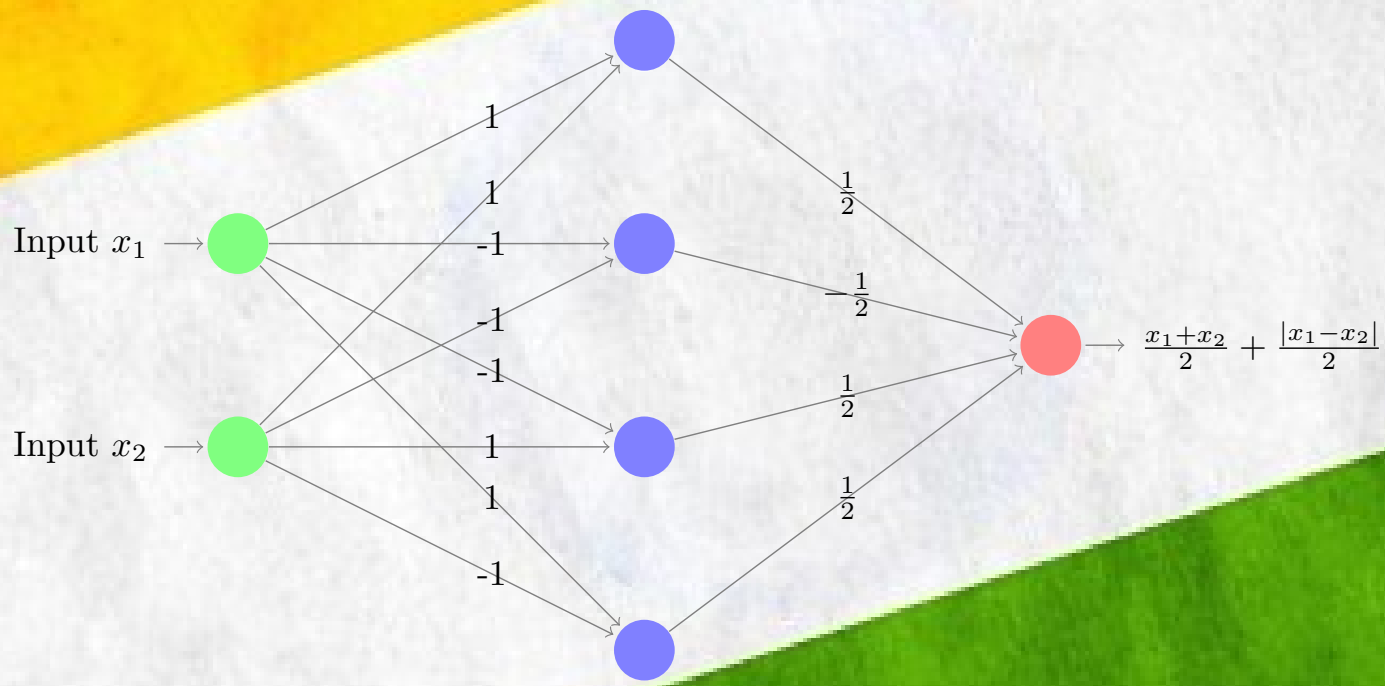
Provable Training Of Two Layer Neural Nets

Anirbit

January 19, 2023

Overview

A depth 2, width 4 net computing the max of 2 numbers - via $\sigma : x \mapsto \max\{0, x\}$ "activation" at the (blue) gates.



Defining Neural Nets

The key component of a neural net is an "activation function" and the most widely used one is the "Rectified Linear Unit (ReLU)",

$$\mathrm{ReLU} : \mathbb{R}^n \ni \mathbf{x} \mapsto (\max\{0, x_1\}, \max\{0, x_2\}, \ldots, \max\{0, x_n\}) \in \mathbb{R}^n$$

What is a neural net?

Given $\left\{ \mathrm{A}_i : \mathbb{R}^{w_{i-1}} \to \mathbb{R}^{w_i} \mid i = 1, \ldots, k+1 \right\}$, a set of $k+1$ affine transformations, it defines a depth $k+1$ "ReLU Deep Neural Net (DNN)" as the following function,

$$\mathbb{R}^{w_0} \ni \mathbf{x} \mapsto \mathbf{N}(\mathbf{x}) = \mathrm{A}_{k+1} \circ \mathrm{ReLU} \circ \mathrm{A}_k \circ \cdots \circ \mathrm{A}_2 \circ \mathrm{ReLU} \circ \mathrm{A}_1 \in \mathbb{R}^{\mathbf{w}_{k+1}}$$

An Example of a $\mathbb{R}^4 \to \mathbb{R}^3$ Neural Architecture

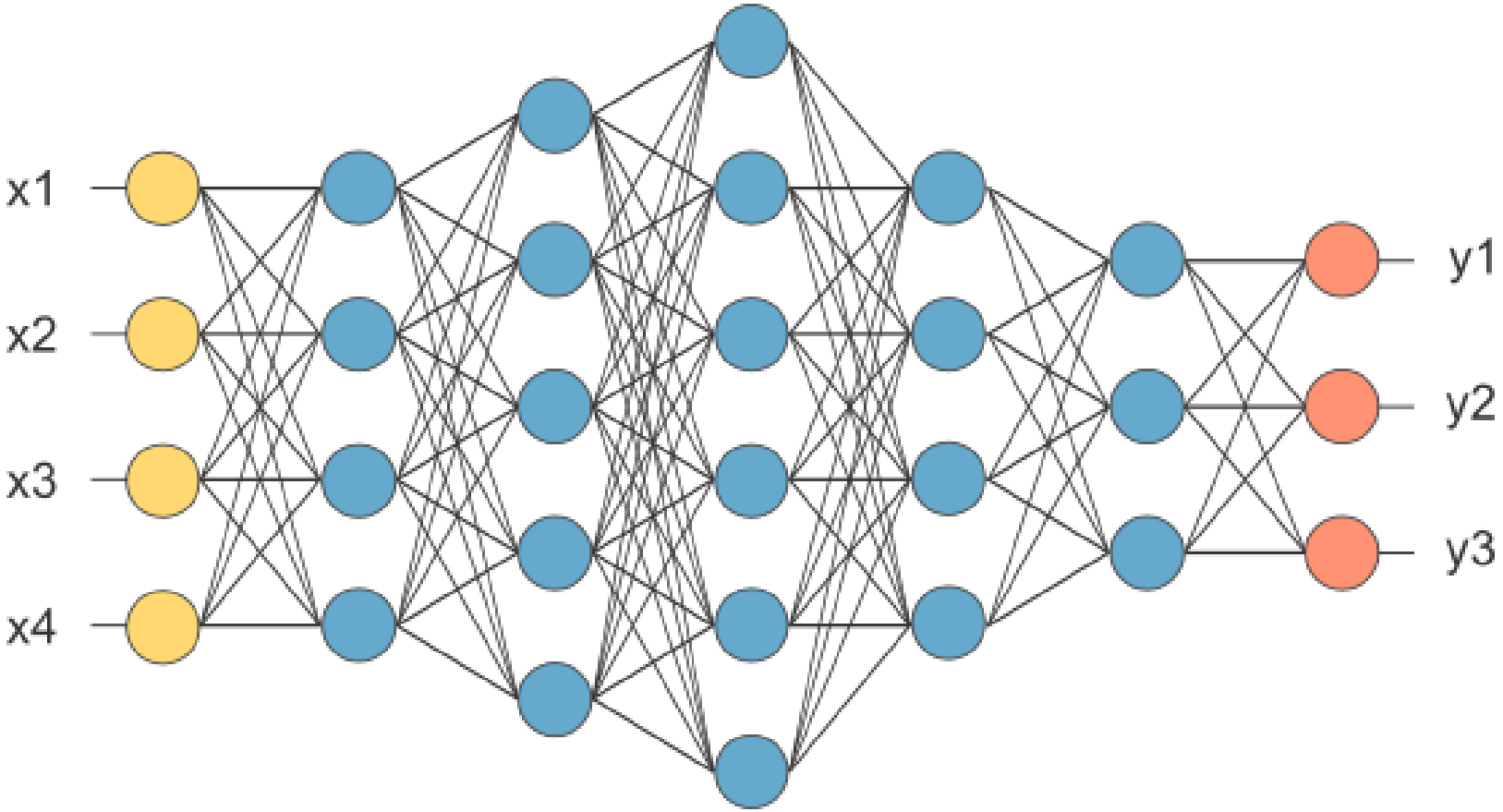


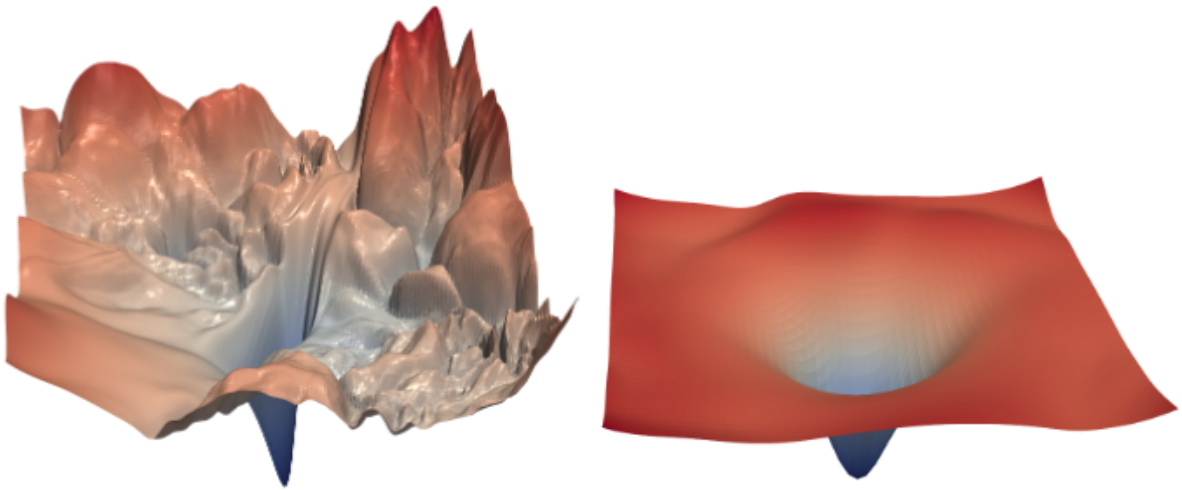
Figure: Unlike the max computing net, this is not a neural function because weights have not been assigned on the edges. **Such a diagram/architecture defines a certain set of neural functions.**

Why Are Neural Functions Exciting?

The above realistic human portraits are outputs of a neural net!

The Neural Landscape Makes Optimization Very Hard! (But Why Do Simple Algorithms Still Work?)

Training neural networks requires minimizing a high-dimensional non-convex loss function – a task that is hard in theory, but sometimes easy in practice. Despite the NP-hardness of training general neural loss functions [2], simple gradient methods often find global minimizers (parameter configurations with zero or near-zero training loss), even when data and labels are randomized before training [42]. However, this good behavior is not universal; the trainability of neural nets is highly dependent on network architecture design choices, the choice of optimizer, variable initialization, and a variety of other considerations. Unfortunately, the effect of each of these choices on the structure of the underlying loss surface is unclear. Because of the prohibitive cost of loss function evaluations (which requires looping over all the data points in the training set), studies in this field have remained predominantly theoretical.



(a) without skip connections (b) with skip connections

This is a famous diagram from the paper **"Visualizing the Loss Landscape of Neural Nets"** by Hao Li, Zheng Xu, Gavin Taylor,

What Is Provably Known About Neural Net Training?

In 2016 we showed the first exact empirical risk minimization on nets.

Theorem (**with R. Arora, A. Basu, P. Mianjy (ICLR 2018)**)

Empirical risk minimization on 1-DNN with a convex loss, such as $\min_{\mathbf{w}_i, a_i, b_i, b} \frac{1}{S} \sum_{i=1}^{S} \|\mathbf{y}_i - \sum_{p=1}^{\text{width}} a_p \max\{0, \langle \mathbf{w}_p, \mathbf{x}_i \rangle + b_p\}\|_2^2$ *can be done in time,* $\mathbf{2^{width} S^{n \times width}} \text{poly}(\mathbf{n}, \mathbf{S}, \mathbf{width})$.

Before the above result, it was not clear if there is any fundamental barrier to getting poly(training data size) **run-time for exact global minima finding on** ReLU **nets.**

Proof of trainability remains open in almost all realistic parameter regimes for nets i.e at finite sizes and without fixing to specific data distributions.

Our results in 2020 probed the challenging trifecta of,

(1) training a ReLU gate,
(2) while using non-realizable data,
AND
(3) wanting the algorithm to resemble SGD

Provable Iterative Non-Gradient Algorithm for Data-Poisoning Resistant Training of a ReLU Gate. (with Sayar, Neural Networks, Vol 151, 2022)

1: **Input:** An arbitrarily chosen starting point of $\mathbf{w}_1 \in \mathbb{R}^n$
2: **for** $t = 1, \ldots$ **do**
3: We sample independently $s_t := \{\mathbf{x}_{t_1}, \ldots, \mathbf{x}_{t_b}\} \sim \mathcal{D}$
- and query the oracle with this set.
4: The Oracle samples $\forall i = 1, \ldots, b, \alpha_{t_i} \sim \{0, 1\}$
- with probability $\{1 - \beta(x_{t_i}), \beta(x_{t_i})\}$
5: The Oracle replies $\forall i = 1, \ldots, b, \underbrace{\mathbf{y_{t_i}} = \alpha_{\mathbf{t_i}} \cdot \xi_{\mathbf{t_i}} + \mathrm{ReLU}(\mathbf{w}_*^\top \mathbf{x_{t_i}})}_{\text{Additively distorted true label!}}$
- s.t $|\xi_{t_i}| \leq \theta_*$
6: Form the gradient (proxy),

$$\mathbf{g}_t := -\frac{1}{b} \sum_{i=1}^{b} 1_{\{\mathbf{y_{t_i}} > \theta_*\}} (y_{t_i} - \mathbf{w}_t^\top \mathbf{x}_{t_i}) \mathbf{x}_{t_i}$$

7: $\mathbf{w}_{t+1} := \mathbf{w}_t - \eta \mathbf{g}_t$

Provable modified S.G.D and yet unproven true S.G.D are virtually indistinguishable for the ReLU gate!

(with Sayar, Neural Networks, Vol 151, 2022)

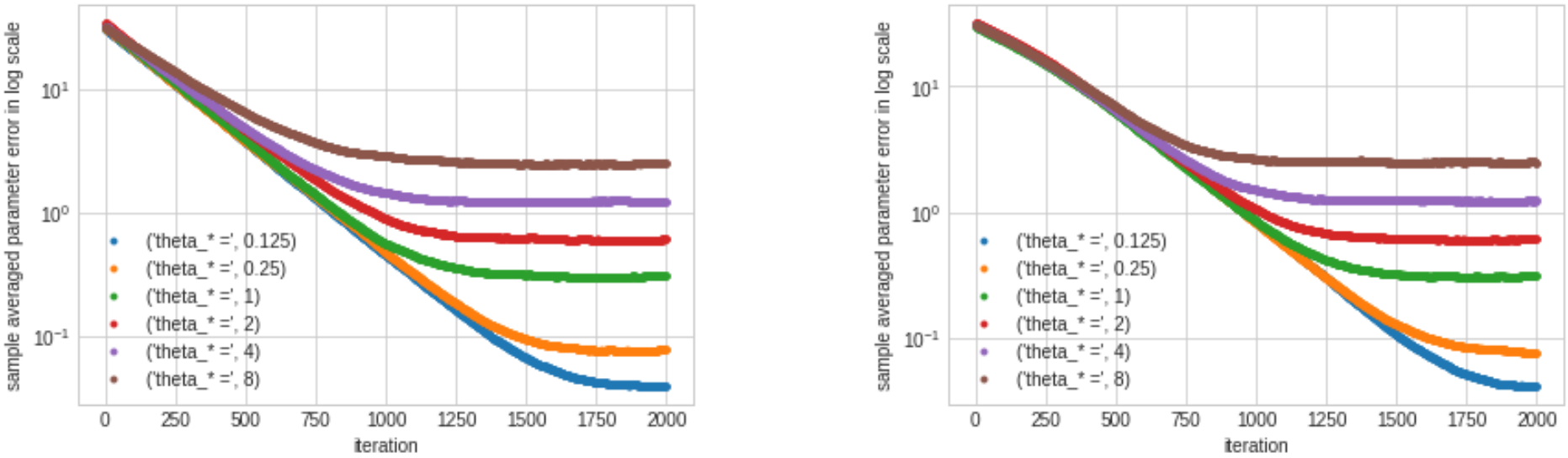


Figure: For the provable modified S.G.D on the (**Left**) and yet unproven S.G.D on the (**Right**), the above figures give the time evolution of the distance to the unperturbed risk minima for a $\mathrm{ReLU} : \mathbb{R}^{500} \to \mathbb{R}$ - the adversary is attacking with a probability of $\frac{1}{2}$ and the mini-batch size is 16 and we vary the allowed attack magnitude.

Provable modified S.G.D and yet unproven true S.G.D are virtually indistinguishable for the ReLU gate!

(with Sayar, Neural Networks, Vol 151, 2022)

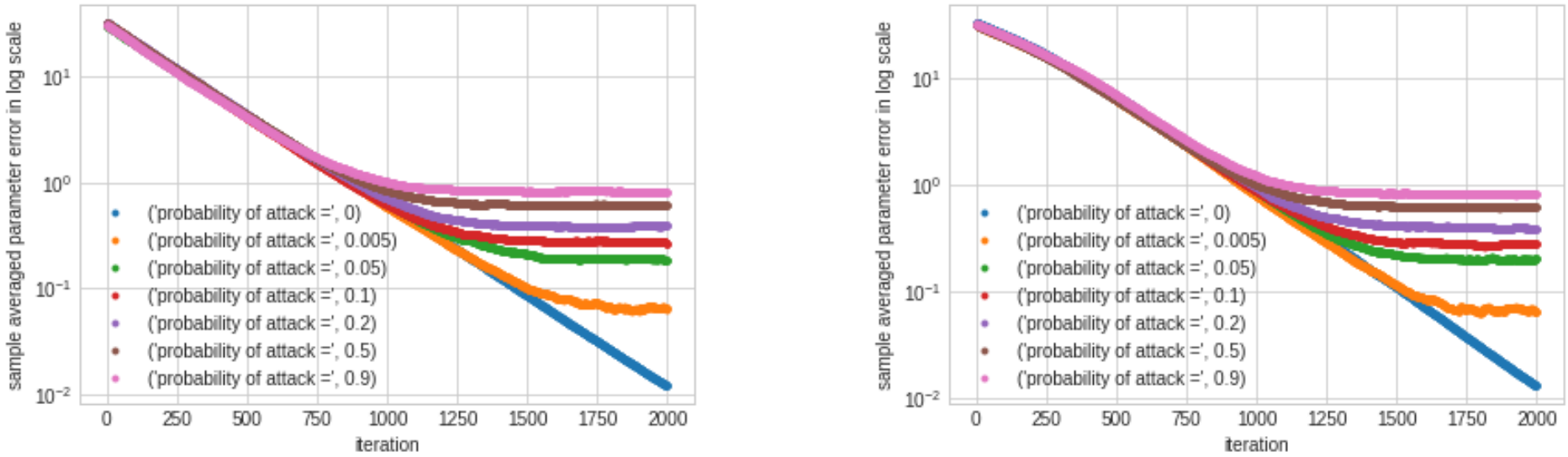


Figure: For the provable modified S.G.D on the (**Left**) and yet unproven S.G.D on the (**Right**), the above figures give the time evolution of the distance to the unperturbed risk minima for a $\mathrm{ReLU} : \mathbb{R}^{500} \to \mathbb{R}$ - the adversary is attacking with a maximum allowed magnitude of 2 and the mini-batch size is 16 and we vary the probability of attack.

Provable modified S.G.D and yet unproven true S.G.D are virtually indistinguishable for the ReLU gate!

(with Sayar, Neural Networks, Vol 151, 2022)

This modified-SGD of ours is possibly the only example of linear time convergence on a ReLU gate while neither fixing the functional form of the marginal distribution of the data nor making symmetry assumptions on it.

Additionally, in this case we can also give guarantees for the approximate convergence of the algorithm when the training labels are subjected to a stochastic and bounded additive poisoning attack.

Approaches To Gradient-Based Provable Neural Training

- Stochastic Gradient Descent (and its variants) - despite being a very simple algorithm - has been surprisingly widely successful in training deep-nets - for various data and various sizes!

- This success of SGD despite the complex non-convexity (and also non-differentiability) of the neural landscape has largely remained unexplained.

In the rest of the talk we shall establish (likely for the first time) that SGD can converge to the global minima of certain neural loss functions - at arbitrary width and data.

To put this in context lets first lets see the history of efforts here!

Approaches To Gradient-Based Provable Neural Training

- An active line of work explaining optimization in neural nets has been the Neural Tangent Kernel (NTK).
- A key idea in these proofs is that if ϵ is the target accuracy, then at a fixed depth, one can consider the regime of widths scaling as $\mathrm{poly}\left(\frac{1}{\epsilon}\right)$ (best case known is a $\mathrm{polylog}\left(\frac{1}{\epsilon}\right)$ one by Jie-Telgarsky) – and then the predictor becomes effectively linear in the weights.
- Clearly this width regime is unrealistically large!

Also, one can do careful studies of how the individual data used in any SGD update affect predictions at other data – and see that this has very different dynamics between real nets and this limit.

See our work, on this "local elasticity" perspective of deep-learning, **arXiv:2111.01166**

Prominent Approaches To Provable Neural Training

- Another recent line of research for analyzing SGD for neural nets is the mean-field approach. Here the key idea is to formulate the study of neural weight dynamics during training as a study of the dynamics of probability distributions on the neural net weight space.
- This replaces a non-convex optimization problem in finite dimensions by a convex optimization problem in infinite dimensions. Clearly such analysis is meaningful only for infinitely wide nets.

Provable Multigate Neural Training

- **2015** – first convergence on multi-gate nets – but for realizable data and with restrictive assumptions about knowing the score function (Janzamin et al. 2015) – [Improved in Awasthi et al. 2021]
- **2016** – Our first deterministic, any width any data provable exact training, Arora et al. 2016.
- **2017** – Zhong et al. 2017 on GD for depth 2 nets with realizable data [Improved in Zhang et al. 2019]
- **2018** – Lazy differential programming (Chizat et al. 2018), First NTK result (Jacot et al. 2018)
- **2019** – multi depth NTK (Allen-Zhu et al. 2019)
- **2021** – ResNet convergence through mean–field (Fang et al. 2021)

In summary, it seems to have remained an unresolved challenge to show convergence of SGD on any neural architecture with a constant number of gates while neither constraining the labels nor the marginal distributional of the data to a specific functional form.

Our Story Begins with arXiv: 2004.06977

(Bin Shi, Weijie Su & Michael Jordan)

- Shi et al. 2020 analyse the effect of learning rate on SGD by studying the corresponding continuous time limit/Stochastic Differential Equation (SDE)
- Formally, for an objective function $\tilde{L}(\mathrm{W})$ where W denotes the weight matrix, the minibatch SGD update at the k-th step is given by

$$\mathrm{W}_{k+1} = \mathrm{W}_k - \frac{s}{B}\sum_{i=1}^{B}\nabla\tilde{L}_i(\mathrm{W}_k)$$

where the indices $i = 1, 2, \ldots, B$ are the random minibatch

Summary of arXiv: 2004.06977:

(Bin Shi, Weijie Su & Michael Jordan)

- Given the above step-size s SGD update, the authors proved its iterates to be close to the following SDE,

$$d\mathrm{W}_s(t) = -\nabla_{\mathrm{W}_s}\tilde{L}(\mathrm{W}_s(t))dt + \sqrt{s}\,d\mathrm{B}(t)$$

(where $B(t)$ is the standard Brownian motion)

- Then the density of W_s evolves according to the following Fokker-Plank-Smoluchowski PDE

$$\frac{\partial \rho_s}{\partial t} = \nabla \cdot (\rho_s \nabla \tilde{L}) + \frac{s}{2}\Delta \rho_s$$

Summary of arXiv: 2004.06977

(Bin Shi, Weijie Su & Michael Jordan)

- They showed that when $\tilde{L}$, is a "Villani Function", solution to the above F.P.S. PDE converges to the following Gibbs measure - with a provable convergence rate.

$$\mu_s(\mathrm{W}_s) = \frac{1}{Z_s} \exp\left(\frac{-2\tilde{L}(\mathrm{W}_s)}{s}\right)$$

where, Z_s is the normalization constant.

- The idea of Villani functions possibly first occurred/were explained in Cedric Villani's 2009 monograph, "Hypocoercivity".

Summary of arXiv: 2004.06977:

(Bin Shi, Weijie Su & Michael Jordan,)

Definition (**Villani Functions**)

A map $f : \mathbb{R}^d \to \mathbb{R}$ is called a Villani function, if it satisfies the following conditions,

1. $f \in C^\infty$
2. $\lim_{||\mathbf{x}||\to\infty} f(\mathbf{x}) = +\infty$
3. $\int_{\mathbb{R}^d} \exp\left(-\frac{2f(\mathbf{x})}{s}\right) d\mathbf{x} < \infty \ \ \forall s > 0$
4. $\lim_{||\mathbf{x}||\to\infty} \left(-\Delta f(\mathbf{x}) + \frac{1}{s} \cdot ||\nabla f(\mathbf{x})||^2\right) = +\infty \ \ \forall s > 0$

Further, any f that satisfies conditions $1, 2$ & 3 is said to be "confining".

Summary of arXiv: 2004.06977

(Bin Shi, Weijie Su & Michael Jordan)

Key to their argument for getting the non-asymptotic convergence is the fact that the Gibbs' measure corresponding to a Villani function satisfies the following Poincaré–type inequality,

Theorem (Shi et al. 2020)

For a Villani function f, for any given $s > 0$, define Gibbs' density, $\mu_s(\mathbf{x}) = \frac{1}{Z_s}\exp\left(-\frac{2f(\mathbf{x})}{s}\right)$, where Z_s is a normalization factor. Then μ_s satisfies a Poincare – type inequality i.e $\exists\ \lambda_s > 0$ (determined by f) s.t $\forall h \in C_c^\infty(\mathbb{R}^d)$ we have,

$$\mathrm{Var}_{\mu_s}[h] \leq \frac{s}{2 \cdot \lambda_s}\,\mathbb{E}_{\mu_s}[\|\nabla h\|^2]$$

Can Neural Nets induce Villani functions?

In their paper, Bin Shi, Weijie Su & Michael Jordan had commented,

"some loss functions used for training neural networks might not satisfy this condition." (page 9)

Lets See!

Can Neural Nets induce Villani functions?

Consider The Following Setup With Neural Nets!

- Define a 2 layer neural net $f(\mathbf{x};\mathbf{a},\mathrm{W}):\mathbb{R}^d\to\mathbb{R}$ with trainable weights $\mathrm{W}\in\mathbb{R}^{p\times d}$, fixed weights $\mathbf{a}\in\mathbb{R}^p$, activation functions $\sigma(\cdot)$ and data $\mathbf{x}\in\mathbb{R}^d$ as

$$f(\mathbf{x};\mathbf{a},\mathrm{W})=\mathbf{a}^\top\sigma(\mathrm{W}\mathbf{x}).$$

- For n data tuples $\{(\mathbf{x}_i,y_i)\in\mathbb{R}^d\times\mathbb{R}\}_{i=1}^n$, $\lambda\in\mathbb{R}_{\geq 0}$, the Frobenius norm regularized loss we consider is,

$$\tilde{L}(\mathrm{W})=\frac{1}{n}\sum_{i=1}^{n}(y_i-f(\mathbf{x}_i;\mathbf{a},\mathrm{W}))^2+\frac{\lambda}{2}\|\mathrm{W}\|_F^2$$

Neural Nets & Villani Functions

Consider The Following Setup With Neural Nets!

- Let the activation $\sigma(\cdot)$ be bounded, C^∞, L–Lipschitz and L'_σ–smooth. Further, the first and the second derivatives of σ also be bounded. Eg. tanh & sigmoid
- We can relax the boundedness assumption, at the cost of foregoing discrete time convergence.

Our Primary Result in arXiv:2210.11452

Theorem (SGD Converges To The Global Minima Of Finite Width Nets)

For the activations considered, the loss $\tilde{L}$ *is* gLip*–gradient Lipschitz &* $\exists \lambda_c > 0$ *s.t* $\forall \lambda > \lambda_c$ *the loss function is a Villani function and,*

- $\forall\ T, \epsilon > 0$, $\exists$ *constants* $A(\tilde{L})$, $B(T, \tilde{L})$ *and* $C(s, \tilde{L})$ *s.t with SGD at constant stepsize* $s = s^* := \min\left(\frac{1}{\mathrm{gLip}(\tilde{L})}, \frac{\epsilon}{2\cdot(A(\tilde{L})+B(T,\tilde{L}))}\right)$
- *And,* $\mathbf{W}^0$ *initialized from* $\rho_{\text{initial}} \in L^2(\frac{1}{\mu_{s^*}})$ *where* $\mu_{s^*} = \frac{1}{Z_{s^*}} \exp\left(-\frac{2\tilde{L}(\mathbf{W})}{s^*}\right)$ *(Z_{s^*} being the normalization factor) and for initialization s.t* $2C(s^*, \tilde{L}) \cdot \|\rho_{\text{initial}} - \mu_{s^*}\|_{\mu_{s^*}^{-1}} \leq \epsilon \cdot e^{\lambda_{s^*} \cdot T}$,

we get,

$$\mathbb{E}\tilde{L}(\mathbf{W}^{\frac{T}{s^*}}) - \inf_{\mathbf{W}} \tilde{L}(\mathbf{W}) \leq \epsilon$$

Remarks & Proof Sketch

- For intuition about the value of the regularizer, consider sigmoid activations and suppose we normalize the data (of max norm B_x) and the outer layer weights ($\mathbf{a}$) s.t $B_x \cdot \|\mathbf{a}\| = 1$, then,

$$\lambda_c = 0.125$$

- This critical value of the regularizer comes from the following condition of being a Villani function,

$$\lim_{\|\mathbf{x}\| \to \infty} \left(-\Delta f(\mathbf{x}) + \frac{1}{s} \cdot \|\nabla f(\mathbf{x})\|^2 \right) = +\infty \ \ \forall s > 0$$

Remarks & Proof Sketch

- The gradient Lipschitz condition on the loss function is required by the standard theorems for converting the continuous-time convergence achieved through Villani functions to a discrete-time convergence result.
- The loss function $\tilde{L}$ for SoftPlus (smooth version of ReLU) does not satisfy the smoothness condition - but the SDE shown earlier still provably converges to the global minima with SoftPlus.

Our Result in arXiv:2210.11452 About SoftPlus Nets

- The $\mathrm{SoftPlus}$ activation function is defined as,

$$\mathrm{SoftPlus}_\beta(x) = \frac{1}{\beta}\log(1+\exp(\beta x))$$

- For any small enough step-size s and $\lambda > \lambda_c := 2$ [1] and for $t \geq \frac{1}{\lambda_s}\log\frac{2\,C(s,\tilde{L})\|\rho_{\text{initial}}-\mu_s\|_{\mu_s^{-1}}}{\epsilon}$, we have that,

$$\mathbb{E}\,\tilde{L}(\mathrm{W}(t)) - \min_{\mathrm{W}}\tilde{L}(\mathrm{W}) \leq \epsilon.$$

(This is linear time convergence)

[1] for data and last layer norm having been been normalized to have product be 1

SGD Performance Remains Sensitive To Data Corruption for $\lambda > \lambda_c$

- SGD experiments on depth 2 sigmoid neural nets confirm that the regularizer coefficient $(\frac{1}{8})$ is not too large to 'overshadow' the empirical loss - we check this via an ablation study on errors w.r.t. adding noise into the true labels.

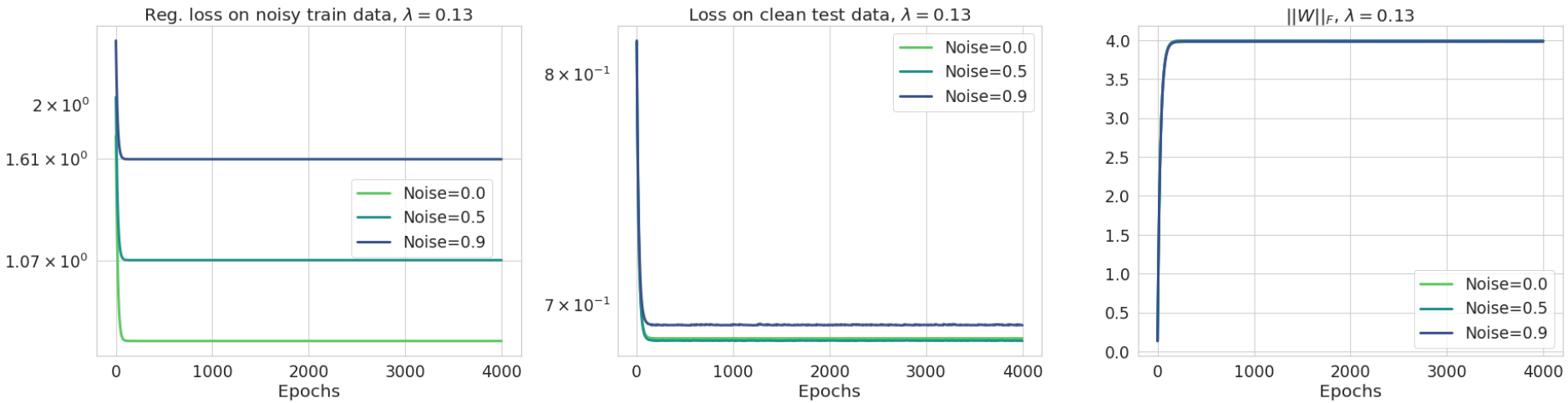


Figure: $\lambda = 0.13$, width $= 10$

A View of What is Known & What Probably Isn't

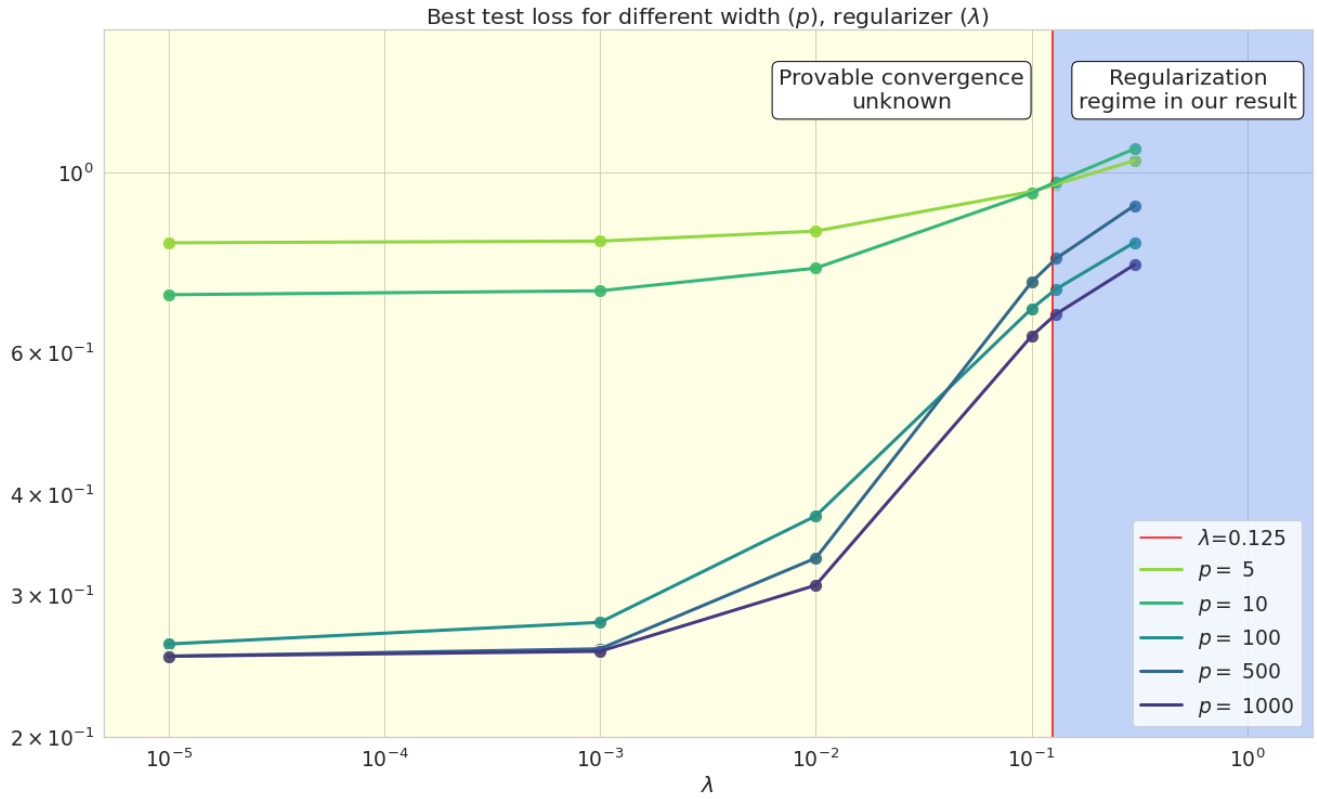


Figure: Best test loss across a range of $(\lambda, p = \text{width})$ for data labelled as,

$$\mathbf{x} \sim \mathcal{U}[0,1)^d, y = \sin\left(\pi \frac{\|\mathbf{x}\|_2^2}{d}\right) + \epsilon;\ \epsilon \sim \mathcal{N}(0, 0.25), d = 20$$

A View of What is Known & What Probably Isn't

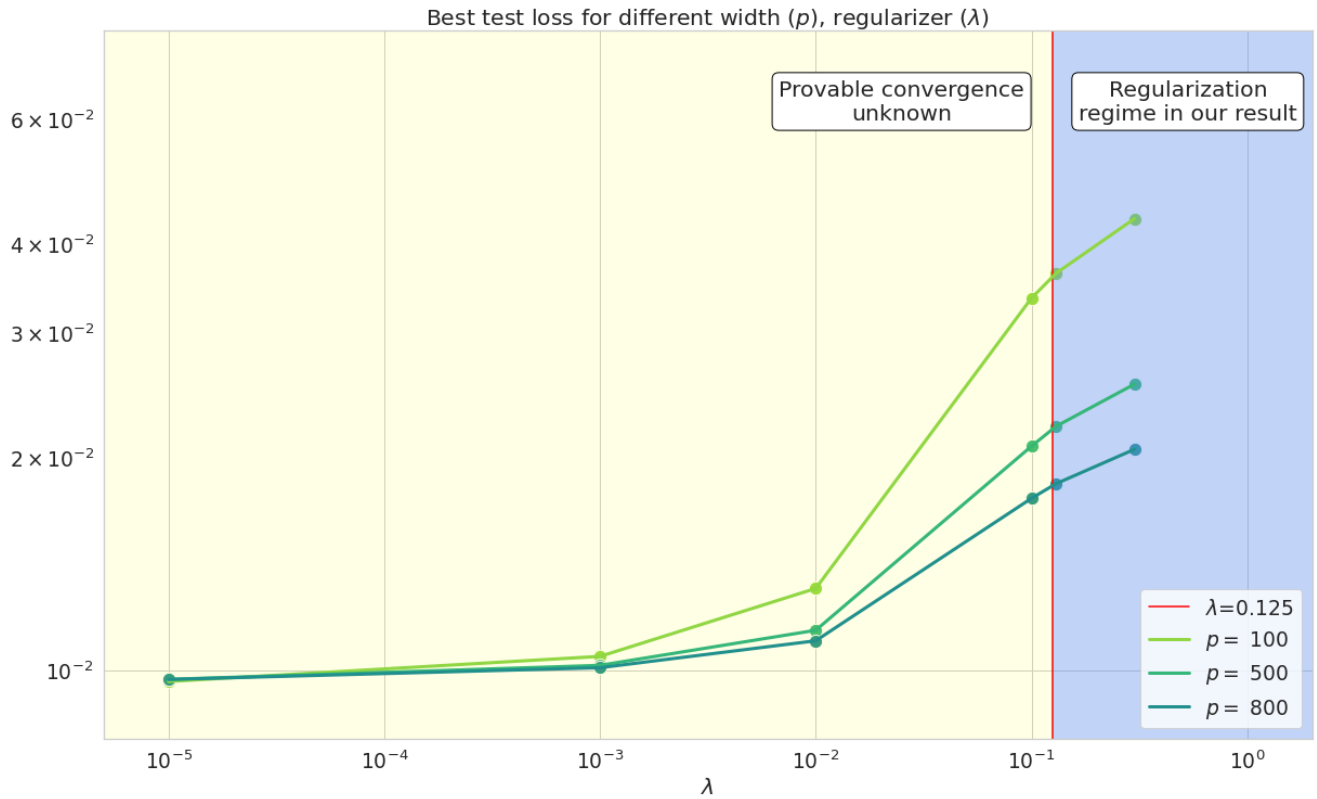


Figure: Best test loss across a range of $(\lambda, \text{width} = p)$ for realizable data that is,

$$\mathbf{x} \sim \mathcal{U}[0,1)^{20}, y = \mathbf{a}^\top \sigma(\mathbf{W}\mathbf{x}) + \epsilon;\ p = 5, \epsilon \sim \mathcal{N}(0, 10^{-2}), a_i \sim \frac{1}{\sqrt{p}}\mathcal{N}(0,1)$$

The Journey Begins!

To best of our knowledge, this is the first result showing global convergence of SGD on neural nets, *without any assumptions on data or the number of gates.* Naturally, we can now ask :
Could this Villani function idea be a general method to prove SGD convergence on various other neural loss functions?

Can we characterize λ_s for the Gibbs' measure of neural nets?
This becomes a very relevant question now that we can establish that for certain neural losses Poincare constant of the Gibbs' measure is the time-scale of SGD convergence.

The Journey Begins!

In **arXiv:2205.11359** (with Sayar & Pulkit) we have given the first width-independent Rademacher complexity bounds for DeepONets - which is a novel architecture which uses an inner-product of two nets to learn maps between Banach spaces
- thus it can solve families of PDEs in one-shot.

As of now there are no provable training for this. A very interesting next step could be understand training of these wholly new kind of neural architectures using this new framework.

Welcome

To

The Center for A.I Fundamentals

(The University of Manchester)

Questions? 

References I

[1] Zeyuan Allen-Zhu, Yuanzhi Li, and Yingyu Liang. "Learning and generalization in overparameterized neural networks, going beyond two layers". In: *Advances in neural information processing systems*. 2019, pp. 6155–6166.

[2] Raman Arora, Amitabh Basu, Poorya Mianjy, and Anirbit Mukherjee. "Understanding Deep Neural Networks with Rectified Linear Units". In: (2016). DOI: 10.48550/ARXIV.1611.01491. URL: https://arxiv.org/abs/1611.01491.

[3] Pranjal Awasthi, Alex Tang, and Aravindan Vijayaraghavan. "Efficient Algorithms for Learning Depth-2 Neural Networks with General ReLU Activations". In: *Advances in Neural Information Processing Systems*. Ed. by M. Ranzato, A. Beygelzimer, Y. Dauphin, P.S. Liang, and J. Wortman Vaughan. Vol. 34. Curran Associates, Inc., 2021, pp. 13485–13496. URL: https://proceedings.neurips.cc/paper/2021/file/700fdb2ba62d4554dc268c65add4b16e-Paper.pdf.

[4] Lenaic Chizat, Edouard Oyallon, and Francis Bach. "On Lazy Training in Differentiable Programming". In: (2018). DOI: 10.48550/ARXIV.1812.07956. URL: https://arxiv.org/abs/1812.07956.

References II

[5] Cong Fang, Jason Lee, Pengkun Yang, and Tong Zhang. "Modeling from Features: a Mean-field Framework for Over-parameterized Deep Neural Networks". In: *Proceedings of Thirty Fourth Conference on Learning Theory*. 2021, pp. 1887–1936. URL: https://proceedings.mlr.press/v134/fang21a.html.

[6] Arthur Jacot, Franck Gabriel, and Clément Hongler. "Neural tangent kernel: Convergence and generalization in neural networks". In: *Advances in neural information processing systems*. 2018, pp. 8571–8580.

[7] Majid Janzamin, Hanie Sedghi, and Anima Anandkumar. *Beating the Perils of Non-Convexity: Guaranteed Training of Neural Networks using Tensor Methods*. 2015. DOI: 10.48550/ARXIV.1506.08473. URL: https://arxiv.org/abs/1506.08473.

[8] Bin Shi, Weijie J. Su, and Michael I. Jordan. *On Learning Rates and Schrödinger Operators*. 2020. DOI: 10.48550/ARXIV.2004.06977. URL: https://arxiv.org/abs/2004.06977.

References III

[9] Xiao Zhang, Yaodong Yu, Lingxiao Wang, and Quanquan Gu. "Learning One-hidden-layer ReLU Networks via Gradient Descent". In: *Proceedings of Machine Learning Research*. Ed. by Kamalika Chaudhuri and Masashi Sugiyama. Vol. 89. Proceedings of Machine Learning Research. PMLR, 2019, pp. 1524–1534. URL: http://proceedings.mlr.press/v89/zhang19g.html.

[10] Kai Zhong, Zhao Song, Prateek Jain, Peter L. Bartlett, and Inderjit S. Dhillon. *Recovery Guarantees for One-hidden-layer Neural Networks*. 2017. DOI: 10.48550/ARXIV.1706.03175. URL: https://arxiv.org/abs/1706.03175.