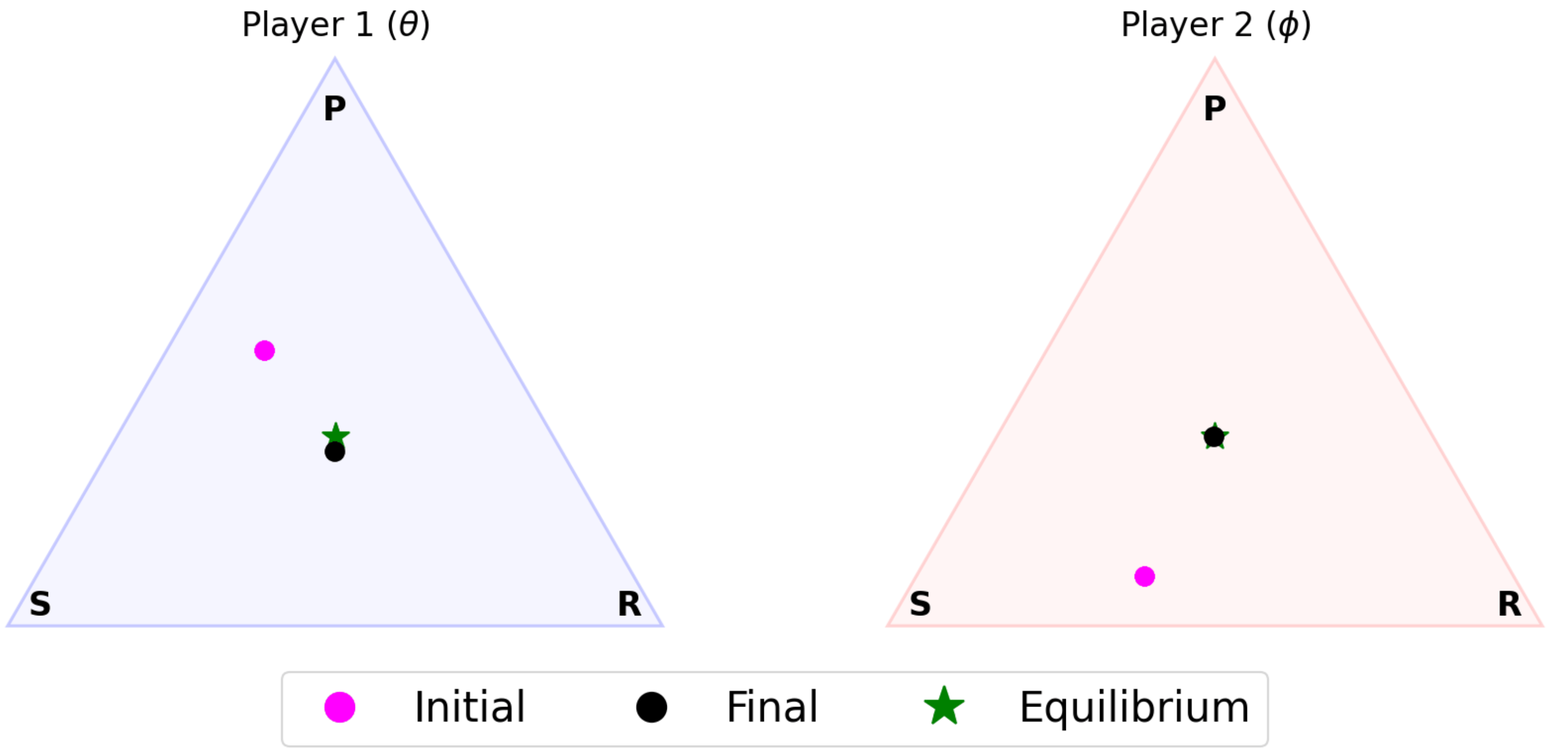
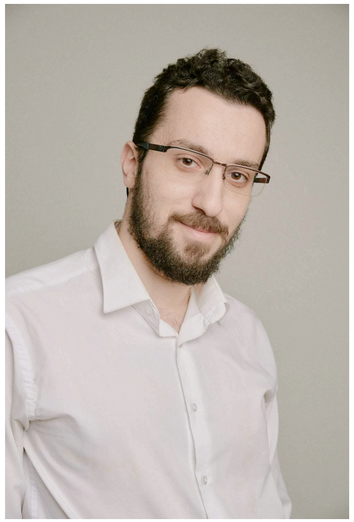
Solving Neural Min-Max Games: The Role of Architecture, Initialization & Dynamics

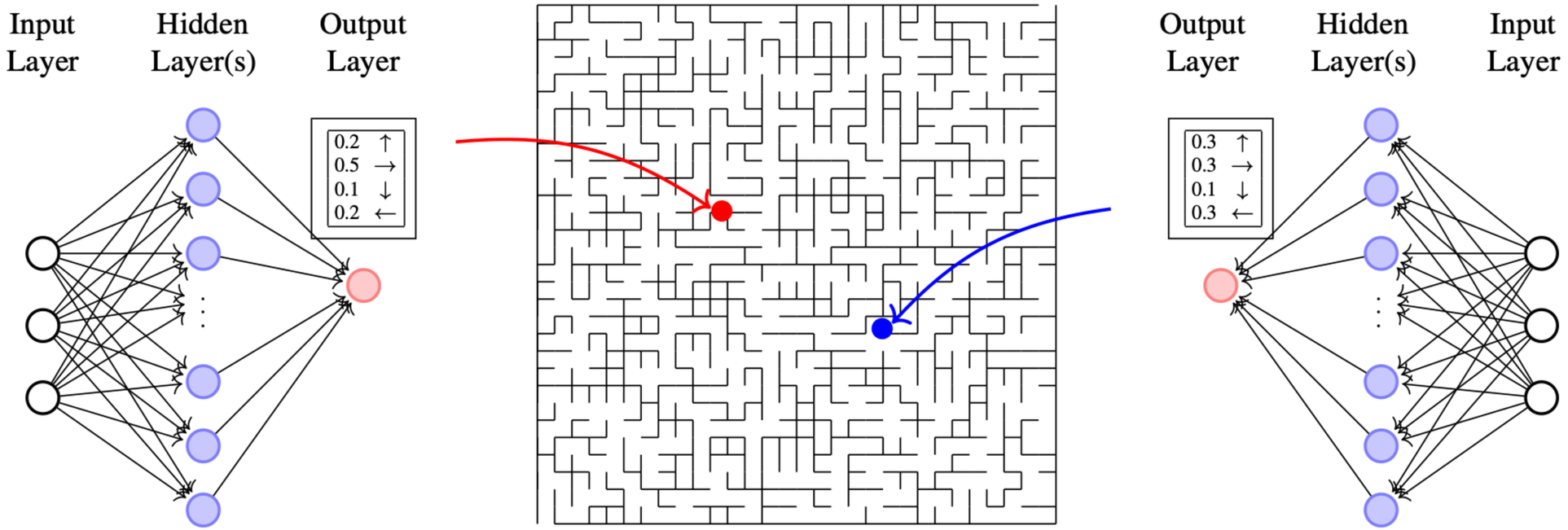




Deep Patel* & **Manolis Vlatakis (UW-Madison)**

NeurIPS 2025 (Spotlight)

The question at the heart of this paper

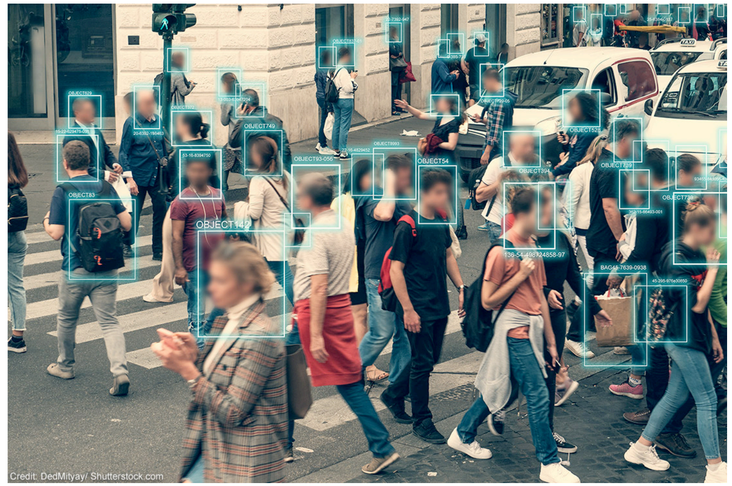


How can two neural networks be designed and trained to compute a solution to a zero-sum game?

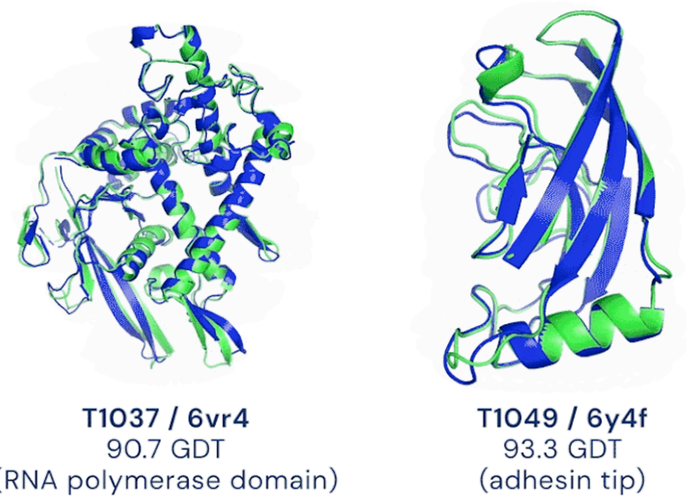
Success of Deep Learning

https://shorturl.at/2COlv

ChatGPT



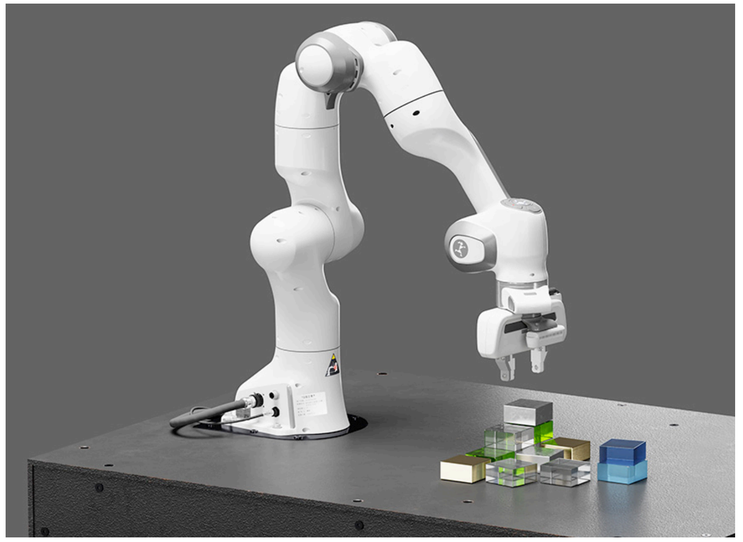
https://shorturl.at/eXGBP



T1037 / 6vr4
90.7 GDT
(RNA polymerase domain)

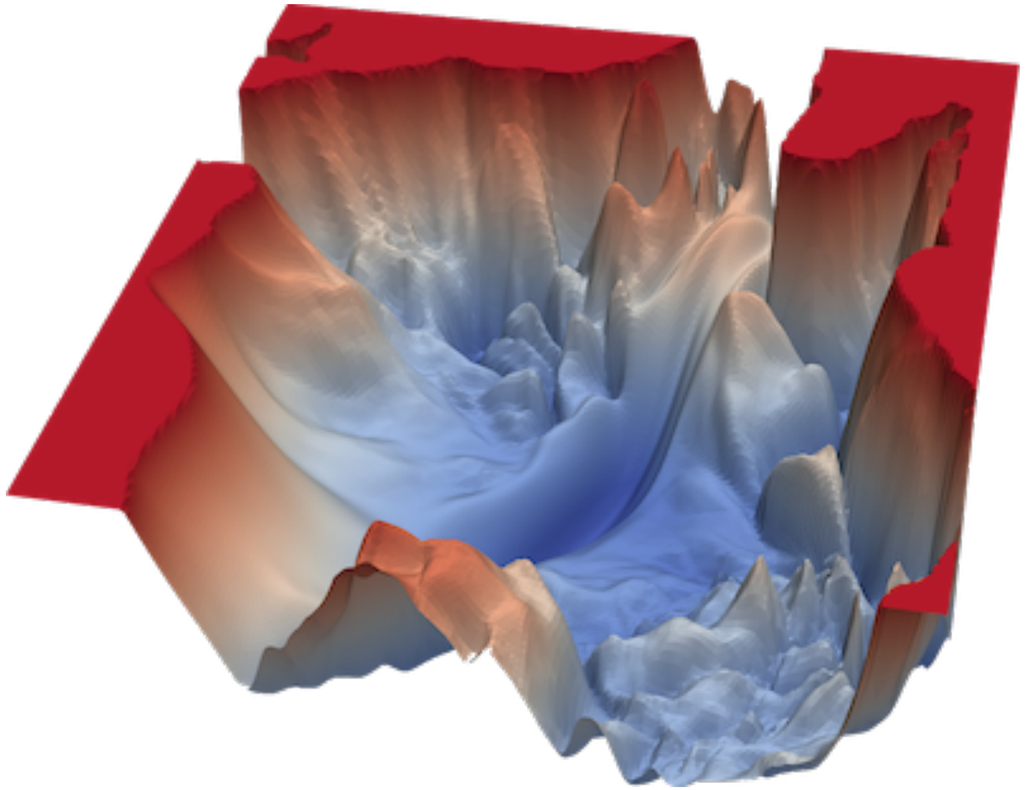
T1049 / 6y4f
93.3 GDT
(adhesin tip)

https://shorturl.at/6Oxs4



https://shorturl.at/ya6zQ

How Theory Tries to Understand Success of Deep Learning



Gradient Descent Finds Global Minima of Deep Neural Networks

Simon Du, Jason Lee, Haochuan Li, Liwei Wang, Xiyu Zhai *Proceedings of the 36th International Conference on Machine Learning*, PMLR 97:1675-1685, 2019.

A Convergence Theory for Deep Learning via Over-Parameterization

Zeyuan Allen-Zhu, Yuanzhi Li, Zhao Song *Proceedings of the 36th International Conference on Machine Learning*, PMLR 97:242-252, 2019.

The Loss Surface of Deep and Wide Neural Networks

Quynh Nguyen, Matthias Hein *Proceedings of the 34th International Conference on Machine Learning*, PMLR 70:2603-2612, 2017.

How SGD Selects the Global Minima in Over-parameterized Learning: A Dynamical Stability Perspective

Gradient descent optimizes over-parameterized deep ReLU networks

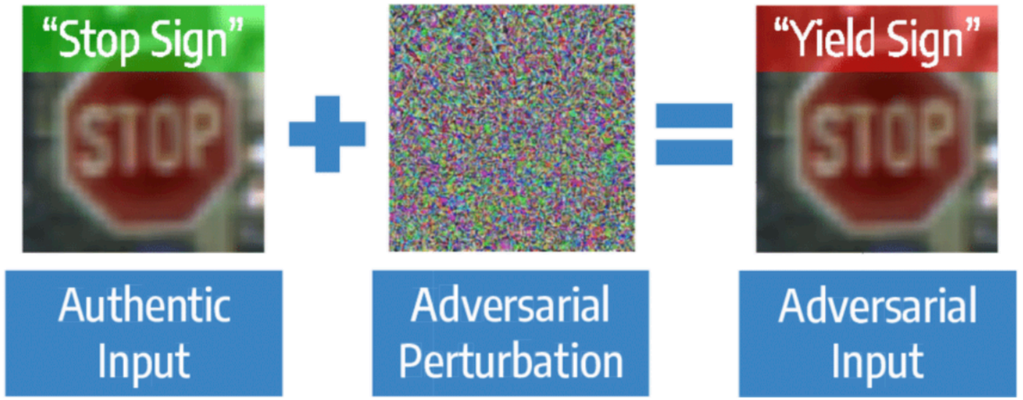
Difan Zou, Yuan Cao, Dongruo Zhou, Quanquan Gu

Image Source: https://www.cs.umd.edu/~tomg/project/landscapes/

Naturally we may ask…

How big a neural network should be so that vanilla methods like (S)GD can converge to global optima?

Rise of Multi-Agent Learning Applications



https://shorturl.at/e7Xbw

https://shorturl.at/krf6V

https://shorturl.at/Opeki

https://shorturl.at/DH09f

https://shorturl.at/l0g1p

Rise of Multi-Agent Learning Applications

- We are now modelling multiple agents learning and making decisions in a non-stationary environment that can react to these decisions. For example,
  - Agents having conflicting interests/objectives
  - Adversaries that can change/corrupt the data/distribution (label noise, distribution shifts)
  - Enforce constraints on learnt models such as those relating to causal inference, privacy, and fairness (and more).
- **This work: Two-player zero-sum games $\Longrightarrow$ Focus on MIN-MAX optimization.**

Moreover, let's recall that…

Generative Adversarial Networks [Goodfellow et al. 16]

$$\arg\min_{\boldsymbol{\theta}^{(G)}} \max_{\boldsymbol{\theta}^{(D)}} V\left(\boldsymbol{\theta}^{(D)}, \boldsymbol{\theta}^{(G)}\right).$$

The minimax game is mostly of interest because it is easily amenable to theoretical analysis. Goodfellow *et al.* (2014b) used this variant of the GAN game to show that learning in this game resembles minimizing the Jensen-Shannon divergence between the data and the model distribution, and that the game converges to its equilibrium if both players' policies can be updated directly in function space. In practice, the players are represented with deep neural nets and updates are made in parameter space, so these results, which depend on convexity, do not apply.

Rise of Multi-Agent Learning Applications

- Environments with large, possibly continuous state and action spaces (e.g. StarCraft II, Go, etc.)
- Neural networks have universal approximation property
- Encoding agents' strategies/policies with neural networks $\Longrightarrow$ Richer strategic behaviour
- **This work: Players are 1-hidden layer (i.e. shallow) neural networks.**

Rise of Multi-Agent Learning Applications

- Environments with large, n… …n spaces (e.g. StarCraft II, Go…
- Neural network…
- Encoding agent… …Richer strategic behavi…
- **This work: Playe… …hidden layer (i.e. shallow) neural networks.**

Hidden Games

Quick Look: Hidden(-Convex/Concave) Zero-Sum Games

Input Games

$$\min_{x_{Alice} \in \mathcal{D}_F} \max_{x_{Bob} \in \mathcal{D}_G} L(F(x_{Alice}; \theta), G(x_{Bob}; \phi))$$

- Parameters θ, ϕ are fixed
- Optimising over inputs
- Example: Adversarial example generation

Quick Look: Hidden(-Convex/Concave) Zero-Sum Games

Input Games

$$\min_{x_{Alice} \in \mathcal{D}_F} \max_{x_{Bob} \in \mathcal{D}_G} L(F(x_{Alice}; \theta), G(x_{Bob}; \phi))$$

- Parameters θ, ϕ are fixed
- Optimising over inputs
- Example: Adversarial example generation

Loss L is non-convex (non-concave) in θ (ϕ)

Quick Look: Hidden(-Convex/Concave) Zero-Sum Games

Input Games

$$\min_{x_{Alice} \in \mathcal{D}_F} \max_{x_{Bob} \in \mathcal{D}_G} L(F(x_{Alice}; \theta), G(x_{Bob}; \phi))$$

- Parameters θ, ϕ are fixed
- Optimising over inputs
- Example: Adversarial example generation

Loss L is convex (concave) in $F(\cdot\,; \theta)$ ($G(\cdot\,; \phi)$)

Loss L is non-convex (non-concave) in θ (ϕ)

Quick Look: Hidden(-Convex/Concave) Zero-Sum Games

Input Games

$$\min_{x_{Alice} \in \mathcal{D}_F} \max_{x_{Bob} \in \mathcal{D}_G} L(F(x_{Alice}; \theta), G(x_{Bob}; \phi))$$

- Parameters θ, ϕ are fixed
- Optimising over inputs
- Example: Adversarial example generation

Neural Games

$$(\theta^\star, \phi^\star) \in \arg\min_{\theta \in \mathbb{R}^m} \arg\max_{\phi \in \mathbb{R}^n} \mathbb{E}_{(x,x') \sim P_{xx'}} \left[L(F(x; \theta), G(x'; \phi)) \right]$$

- Optimize over parameters θ, ϕ for given data
- GANs, DIRL, etc.

Quick Look: Hidden(-Convex/Concave) Zero-Sum Games

Input Games

$$\min_{x_{Alice} \in \mathcal{D}_F} \max_{x_{Bob} \in \mathcal{D}_G} L(F(x_{Alice}; \theta), G(x_{Bob}; \phi))$$

- Parameters θ, ϕ are fixed
- Optimising over inputs
- Example: Adversarial example generation

Neural Games

$$(\theta^\star, \phi^\star) \in \arg\min_{\theta \in \mathbb{R}^m} \arg\max_{\phi \in \mathbb{R}^n} \mathbb{E}_{(x,x') \sim P_{xx'}} \left[L(F(x; \theta), G(x'; \phi)) \right]$$

- Optimize over parameters θ, ϕ for given data
- GANs, DIRL, etc.

Loss L is non-convex (non-concave) in θ (ϕ)

Quick Look: Hidden(-Convex/Concave) Zero-Sum Games

Input Games

$$\min_{x_{Alice} \in \mathcal{D}_F} \max_{x_{Bob} \in \mathcal{D}_G} L(F(x_{Alice}; \theta), G(x_{Bob}; \phi))$$

- Parameters θ, ϕ are fixed
- Optimising over inputs
- Example: Adversarial example generation

Loss L is convex (concave) in $F(\cdot\,; \theta)$ ($G(\cdot\,; \phi)$)

Neural Games

$$(\theta^\star, \phi^\star) \in \arg\min_{\theta \in \mathbb{R}^m} \arg\max_{\phi \in \mathbb{R}^n} \mathbb{E}_{(x,x') \sim P_{xx'}} \left[L(F(x; \theta), G(x'; \phi)) \right]$$

- Optimize over parameters θ, ϕ for given data
- GANs, DIRL, etc.

Loss L is non-convex (non-concave) in θ (ϕ)

Naturally we may ask…

How big a neural network should be so that vanilla methods like ~~(S)GD~~ AltGDA can converge to ~~global optima~~ a saddle point?

Our Results

Input Games

$$\min_{x_{Alice} \in \mathcal{D}_F} \max_{x_{Bob} \in \mathcal{D}_G} L(F(x_{Alice}; \theta), G(x_{Bob}; \phi))$$

Theorem 1 (Informal). For $\varepsilon - \ell_2$-reg. bilinear zero-sum hidden games, w.h.p. AltGDA converges to ε-saddle point if both shallow neural network players have a "good" Gaussian random init.

Our Results

Input Games

$$\min_{x_{Alice} \in \mathcal{D}_F} \max_{x_{Bob} \in \mathcal{D}_G} L(F(x_{Alice}; \theta), G(x_{Bob}; \phi))$$

Theorem 1 (Informal). For $\varepsilon - \ell_2$-reg. bilinear zero-sum hidden games, w.h.p. AltGDA converges to ε-saddle point if both shallow neural network players have a "good" Gaussian random init.

Neural Games

$$(\theta^\star, \phi^\star) \in \arg\min_{\theta \in \mathbb{R}^m} \arg\max_{\phi \in \mathbb{R}^n} \mathbb{E}_{(x,x') \sim P_{xx'}} \left[L(F(x; \theta), G(x'; \phi)) \right]$$

Theorem 2 (Informal). For a broad class of hidden-convex-concave zero-sum games, w.h.p. AltGDA converges to a saddle point if both shallow neural network players have a "good" Gaussian random init. and if their hidden-layer widths scale as n^3 where n is the size of the dataset.

Thank you!

Poster ID: 119822

Thu 4 Dec (4:30—7:30 PM)