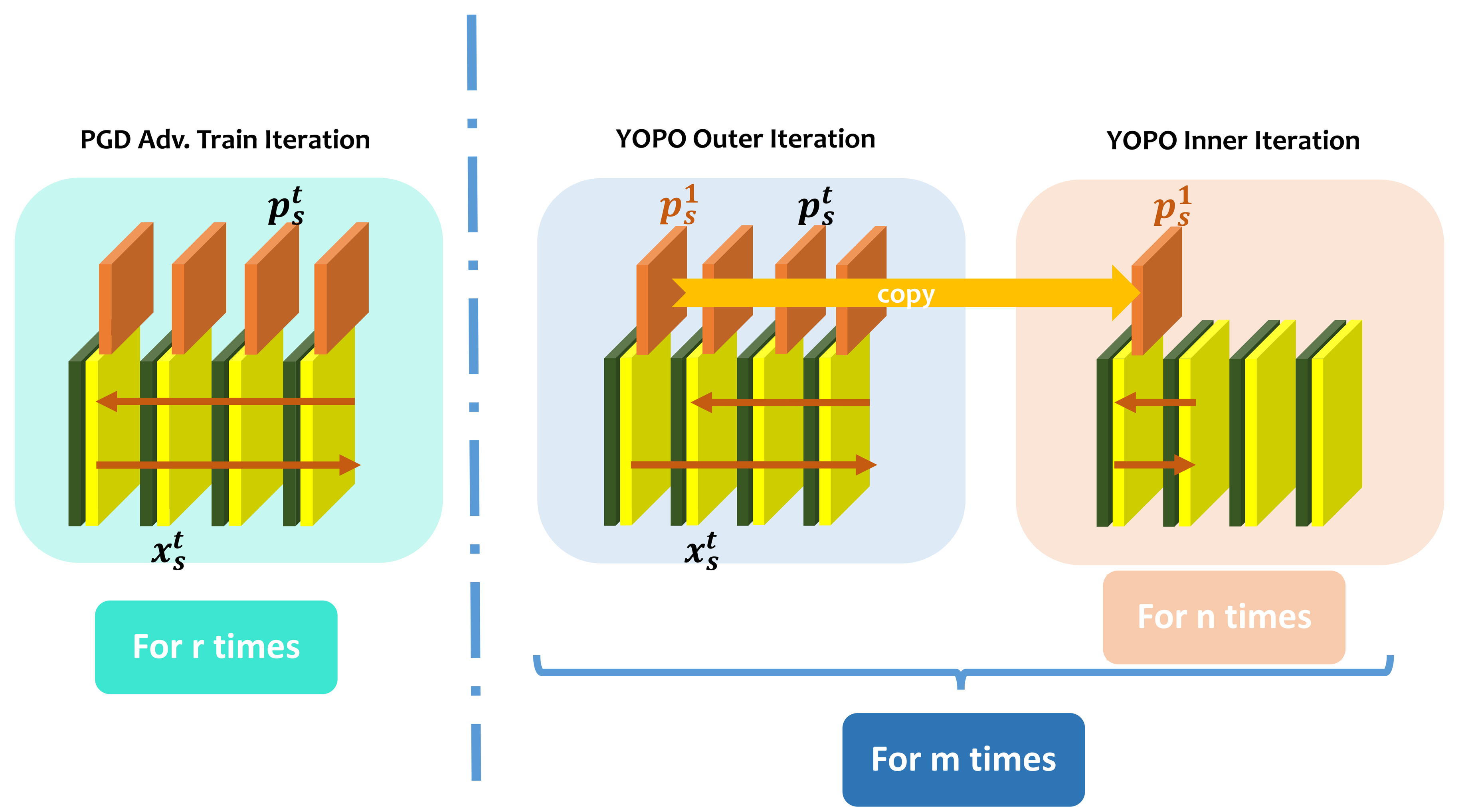
for r times. $\Pi_{x+\mathcal{S}}$ denotes projection to some neighbourhood of x.

PGD Adversarial training: *one of the most successful approach for building robust models so far for defending adversarial examples*

$$\min_\theta \mathbb{E}_{(x,y)\sim\mathcal{D}} \left[\max_{\delta \in \mathcal{S}} \ell(f(x+\delta;\theta), y) \right], \qquad (1)$$

where the inner maximization optimization is solved by PGD attacks.

Pipeline of YOPO (see Algorithm 1 below)



Pipeline of YOPO-m-n described in Algorithm 1. The yellow and olive blocks represent feature maps while the orange blocks represent the gradients of the loss w.r.t. feature maps of each layer. t denotes the index of layer.

A Simple Understanding: Splitting Adv. Training

We recast the problem 1 as

$$\min_\theta \max_\eta \sum_{i=1}^{S} L(g_{\theta^*}(f_0(x_i+\eta_i,\theta_0)), y_i) \qquad (3)$$

where $g_{\theta^*} = f_{T-1}^{\theta_{T_1}} \circ f_{T-2}^{\theta_{T-2}} \circ \cdots f_1^{\theta_1}$ denotes the network function without the first layer. Here θ^* is defined as $[\theta_1, \cdots, \theta_{T-1}]$, where θ_t is the parameter of t-th layer and T is the number of layers.

Algorithm 1 pseudocode for YOPO-m-n

```
initialize perturbation η
for k = 1 to m do
    p ← ∇_{f_0} ℓ(x + η)
    for i = 1 to n do
        η ← η + α · p · ∇_x f_0(x + η)              –splitting
    end for
    accumulate gradient U ← U + ∇_θ ℓ(x + η)
                    –use intermediate adversarial examples
end for
Use U tp perform SGD / momentum SGD =0
```

Differential Game

The robust optimization problem (1) can be written as a differential game as follows,

$$\min_\theta \max_{\|\eta_i\|\le\epsilon} J(\theta,\eta) = \frac{1}{N}\sum_{i=1}^{N} \ell_i(x_{i,T}) + \frac{1}{N}\sum_{i=1}^{N}\sum_{t=0}^{T-1} R_t(x_{i,t};\theta_t)$$

$$\text{subject to } x_{i,1} = f_0(x_{i,0}+\eta_i,\theta_0), i = 1,2,\cdots,N$$
$$x_{i,t+1} = f_t(x_{i,t},\theta_t), t = 1,2,\cdots,T-1 \qquad (2)$$

Here, the dynamics $\{f_t(x_t,\theta_t), t = 0,1,\ldots,T-1\}$ represent a deep neural network, T denotes the number of layers, $\theta_t \in \Theta_t$ denotes the parameters in layer t ($\theta = \{\theta_t\}_t$), the function $f_t : \mathbb{R}^{d_t} \times \Theta_t \to \mathbb{R}^{d_{t+1}}$ is a nonlinear transformation for one layer of neural network where d_t is the dimension of the t-th feature map and $\{x_{i,0}, i = 1,\ldots,N\}$ is the training dataset. The variable $\eta = (\eta_1,\cdots,\eta_N)$ is the adversarial perturbation and we constrain it in an ∞-ball. Function ℓ_i is a data fitting loss function and R_t is the regularization weights θ_t such as the L_2-norm. By casting the problem of adversarial training as a differential game (2), we regard θ and η as two competing players, each trying to minimize/maximize the loss function $J(\theta,\eta)$ respectively.

Pontryagin's Maximum Principle for Adversarial Training

Pontryagin type of maximal principle [4] provides necessary conditions for optimality with a layer-wise maximization requirement on the Hamiltonian function. For each layer $t \in [T] := \{0,1,\ldots,T-1\}$, we define the Hamiltonian function $H_t : \mathbb{R}^{d_t} \times \mathbb{R}^{d_{t+1}} \times \Theta_t \to \mathbb{R}$ as

$$H_t(x,p,\theta_t) = p \cdot f_t(x,\theta_t) - \frac{1}{B} R_t(x,\theta_t).$$

where B denotes batch size. Here, we present the PMP for our discrete time differential game (2).

Theorem 1. *(PMP for adversarial training) Assume ℓ_i is twice continuous differentiable, $f_t(\cdot,\theta), R_t(\cdot,\theta)$ are twice continuously differentiable with respect to x; $f_t(\cdot,\theta), R_t(\cdot,\theta)$ together with their x partial derivatives are uniformly bounded in t and θ, and the sets $\{f_t(x,\theta) : \theta \in \Theta_t\}$ and $\{R_t(x,\theta) : \theta \in \Theta_t\}$ are convex for every t and $x \in \mathbb{R}^{d_t}$. Denote θ^* as the solution of the problem (2), then there exists co-state processes $p_i^* := \{p_{i,t}^* : t \in [T]\}$ such that the following holds for all $t \in [T]$ and $i \in [B]$:*

$$x_{i,t+1}^* = \nabla_p H_t(x_{i,t}^*, p_{i,t+1}^*, \theta_t^*), \qquad x_{i,0}^* = x_{i,0} + \eta_i^* \qquad (4)$$

$$p_{i,t}^* = \nabla_x H_t(x_{i,t}^*, p_{i,t+1}^*, \theta_t^*), \qquad p_{i,T}^* = -\frac{1}{B}\nabla \ell_i(x_{i,T}^*) \qquad (5)$$

At the same time, the parameters of the first layer $\theta_0^* \in \Theta_0$ and the optimal adversarial perturbation η_i^* satisfy

$$\sum_{i=1}^{B} H_0(x_{i,0}^* + \eta_i, p_{i,1}^*, \theta_0^*) \ge \sum_{i=1}^{B} H_0(x_{i,0}^* + \eta_i^*, p_{i,1}^*, \theta_0^*) \ge \sum_{i=1}^{B} H_0(x_{i,0}^* + \eta_i^*, p_{i,1}^*, \theta_0), \qquad (6)$$

$$\forall \theta_0 \in \Theta_0, \|\eta_i\|_\infty \le \epsilon \qquad (7)$$

and the parameters of the other layers $\theta_t^* \in \Theta_t, t \in [T]$ maximize the Hamiltonian functions

$$\sum_{i=1}^{B} H_t(x_{i,t}^*, p_{i,t+1}^*, \theta_t^*) \ge \sum_{i=1}^{B} H_t(x_{i,t}^*, p_{i,t+1}^*, \theta_t), \ \forall \theta_t \in \Theta_t \qquad (8)$$

We utilize this PMP for adversarial problem to design a general YOPO algorithm. Gradient based optimized YOPO can be proved to be equivalent to Algorithm 1. Details can be seen in our paper.

Experiments Results (blue denotes comparable performance)

Training Methods	Clean Data	PGD-20 Attack	Training Time (mins)
Natural train	95.03%	0.00%	233
PGD-3 [2]	90.07%	39.18%	1134
PGD-5 [2]	89.65%	43.85%	1574
PGD-10 [2]	87.30%	47.04%	2713
Free-8 [3][1]	86.29%	47.00%	667
YOPO-3-5 (Ours)	87.27%	43.04%	299
YOPO-5-3 (Ours)	86.70%	47.98%	**476**

[1] Code from https://github.com/ashafahi/free_adv_train.

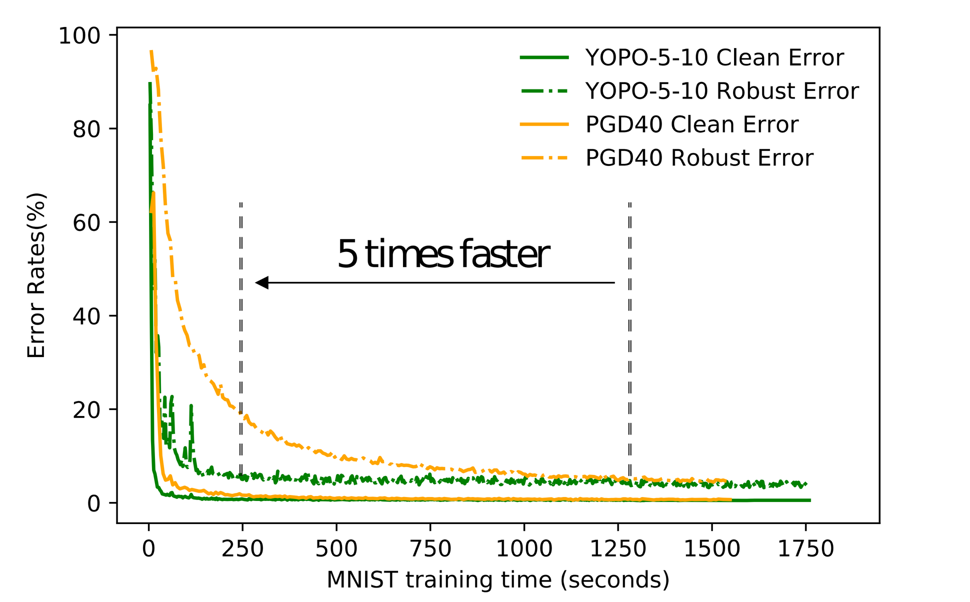
Table 1: Results of Wide ResNet34 for CIFAR10.

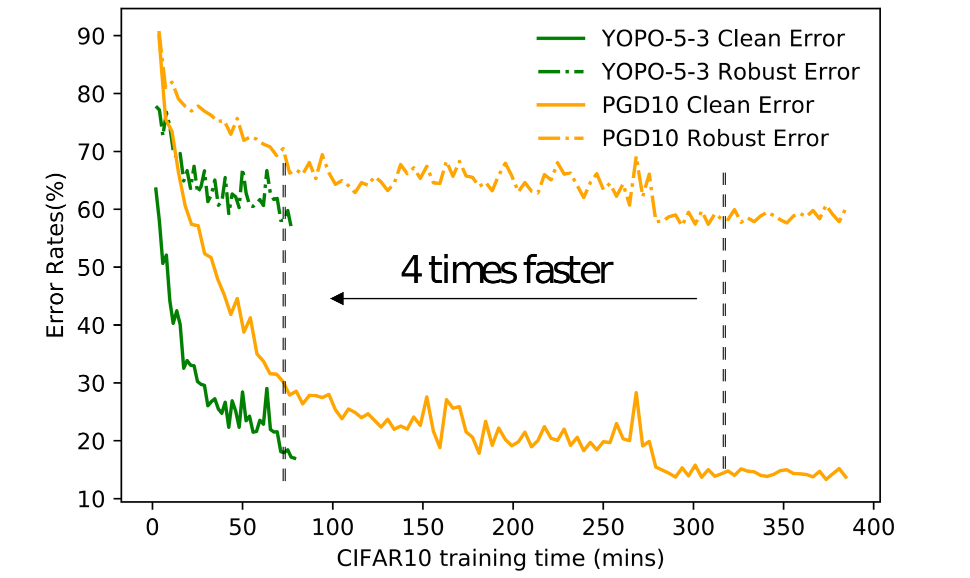
We also combine YOPO with TRADES's [1] minimax objective to achieve the state-of-the-art defense results:

Training Methods	Clean Data	PGD-20 Attack	CW Attack	Training Time (mins)
TRADES-10 [1]	86.14%	44.50%	58.40%	633
TRADES-YOPO-3-4 (Ours)	87.82%	46.13%	59.48%	259
TRADES-YOPO-2-5 (Ours)	88.15%	42.48%	59.25%	218

Table 2: Results of training PreAct-Res18 for CIFAR10 with TRADES objective

Accelerating Effects





Experiments about performance w.r.t. training time on mnist and cifar10.

Remarks

Specifically, our algorithms accelerates the training from two aspects:

- Split the computation of adversarial examples
- Re-use the "half-way" intermediate adversarial examples

References

[1] Hongyang Zhang, et al. (2019) Theoretically Principled Trade-off between Robustness and Accuracy. *ICML*.
[2] Aleksander Madry, et al. (2018) Towards deep learning models resistant to adversarial attacks. *ICLR*.
[3] Ali Shafahi, et al. (2019) Adversarial training for free! *NeurIPS*
[4] Boltyanskii Vladimir Grigor'evich, et al. (1960) The theory of optimal processes. I. The maximum principle