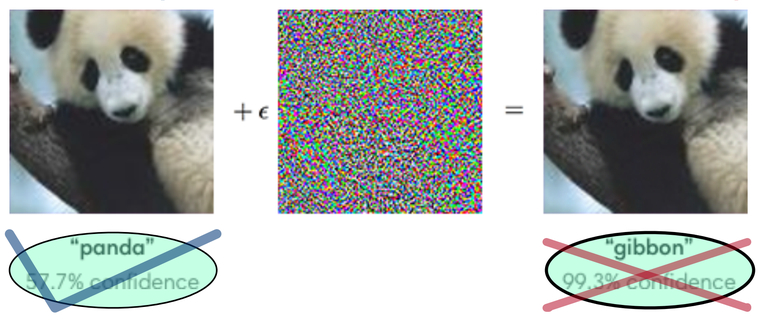
Learning to Defense by Learning to Attack

Haoming Jiang[†], Zhehui Chen[†], Yuyang Shi[†], Bo Dai[◇], Tuo Zhao[†]
[†]Georgia Tech [◇]Google Brain

Background

Deep Neural Networks are **powerful** and **successful** in many applications, e.g., image classification. However, they are also **vulnerable** to adversarial attacks, i.e., slightly perturbed images can easily **fool** neural networks.



Question: **Robust** Classification?

Adversarial Training

Adversarial Training provides a principled approach to improve robustness. minimize the empirical risk of training samples with certain injected maximal perturbations.

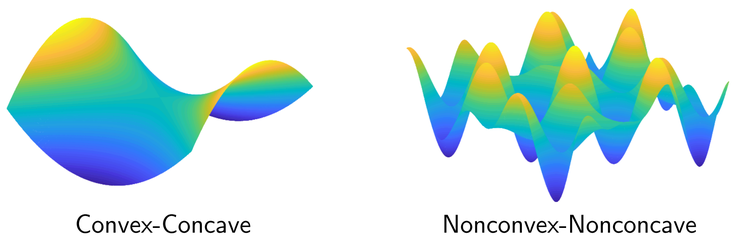
Formulation: Given n samples $\{(x_i, y_i)\}_{i=1}^n$, we solve the following **minimax** problem:

$$\min_\theta \frac{1}{n}\sum_{i=1}^n \left[\max_{\delta_i\in\mathcal{B}(\epsilon)} \ell(f(x_i+\delta_i,\theta),y_i)\right], \quad (1)$$

- $x_i,\ y_i,\ \delta_i$: i-th sample, label, and perturbation;
- $\mathcal{B}(\epsilon)$: Perturbation constraints (e.g., $||\delta_i||_\infty \le \epsilon$);
- ℓ : Surrogate loss function (e.g., Cross-entropy);
- $f(\cdot;\theta)$: Neural network classifier with parameter θ.

Two-player Game: Attacker **generates** adversarial samples to fool classifier; while classifier aims to **correctly classify** adversarial samples generated by attacker.

Solving problem (1) is very challenging:
- Landscape is highly **nonconvex-nonconcave**, and finding global equilibria is very difficult.
- We need to **search over huge space** for generating adversarial perturbations.



Fundamental Hardness

Adversarial Training is essentially solving

$$\min_\theta \frac{1}{n}\sum_{i=1}^n h(x_i,\theta,y_i),$$

where $h(x_i,\theta,y_i) = \max_{\delta_i\in\mathcal{B}(\epsilon)} \ell(f(x_i+\delta_i,\theta),y_i)$.

Ideal: At the $(t+1)$-th iteration,

- Inner Maximization (Attacker):

$$\widetilde{\delta}_i^{(t+1)} = \operatorname{argmax}_{\delta_i\in\mathcal{B}(\epsilon)} \ell(f(x_i+\delta_i,\theta^{(t)}),y_i);$$

- Outer Minimization (Classifier):

$$\theta^{(t+1)} = \theta^{(t)} - \eta\nabla_\theta \ell(f(x_i+\widetilde{\delta}_i^{(t+1)},\theta^{(t)}),y_i)$$
$$= \theta^{(t)} - \eta\nabla_\theta h(x_i,\theta^{(t)},y_i).$$

Difficulty: Since the inner maximization problem is highly nonconcave, we obtain $\delta_i^{(t+1)} \neq \widetilde{\delta}_i^{(t+1)}$. Therefore,

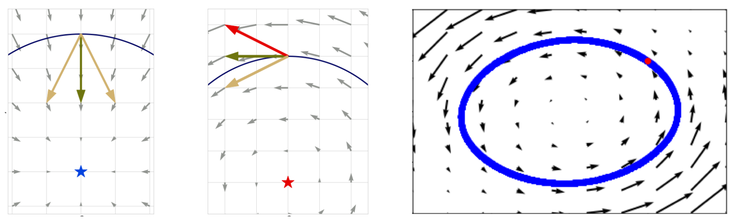
$$\frac{\partial \ell(f(x_i+\delta_i^{(t+1)},\theta),y_i)}{\partial\theta} \neq \frac{\partial h(x_i,\theta^{(t)},y_i)}{\partial\theta}.$$

Handcrafted Algorithms

Reality: We solve the inner maximization problem by some optimization-based algorithms:

- Fast Gradient Sign Method (FGSM);
- Projected Gradient Method (PGM);
- Carlini-Wagner Attack (CW).

Existing algorithms often fail to converge.

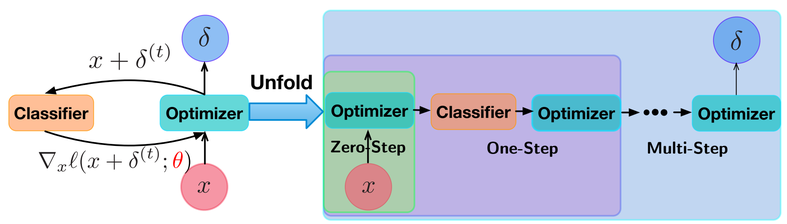


Let us Go beyond Handcrafted Algorithms!

Learning to Learn/Optimize (L2L)

High Level Idea:
- Cast the optimizer as a **learning** model;
- Allow the model to learn to **exploit** the perturbation structure automatically.

Implementation: Parameterize the **optimizer** as a neural network, and learn its parameters.



L2L in Adversarial Training:

$$\min_\theta \max_\phi \ell(f(x_i+g(\mathcal{A}(x_i,y_i,\theta);\phi);\theta),y_i), \quad (2)$$

subject to $g(\mathcal{A}(x_i,y_i,\theta);\phi) \in \mathcal{B}(\epsilon)\ \ \forall i\in\{1,\cdots,n\}$.

- $g(\cdot;\phi)$: Neural network optimizer with parameter ϕ;
- $\mathcal{A}(x,y,\theta)$: Input of g (decided by algorithm)

Problem (2) is closely related to generative adversarial networks (GAN). Both contain two networks: generator and classifier, that are jointly trained.

Simple L2L.

Simply use clean feature with $\mathcal{A}(x,y,\theta) = x$.

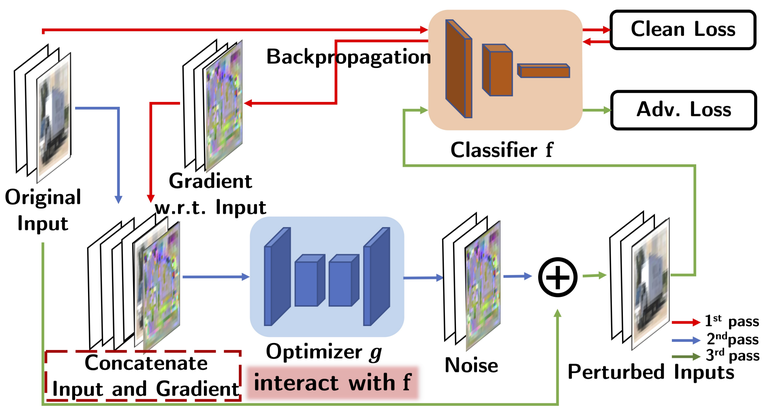
The perturbation $\delta_i = g(x_i;\phi)$. Under this setting, L2L training is similar to GAN training. However, optimizer g lacks the interaction with classifier f.

Grad L2L.

Motivated by the gradient ascent method with

$$\mathcal{A}(x,y,\theta) = [x, \nabla_x \ell(f(x,\theta),y)].$$

The perturbation $\delta_i = g(x_i, \nabla_x\ell(f(x_i,\theta),y_i);\phi)$. Under this setting, optimizer g interacts with classifier f via $\nabla_x\ell$ and exploits the gradient information, which essentially represents an optimization algorithm.



Grad L2L Architecture

Learning to Learn/Optimize (Cont'd)

Multi-Step Grad.

Use Grad L2L as the cell of recurrent neural network (RNN) to mimic PGM.

To avoid the significant computational burden in RNN's backpropagation, we mainly focus on 2-Step Grad, in which the corresponding perturbation becomes:

$$\delta_i = \Pi_{\mathcal{B}(\epsilon)}\Big(\delta_i^{(0)} + g\big(x_i+\delta_i^{(0)}, \nabla_x\ell(f(x_i+\delta_i^{(0)};\theta),y_i);\phi\big)\Big),$$

where $\delta_i^{(0)} = g(x_i, \nabla_x\ell(f(x_i,\theta),y_i);\phi)$ and $\Pi_{\mathcal{B}(\epsilon)}$ denotes the projection to $\mathcal{B}(\epsilon)$.

Advantages of L2L:
- ◇ Attacker can yield strong perturbations;
- ◇ Learn common structure across all perturbations;
- ◇ Ease the training process via overparametrization;
- ◇ Reduce search space and enjoy computational efficiency.

Experiments

Classifier: A 32-layer Wide Residual Network.
Tolerance: $\epsilon = 8$ pixels (RGB scale).

Attacker Architectures

Conv:	$[k=3\times3, c=128, s=1, p=1]$, BN+ReLU
ResBlocks:	[channel = 256]
ResBlocks:	[channel = 128], BN
DeConv:	$[k=4\times4, c=16, s=2, p=1]$, BN+ReLU
Conv:	$[k=3\times3, c=3, s=1, p=1]$, tanh

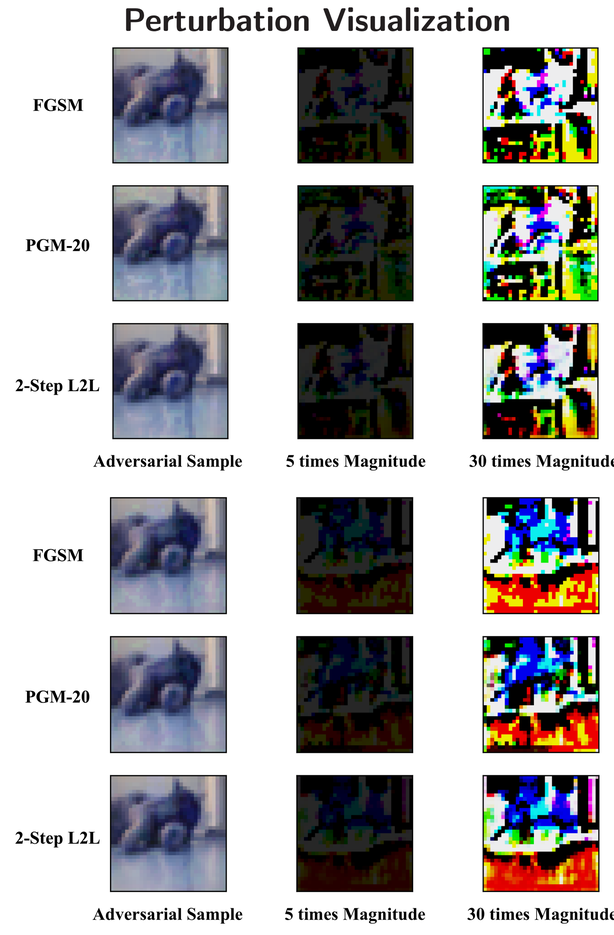
Running Time per Epoch on CIFAR-10

Plain Net		PGM Net
106.5 ± 1.5 s		1310.8± 14.2 s
Naive L2L	Grad L2L	2-Step L2L
293.7 ±3.1 s	617.5± 6.1s	805.1 ± 8.1 s

Accuracy (Worst over 5 Runs)

Data Set	CIFAR-10			CIFAR-100		
Evaluation	Clean	PGM	CW	Clean	PGM	CW
Plain	95.22	0.00	0.00	76.10	0.13	0.00
PGM	87.30	47.04	–	62.68	23.75	25.95
Naive L2L	94.53	0.01	0	–	–	–
Grad L2L	85.84	51.17	53.5	62.18	28.67	29.65
2-Step L2L	85.35	**54.32**	**57.07**	60.95	**31.03**	**32.28**

Perturbation Visualization



Top: PGM-20 Net; Bottom: 2-Step L2L Net.

Reference

[1] Explaining and Harnessing Adversarial Examples, 2014.
[2] Learning to learn by gradient descent by gradient descent, 2016.
[3] Towards deep learning models resistant to adversarial attacks, 2017.