Towards Principled Methodologies and Efficient Algorithms for Minimax Machine Learning

Tuo Zhao

Georgia Tech, Jun. 26. 2019

Joint work with Haoming Jiang, Minshuo Chen (Georgia Tech), Bo Dai (Google Brain), Zhaoran Wang (Northwestern U) and others.

Background

Minimax Machine Learning

Conventional Empirical Risk Minimization: Given training data $z_1, ..., z_n$, we minimize an empirical risk function,

$$\min_{\theta} \frac{1}{n} \sum_{i=1}^{n} f(z_i; \theta).$$

Minimax Formulation: We solve a minimax problem,

$$\min_{\theta} \max_{\phi} \frac{1}{n} \sum_{i=1}^{n} f(z_i; \theta, \phi).$$

More **Flexible**.

Minimax Machine Learning

Conventional Empirical Risk Minimization: Given training data $z_1, ..., z_n$, we minimize an empirical risk function,

$$\min_{\theta} \frac{1}{n} \sum_{i=1}^{n} f(z_i; \theta).$$

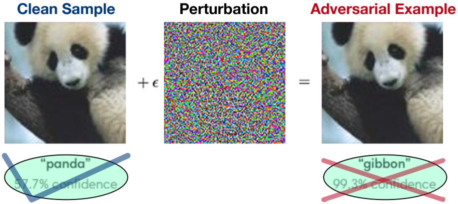
Minimax Formulation: We solve a minimax problem,

$$\min_{\theta} \max_{\phi} \frac{1}{n} \sum_{i=1}^{n} f(z_i; \theta, \phi).$$

More **Flexible**.

Motivating Application: Robust Deep Learning

Neural Networks are vulnerable to adversarial examples (Goodfellow et al. 2014, Madry et al. 2017).

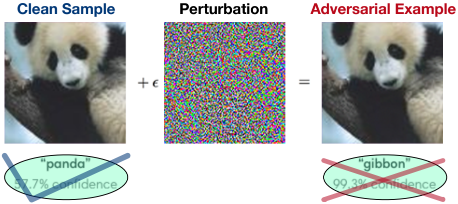


- Adversarial Perturbation: $\max_{\delta_i \in \mathcal{B}} \ell(f(x_i + \delta_i; \theta), y_i)$,
- Adversarial Training: $\min_{\theta} \frac{1}{n} \sum_{i=1}^{n} \max_{\delta_i \in \mathcal{B}} \ell(f(x_i + \delta_i; \theta), y_i)$,

where $\delta_i \in \mathcal{B}$ denotes the imperceptible perturbation.

Motivating Application: Robust Deep Learning

Neural Networks are vulnerable to adversarial examples (Goodfellow et al. 2014, Madry et al. 2017).



- Adversarial Perturbation: $\max_{\delta_i \in \mathcal{B}} \ell(f(x_i + \delta_i; \theta), y_i)$,
- Adversarial Training: $\min_{\theta} \frac{1}{n} \sum_{i=1}^{n} \max_{\delta_i \in \mathcal{B}} \ell(f(x_i + \delta_i; \theta), y_i)$,

  where $\delta_i \in \mathcal{B}$ denotes the imperceptible perturbation.

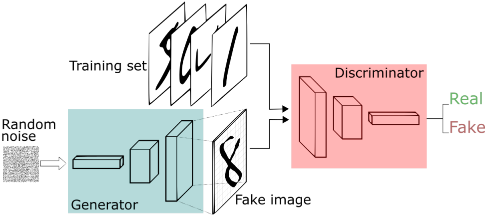
Motivating Application: Image Generation

Brock et al. (2019)

All are fake!

Motivating Application: Unsupervised Learning

Generative Adversarial Network: Goodfellow et al. (2014), Arjovsky et al. (2017), Miyato et al. (2018), Brock et al. (2019)

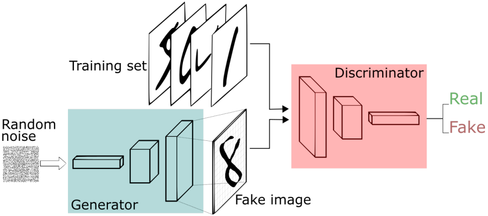


$$\min_{\theta}\max_{\mathcal{W}}\frac{1}{n}\sum_{i=1}^{n}\phi\left(\mathcal{A}(D_{\mathcal{W}}(x_i))\right)+\mathbb{E}_{x\sim\mathcal{D}_{G_\theta}}[\phi\left(1-\mathcal{A}(D_{\mathcal{W}}(x))\right)].$$

$D_{\mathcal{W}}$: Discriminator; G_θ: Generator; ϕ: $\log(\cdot)$ and $\mathcal{A}$: Softmax.

Motivating Application: Unsupervised Learning

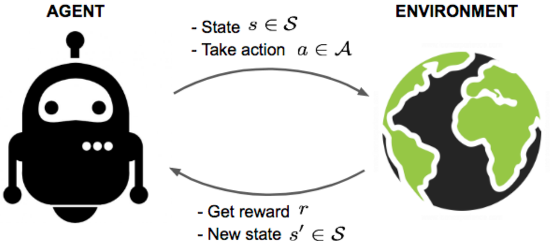
Generative Adversarial Network: Goodfellow et al. (2014), Arjovsky et al. (2017), Miyato et al. (2018), Brock et al. (2019)



$$\min_{\theta} \max_{\mathcal{W}} \frac{1}{n} \sum_{i=1}^{n} \phi\left(\mathcal{A}(D_{\mathcal{W}}(x_i))\right) + \mathbb{E}_{x \sim \mathcal{D}_{G_\theta}}\left[\phi\left(1 - \mathcal{A}(D_{\mathcal{W}}(x))\right)\right].$$

$D_{\mathcal{W}}$: Discriminator; G_θ: Generator; ϕ: $\log(\dot{})$ and $\mathcal{A}$: Softmax.

Motivating Application: Reinforcement Learning



Motivating Application: Reinforcement Learning

Minimax Formulation: Given $\mathcal{M} = (\mathcal{A}, \mathcal{A}, P, R, \gamma)$, we solve

$$\min_{\pi, V} \max_{\nu} L(\pi, V; \nu) = 2\mathbb{E}_{s,a,s'}[\nu(s,a)(R(s,a) + \gamma V(s') - \lambda \log(\pi(a|s))] - E_{s,a,s'}\nu^2(s,a),$$

where s denotes the state, a denotes the action, and

- Policy: $\pi: \ \mathcal{S} \to \mathcal{P}(\mathcal{A})$,
- Value: $V: \ \mathcal{S} \to R$,
- Reward: $R: \ \mathcal{S} \times \mathcal{A} \to R$,
- Axillary Dual: $\nu: \ \mathcal{S} \times \mathcal{A} \to R$.

The policy π is parameterized as a neural network, where as ν is parameterized as a reproducing kernel function (Dai et al. 2018).

Successes of Minimax Machine Learning

- Adversarial Robust Learning
- Unsupervised Learning
- Learning with Constraints
- Reinforcement Learning
- Domain Adaptation
- Generative Adversarial Imitation Learning
- ...

$\Longrightarrow$ Identify the fundamental **hardness** of minimax machine learning and make **optimization** easier.

Challenges

Minimax Optimization

General Formula:

$$\min_{x \in \mathcal{X}} \max_{y \in \mathcal{Y}} f(x, y),$$

$\mathcal{X} \subset \mathbb{R}^d$, $\mathcal{Y} \subset \mathbb{R}^p$, f is some continuous function.

Two Stage Optimization:

- Stage 1: $g(x) = \max_{y \in \mathcal{Y}} f(x, y)$,
- Stage 2: $\min_{x \in \mathcal{X}} g(x)$,
- Solve Stage 2 using gradient descent.

Limitation: A global maximum of $\max_{y \in \mathcal{Y}} f(x, y)$ needs to be obtained for evaluating $\nabla g(x)$ (Envelope Theorem, Afriat et al. (1971)).

Minimax Optimization

General Formula:

$$\min_{x \in \mathcal{X}} \max_{y \in \mathcal{Y}} f(x, y),$$

$\mathcal{X} \subset \mathbb{R}^d$, $\mathcal{Y} \subset \mathbb{R}^p$, f is some continuous function.

Two Stage Optimization:

- Stage 1: $g(x) = \max_{y \in \mathcal{Y}} f(x, y)$,
- Stage 2: $\min_{x \in \mathcal{X}} g(x)$,
- Solve Stage 2 using gradient descent.

Limitation: A global maximum of $\max_{y \in \mathcal{Y}} f(x, y)$ needs to be obtained for evaluating $\nabla g(x)$ (Envelope Theorem, Afriat et al. (1971)).

Minimax Optimization

General Formula:

$$\min_{x\in\mathcal{X}}\max_{y\in\mathcal{Y}} f(x,y),$$

$\mathcal{X}\subset\mathbb{R}^d$, $\mathcal{Y}\subset\mathbb{R}^p$, f is some continuous function.

Two Stage Optimization:

- Stage 1: $g(x)=\max_{y\in\mathcal{Y}} f(x,y)$,
- Stage 2: $\min_{x\in\mathcal{X}} g(x)$,
- Solve Stage 2 using gradient descent.

Limitation: A global maximum of $\max_{y\in\mathcal{Y}} f(x,y)$ needs to be obtained for evaluating $\nabla g(x)$ (Envelope Theorem, Afriat et al. (1971)).

Existing Literature

Bilinear Saddle Point Problem:

$$\min_{x\in\mathcal{X}}\left\{p(x)+\max_{y\in\mathcal{Y}}\langle Ax,y\rangle-q(y)\right\}.$$

$\mathcal{X}\subset\mathbb{R}^d$ and $\mathcal{Y}\subset\mathbb{R}^p$: closed convex domain; $A\in\mathbb{R}^{p\times d}$;
$p(\cdot)$ and $q(\cdot)$: convex functions satisfying certain assumptions.

Nice Structure: **Convex** in x and **Concave** in y; **Bilinear** interaction (can be slightly relaxed).

Algorithms with Theoretical Guarantees:

Primal-Dual Algorihtm, Mirror-Prox Algorithm $\cdots$ (Nemirovski 2005, Chen et al. 2014, Dang et al. 2015).

Existing Literature

Bilinear Saddle Point Problem:

$$\min_{x\in\mathcal{X}}\left\{p(x)+\max_{y\in\mathcal{Y}}\langle Ax,y\rangle-q(y)\right\}.$$

$\mathcal{X}\subset\mathbb{R}^d$ and $\mathcal{Y}\subset\mathbb{R}^p$: closed convex domain; $A\in\mathbb{R}^{p\times d}$;
$p(\cdot)$ and $q(\cdot)$: convex functions satisfying certain assumptions.

Nice Structure: **Convex** in x and **Concave** in y; **Bilinear** interaction (can be slightly relaxed).

Algorithms with Theoretical Guarantees:

Primal-Dual Algorihtm, Mirror-Prox Algorithm $\cdots$ (Nemirovski 2005, Chen et al. 2014, Dang et al. 2015).

Existing Literature

Bilinear Saddle Point Problem:

$$\min_{x \in \mathcal{X}} \left\{ p(x) + \max_{y \in \mathcal{Y}} \langle Ax, y \rangle - q(y) \right\}.$$

$\mathcal{X} \subset \mathbb{R}^d$ and $\mathcal{Y} \subset \mathbb{R}^p$: closed convex domain; $A \in \mathbb{R}^{p \times d}$;
$p(\cdot)$ and $q(\cdot)$: convex functions satisfying certain assumptions.

Nice Structure: **Convex** in x and **Concave** in y; **Bilinear** interaction (can be slightly relaxed).

Algorithms with Theoretical Guarantees:

Primal-Dual Algorihtm, Mirror-Prox Algorithm $\cdots$ (Nemirovski 2005, Chen et al. 2014, Dang et al. 2015).

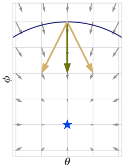
Challenges: Nonconcavity of Inner Maximization

Recall Stage 2: $\min_{x \in \mathcal{X}} \left\{ g(x) := \max_{y \in \mathcal{Y}} f(x, y) \right\}.$

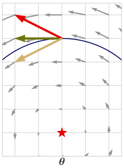
Why Fail to Converge?: $\widetilde{y} \neq \arg\max_y f(x, y)$ may even lead to

$$\left\langle \frac{\partial g(x)}{\partial x}, \frac{\partial f(x, \widetilde{y})}{\partial x} \right\rangle \ll 0.$$

Noisy Gradient

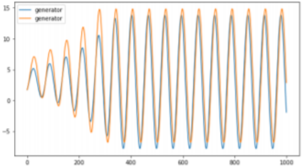


Minimization



Minmax

Limit Cycles



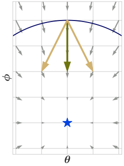
Challenges: Nonconcavity of Inner Maximization

Recall Stage 2: $\min_{x\in\mathcal{X}} \left\{ g(x) := \max_{y\in\mathcal{Y}} f(x,y) \right\}.$

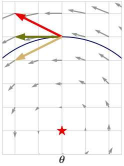
Why Fail to Converge?: $\widetilde{y} \neq \arg\max_y f(x,y)$ may even lead to

$$\left\langle \frac{\partial g(x)}{\partial x}, \frac{\partial f(x,\widetilde{y})}{\partial x} \right\rangle \ll 0.$$

Noisy Gradient

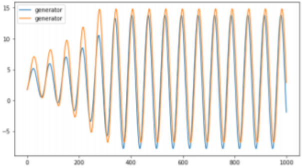


Minimization



Minmax

Limit Cycles



Our Proposed Solutions

State of the Art:

- Convex-concave: Well studied.
- Nonconvex-concave: Limitedly studied.
  Reinforcement Learning: Dai et al. (2018); Constrained OptimizationChen et al. (2019); $\cdots$
- Beyond: No algorithm works well.

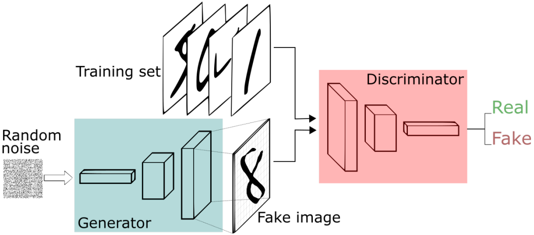
Our Solutions:

Improving Landscape and **Learning to Optimize**

Generative Adversarial Networks

Generative Adversarial Networks

Highly Nonconvex-Nonconcave Minimax Problem:

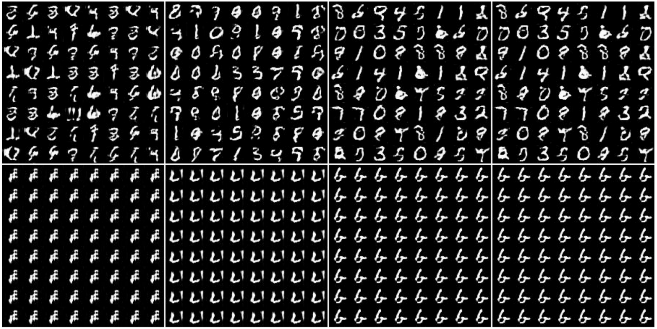


$$\min_{\theta} \max_{\mathcal{W}} \frac{1}{n} \sum_{i=1}^{n} \phi\left(\mathcal{A}(D_{\mathcal{W}}(x_i))\right) + \mathbb{E}_{x \sim \mathcal{D}_{G_\theta}} [\phi\left(1 - \mathcal{A}(D_{\mathcal{W}}(x))\right)].$$

$D_{\mathcal{W}}$: Discriminator; G_θ: Generator; $\phi, \mathcal{A}$: Properly chosen functions (e.g., $\log(\cdot)$ and Softmax).

Generative Adversarial Networks

Instability Issue: Mode Collapse



10k steps 20k steps 50K steps 100k steps

Stabilizing GAN Training

Better Algorithm:

- Two Time-Scale Update
- Functional Gradient
- Progressive Learning
- ...

Better Landscape:

- Gradient Penalty
- Weight Clipping
- Orthogonal Regularization
- Spectral Normalization
- ...

Algorithm works only if the **landscape** is good enough.

Stabilizing GAN Training

Better Algorithm:

- Two Time-Scale Update
- Functional Gradient
- Progressive Learning
- ...

Better Landscape:

- Gradient Penalty
- Weight Clipping
- Orthogonal Regularization
- Spectral Normalization
- ...

Algorithm works only if the **landscape** is good enough.

Better Optimization Landscape

Lipschitz Continuous Discriminator:

- An L-layer discriminator can be formulated as follows:

$$D_{\mathcal{W}}(x) = W_L \sigma_{L-1}(W_{L-1} \cdots \sigma_1(W_1 x) \cdots),$$

where W_i's are weight matrices and σ_i's are activations.

- 1-Lipschitz condition:

$$|D_{\mathcal{W}}(x) - D_{\mathcal{W}}(x')| \leq \|x - x'\|$$

Inspired by Wasserstein GAN (Arjovsky et al., 2017)

Empirically works well, but why?

Better Optimization Landscape

Lipschitz Continuous Discriminator:

- An L-layer discriminator can be formulated as follows:

$$D_{\mathcal{W}}(x) = W_L \sigma_{L-1}(W_{L-1} \cdots \sigma_1(W_1 x) \cdots),$$

where W_i's are weight matrices and σ_i's are activations.

- 1-Lipschitz condition:

$$|D_{\mathcal{W}}(x) - D_{\mathcal{W}}(x')| \leq \left\| x - x' \right\|$$

Inspired by Wasserstein GAN (Arjovsky et al., 2017)

Empirically works well, but why?

Control Weight Matrix Scaling

Scaling Issue: Consider a simple 2-layer discriminator with ReLU activation ($\sigma(\cdot) = \max(\cdot, 0)$):

$$D_{\mathcal{W}}(x) = W_2 \sigma(W_1 x).$$

Since the ReLU activation is homogeneous, we can rescale the weight matrices by a factor $\lambda > 0$ as

$$W_1 \Rightarrow \lambda \cdot W_1 \quad W_2 \Rightarrow W_2/\lambda.$$

Although the neural network model remains the same, the optimization landscape becomes worse.

Orthogonal Regularization:

$$\min_{W_1, W_2} \mathcal{L}(W_1, W_2) + \lambda \Big(\left\| W_1^\top W_1 - I \right\|_{\mathrm{F}}^2 + \left\| W_2^\top W_2 - I \right\|_{\mathrm{F}}^2 \Big).$$

Control Weight Matrix Scaling

Scaling Issue: Consider a simple 2-layer discriminator with ReLU activation ($\sigma(\cdot) = \max(\cdot, 0)$):

$$D_{\mathcal{W}}(x) = W_2\sigma(W_1 x).$$

Since the ReLU activation is homogeneous, we can rescale the weight matrices by a factor $\lambda > 0$ as

$$W_1 \Rightarrow \lambda \cdot W_1 \quad W_2 \Rightarrow W_2/\lambda.$$

Although the neural network model remains the same, the optimization landscape becomes worse.

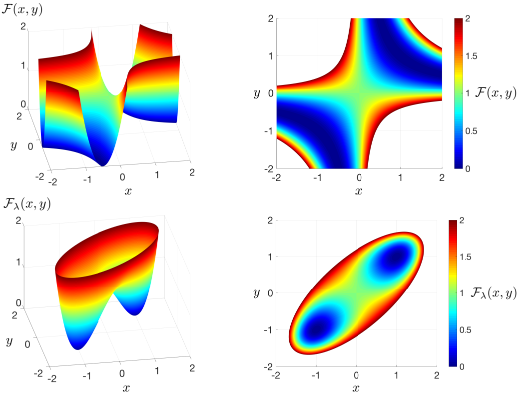
Orthogonal Regularization:

$$\min_{W_1, W_2} \mathcal{L}(W_1, W_2) + \lambda\Big(\left\| W_1^\top W_1 - I \right\|_{\mathrm{F}}^2 + \left\| W_2^\top W_2 - I \right\|_{\mathrm{F}}^2 \Big).$$

Illustrations of Landscape

$$\min_{x,y} \mathcal{F}(x,y) = (1 - xy)^2,$$

$$\min_{x,y} \mathcal{F}_\lambda(x,y) = (1 - xy)^2 + \lambda(x^2 - y^2)^2.$$



Also Improves Generalization

Theorem (Informal, Jiang et al. 2019)

Under some technical assumptions and assume

- $\|W_i\|_2 \leq B_{W_i}$ *for* $i \in [L]$ *and* $\|x_k\|_2 \leq B_x$ *for* $k \in [n]$.
- *Generator and discriminator are well trained, i.e.,*

$$d_{\mathcal{F},\phi}(\widehat{\mu}_n, \nu_n) - \inf_{\nu \in \mathcal{D}_G} d_{\mathcal{F},\phi}(\widehat{\mu}_n, \nu) \leq \epsilon,$$

where $d_{\mathcal{F},\phi}(\cdot,\cdot)$ *is the* **neural distance** *with probability at least* $1-\delta$, *we have*

$$d_{\mathcal{F},\phi}(\mu, \nu_n) - \inf_{\nu \in \mathcal{D}_G} d_{\mathcal{F},\phi}(\mu, \nu) \leq \widetilde{O}\left(\frac{B_x \prod_{i=1}^{L} B_{W_i} \sqrt{d^2 L}}{\sqrt{n}}\right).$$

From Lipschitz Continuity to Generalization

Importance of Spectrum Control:

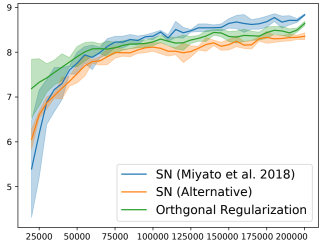
$$d_{\mathcal{F},\phi}(\mu,\nu_n) - \inf_{\nu\in\mathcal{D}_G} d_{\mathcal{F},\phi}(\mu,\nu) \le \widetilde{O}\left(\frac{B_x \prod_{i=1}^{L} B_{W_i}\sqrt{d^2L}}{\sqrt{n}}\right).$$

1-Lipschitz $\Longrightarrow$ polynomial bound $\widetilde{O}\left(\sqrt{\frac{d^2L}{n}}\right)$.

> Controling the product of spectral norms avoids bad landscape and benefits the generalization of GANs.

Better then Orthogonal Regularization

Spectral Normalization (SN, Miyato et al. 2018):

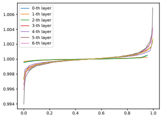


Inception Score on STL-10

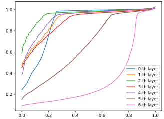
Miyato et al. (2018) > Orth. Reg. > SN (Alternative)

Better than Spectral Normalization

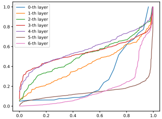
Singular Value Decay: Decay patterns of sorted singular values of weight matrices.



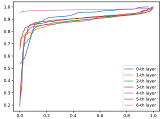
Orthogonal Reg.
No Decay
IS: 8.77



Miyato et al. 2018
Slow Decay
IS: 8.83



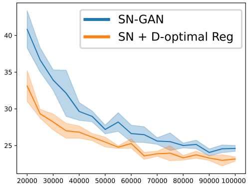
SN (Alt.)
Fast Decay
IS: 8.69



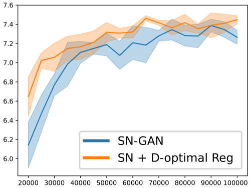
Jiang et al. (2019)
Slower Decay
IS: **9.25**

Observation: Slow singular value decay is better than both no decay and fast decay.

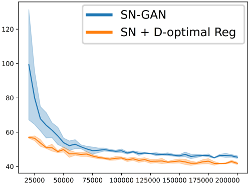
Experiments (CIFAR10 and STL-10)



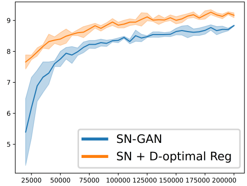
CIFAR: FID



CIFAR: Inception Score

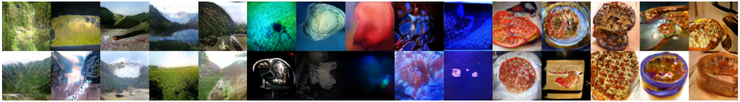


STL: FID



STL: Inception Score

Experiments (ImageNet)



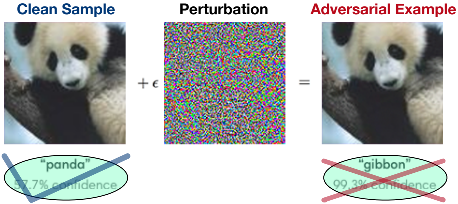
Valley Jellyfish Pizza

Anemone Shoji Brain Coral

Cardoon Altar Jack-o'-lantern

Adversarial Robust Learning

Adversarial Training



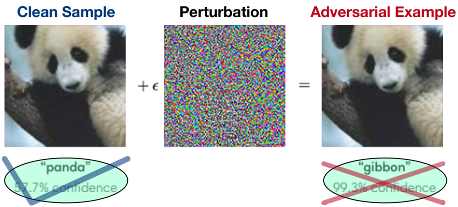
Highly Nonconvex-Nonconcave Minimax Problem:

$$\min_{\theta} \frac{1}{n} \sum_{i=1}^{n} (\max_{\delta_i \in \mathcal{B}} \ell(f(x_i + \delta_i; \theta), y_i).$$

x_i: feature; y_i: label; δ_i: perturbation;

$f(\cdot; \theta)$: neural network; ℓ: loss function; $\mathcal{B}$: constraint;

Adversarial Training



Highly Nonconvex-Nonconcave Minimax Problem:

$$\min_{\theta} \frac{1}{n} \sum_{i=1}^{n} (\max_{\delta_i \in \mathcal{B}} \ell(f(x_i + \delta_i; \theta), y_i).$$

x_i: feature; y_i: label; δ_i: perturbation;

$f(\cdot;\theta)$: neural network; ℓ: loss function; $\mathcal{B}$: constraint;

Adversarial Training

$$\min_{\theta} \frac{1}{n} \sum_{i=1}^{n} (\max_{\delta_i \in \mathcal{B}} \ell(f(x_i + \delta_i; \theta), y_i).$$

Two Stage Optimization:

- Inner Maximization Problem (Attack)
- Outer Minimization Problem (Defense)

Popular Approaches for Attack:

- Fast Gradient Sign Method (Goodfellow et al. (2014))
- Projected Gradient Method (Kurakin et al. (2016))
- Carlini-Wagner Attack (Paszke et al. (2017))

Adversarial Training

$$\min_{\theta} \frac{1}{n} \sum_{i=1}^{n} (\max_{\delta_i \in \mathcal{B}} \ell(f(x_i + \delta_i; \theta), y_i).$$

Two Stage Optimization:

- Inner Maximization Problem (Attack)
- Outer Minimization Problem (Defense)

Popular Approaches for Attack:

- Fast Gradient Sign Method (Goodfellow et al. (2014))
- Projected Gradient Method (Kurakin et al. (2016))
- Carlini-Wagner Attack (Paszke et al. (2017))

Adversarial Training

$$\min_{\theta} \frac{1}{n} \sum_{i=1}^{n} (\max_{\delta_i \in \mathcal{B}} \ell(f(x_i + \delta_i; \theta), y_i).$$

Two Stage Optimization:

- Inner Maximization Problem (Attack)
- Outer Minimization Problem (Defense)

Popular Approaches for Attack:

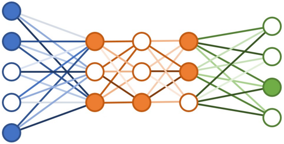
- Fast Gradient Sign Method (Goodfellow et al. (2014))
- Projected Gradient Method (Kurakin et al. (2016))
- Carlini-Wagner Attack (Paszke et al. (2017))

Learn to Learn/Optimize (L2L)

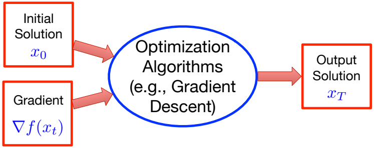
High Level Idea:

- Cast the optimizer as a **learning** model;
- Allow the model to learn to **exploit** structure automatically.

Implementation: Parameterize **optimizer** as a neural network, and learn its parameters (Andrychowicz et al. 2016).



V. S.



Learn to Learn/Optimize (L2L)

Advantages:

- **Attacker** Network is powerful in representation.
  $\Longrightarrow$ Yield **strong** and **flexible** perturbations.
- Shared attacker model.
  $\Longrightarrow$ Learn **common** structures across all perturbations.
- Learning through overparametrization.
  $\Longrightarrow$ **Ease** the training process.
- Reduce search space.
  $\Longrightarrow$ Computational efficiency

Learn to Learn/Optimize (L2L)

New Formulation:

$$\min_{\theta}\max_{\phi}\frac{1}{n}\sum_{i=1}^{n}\left[\ell(f(x_i+g(\mathcal{A}(x_i,y_i,\theta);\phi);\theta),y_i)\right],$$

Notations:

- $f(\cdot;\theta)$: Classifier;
- $g(\cdot;\phi)$: Attacker/Optimizer;
- $\mathcal{A}(x_i,y_i,\theta)$: Input of Optimizer g (**Interact g with f via $\mathcal{A}$**).

Learn to Learn/Optimize (L2L)

New Formulation:

$$\min_{\theta} \max_{\phi} \frac{1}{n} \sum_{i=1}^{n} \left[\ell(f(x_i + g(\mathcal{A}(x_i, y_i, \theta); \phi); \theta), y_i) \right],$$

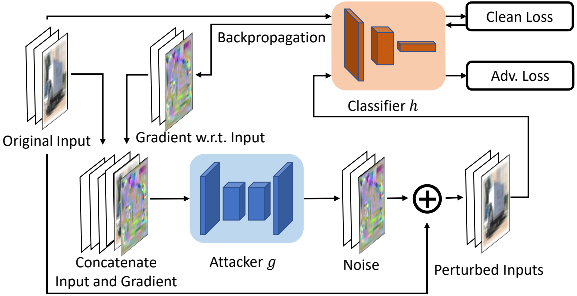
Notations:

- $f(\cdot;\theta)$: Classifier;
- $g(\cdot;\phi)$: Attacker/Optimizer;
- $\mathcal{A}(x_i, y_i, \theta)$: Input of Optimizer g (**Interact g with f via $\mathcal{A}$**).

Learn to Attack:

Grad L2L: Motivated by gradient ascent with

$$\mathcal{A}(x_i, y_i, \theta) = [x_i, \nabla_x \ell(f(x_i; \theta), y_i)].$$

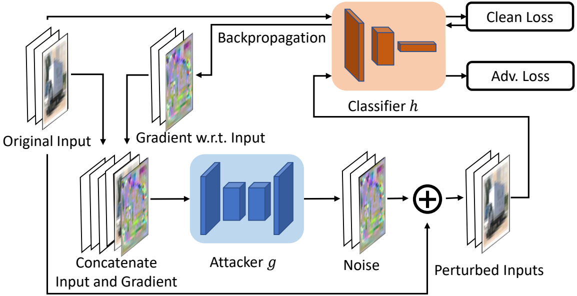


Multi-Step Grad L2L: Recursively apply Grad L2L (RNN).

Learn to Attack:

Grad L2L: Motivated by gradient ascent with

$$\mathcal{A}(x_i, y_i, \theta) = [x_i, \nabla_x \ell(f(x_i; \theta), y_i)].$$

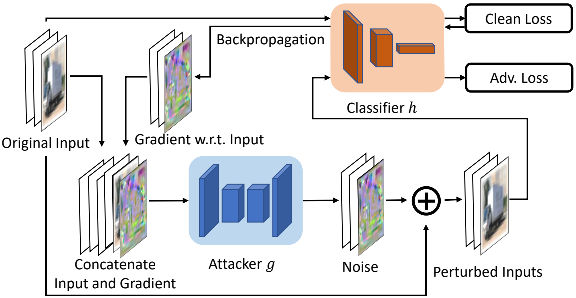


Multi-Step Grad L2L: Recursively apply Grad L2L (RNN).

Learn to Attack:

Grad L2L: Motivated by gradient ascent with

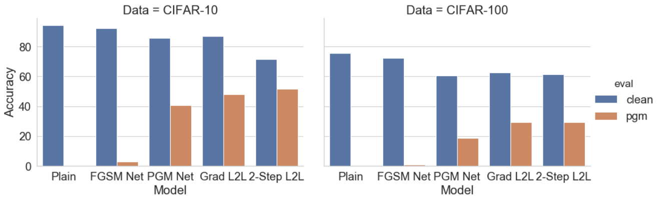
$$\mathcal{A}(x_i, y_i, \theta) = [x_i, \nabla_x \ell(f(x_i; \theta), y_i)].$$



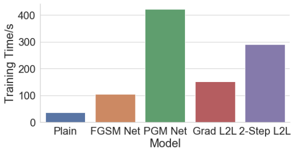
Multi-Step Grad L2L: Recursively apply Grad L2L (RNN).

Experiments

Accuracy on Clean Samples and PGM adversaries



Per Iteration Computational Cost



Reinforcement Learning

Smoothed Bellman Error Minimization

Minimax Formulation: Given $\mathcal{M} = (\mathcal{A}, \mathcal{A}, P, R, \gamma)$, we solve

$$\min_{\pi,V} \max_{\nu} L(\pi, V; \nu) = 2\mathbb{E}_{s,a,s'}[\nu(s,a)(R(s,a) + \gamma V(s') - \lambda \log(\pi(a|s))] - E_{s,a,s'}\nu^2(s,a),$$

where s denotes the state, a denotes the action, and

- Policy: $\pi: \mathcal{S} \to \mathcal{P}(\mathcal{A})$,
- Value: $V: \mathcal{S} \to R$,
- Reward: $R: \mathcal{S} \times \mathcal{A} \to R$,
- Axillary Dual: $\nu: \mathcal{S} \times \mathcal{A} \to R$.

The policy π and ν are parameterized as a neural network and a reproducing kernel function, respectively (Dai et al. 2018).

Smoothed Bellman Error Minimization

Minimax Formulation: Given $\mathcal{M} = (\mathcal{A}, \mathcal{A}, P, R, \gamma)$, we solve

$$\min_{\pi,V} \max_{\nu} L(\pi, V; \nu) = 2\mathbb{E}_{s,a,s'}[\nu(s,a)(R(s,a) + \gamma V(s') - \lambda \log(\pi(a|s))] - E_{s,a,s'}\nu^2(s,a),$$

where s denotes the state, a denotes the action, and

- Policy: $\pi : \mathcal{S} \to \mathcal{P}(\mathcal{A})$,
- Value: $V : \mathcal{S} \to R$,
- Reward: $R : \mathcal{S} \times \mathcal{A} \to R$,
- Axillary Dual: $\nu : \mathcal{S} \times \mathcal{A} \to R$.

The policy π and ν are parameterized as a neural network and a reproducing kernel function, respectively (Dai et al. 2018).

Smoothed Bellman Error Minimization

Minimax Formulation: Given $\mathcal{M} = (\mathcal{A}, \mathcal{A}, P, R, \gamma)$, we solve

$$\min_{\pi, V} \max_{\nu} L(\pi, V; \nu) = 2\mathbb{E}_{s,a,s'}[\nu(s,a)(R(s,a) + \gamma V(s') - \lambda \log(\pi(a|s))] - E_{s,a,s'}\nu^2(s,a),$$

where s denotes the state, a denotes the action, and

- Policy: $\pi : \mathcal{S} \to \mathcal{P}(\mathcal{A})$,
- Value: $V : \mathcal{S} \to R$,
- Reward: $R : \mathcal{S} \times \mathcal{A} \to R$,
- Axillary Dual: $\nu : \mathcal{S} \times \mathcal{A} \to R$.

The policy π and ν are parameterized as a neural network and a reproducing kernel function, respectively (Dai et al. 2018).

Parameterization of V, π and ν

State Approximation: There exists a feature vector $\psi(s)$ associated with every state $s \in \mathcal{S}$.

Neural Networks for π and V:

$$\pi(a_j|s) = f_j(\psi(s); \Theta) \quad \text{and} \quad V(s) = h(\psi(s), \Delta),$$

where f_j is a neural network with parameter Θ and $\sum_{a_j \in \mathcal{A}} \pi(a_j|s) = 1$.

Reproducing Kernel Functions for ν:

$$\nu(a_j|s) = g_j(\psi(s); \Omega),$$

where g_j is a reproducing kernel function with parameter Ω.

Benefit of Reproducing Kernel Parameterization

Alternative Minimax Formulation:

$$\min_{\Delta,\Theta} \max_{\Omega \in \mathcal{C}} \mathcal{L}(\Delta, \Theta, \Omega) - \mathcal{R}(\Omega)$$

,

where $\mathcal{R}(\Omega)$ is a strongly concave regularizer.

Stochastic Alternating Gradient Algorithm:

$$\Omega^{(t+1)} = \Pi_{\mathcal{C}}(\Omega^{(t)} + \eta_\Omega \nabla_\Omega \widetilde{L}(\Delta^{(t)}, \Theta^{(t)}, \Omega^{(t)})),$$
$$\Delta^{(t+1)} = \Delta^{(t)} - \eta_\Delta \nabla_\Delta \widetilde{L}'(\Delta^{(t)}, \Theta^{(t)}, \Omega^{(t+1)}),$$
$$V^{(t+1)} = V^{(t)} - \eta_V \nabla_V \widetilde{L}'(\Delta^{(t)}, \Theta^{(t)}, \Omega^{(t+1)}),$$

where η_V, η_Δ and η_Ω are properly chosen step sizes, and $\widetilde{L}$ and $\widetilde{L}'$ are unbiased independent stochastic approximations of $\mathcal{L}$.

Benefit of Reproducing Kernel Parameterization

Alternative Minimax Formulation:

$$\min_{\Delta,\Theta}\max_{\Omega\in\mathcal{C}}\mathcal{L}(\Delta,\Theta,\Omega)-\mathcal{R}(\Omega)$$

,

where $\mathcal{R}(\Omega)$ is a strongly concave regularizer.

Stochastic Alternating Gradient Algorithm:

$$\begin{aligned}
\Omega^{(t+1)} &= \Pi_{\mathcal{C}}(\Omega^{(t)}+\eta_\Omega\nabla_\Omega\widetilde{L}(\Delta^{(t)},\Theta^{(t)},\Omega^{(t)})),\\
\Delta^{(t+1)} &= \Delta^{(t)}-\eta_\Delta\nabla_\Delta\widetilde{L}'(\Delta^{(t)},\Theta^{(t)},\Omega^{(t+1)}),\\
V^{(t+1)} &= V^{(t)}-\eta_V\nabla_V\widetilde{L}'(\Delta^{(t)},\Theta^{(t)},\Omega^{(t+1)}),
\end{aligned}$$

where η_V, η_Δ and η_Ω are properly chosen step sizes, and $\widetilde{L}$ and $\widetilde{L}'$ are unbiased independent stochastic approximations of $\mathcal{L}$.

Sublinear Convergence

Theorem (Informal, Chen et al. 2019)

Given a pre-specified error $\epsilon > 0$, we assume that $\mathcal{L}(\Delta, \Theta, \Omega)$ is sufficiently smooth in $\Delta, \Theta, \Omega \in \mathcal{C}$, and strongly concave in Ω. Given properly chosen step sizes and a batch size of $O(1/\epsilon)$, we need at most

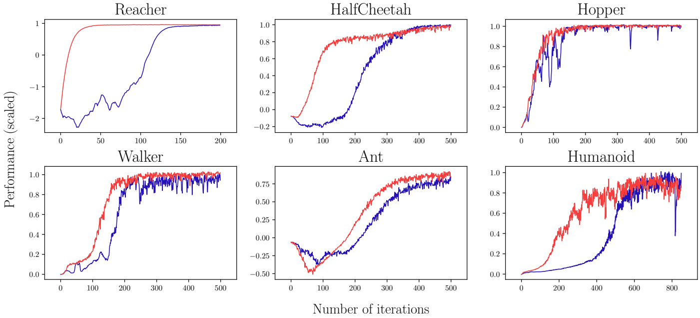
$$T = \widetilde{O}(1/\epsilon)$$

iterations such that

$$\min_{1 \le t \le T} \mathbb{E} \left\| \nabla_\Delta \mathcal{L}(\Delta^t, \Theta^{(t)}, \Omega^{(t+1)}) \right\|_2^2 + \mathbb{E} \left\| \nabla_\Theta \mathcal{L}(\Delta^t, \Theta^{(t)}, \Omega^{(t+1)}) \right\|_2^2$$
$$+ \mathbb{E} \left\| \Omega^{(t)} - \Pi_{\mathcal{C}}(\Omega^{(t)} + \nabla_\Omega \widetilde{L}(\Delta^{(t)}, \Theta^{(t)}, \Omega^{(t)})) \right\|_2^2 \le \epsilon.$$

Experiments

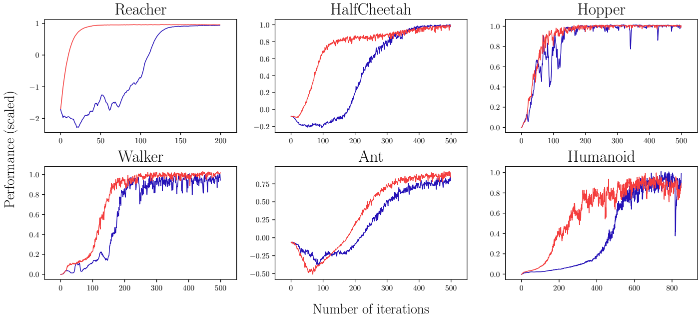
Reproducing Kernel v.s. Neural Networks for ν.



The reproducing kernel parameterization leads to an easier optimization problem. However, it might not be advantageous on more complicated problems.

Experiments

Reproducing Kernel v.s. Neural Networks for ν.



The reproducing kernel parameterization leads to an easier optimization problem. However, it might not be advantageous on more complicated problems.

Take Home Messages

Summary

- Minimax optimization is very difficult in general;
- Heuristics leverage specific structures in machine learning problems;
- Normalization techniques improve the optimization landscape, and stabilize the training of GAN;
- The learning to optimize techniques have potentials to outperform hand-designed algorithms;
- The "large-batch" stochastic alternating gradient descent attains sublinear convergence to some stationary solution for nonconvex-concave stochastic minimax optimization problems;
- Lots of new problems, and open to everyone!

References

[1] Jiang et al., “On Computation and Generalization of Generative Adversarial Networks under Spectrum Control”. International Conference on Learning Representations (ICLR), 2019

[2] Jiang et al., “Learning to Defense by Learning to Attack”. ICLR Workshop on Deep Generative Models for Highly Structured Data, 2019

[3] Chen et al., “On Computation and Generalization of Generative Adversarial Imitation Learning”. Submitted.

[4] Chen et al., “On Landscape of Lagrangian Functions and Stochastic Search for Constrained Nonconvex Optimization”. International Conference on Artificial Intelligence and Statistics (AISTATS), 2019

[5] Liu et al., “Deep Hyperspherical Learning”. Annual Conference on Neural Information Processing Systems (NIPS), 2017

[6] Li et al. “Symmetry, Saddle Points and Global Optimization Landscape of Nonconvex Matrix Factorization”, IEEE Transactions on Information Theory (TIT), 2019.

Questions?