Early Neuron Alignment in Two-layer ReLU Networks with Small Initialization

Hancheng Min
Enrique Mallada
René Vidal

Nov 16th-17th
DeepMath 2023

Penn Engineering
UNIVERSITY of PENNSYLVANIA

Mysteries of deep learning

Why do simple neural network training algorithms (gradient flow/descent) find a global minimum of a non-convex loss function?

Why gradient flow/descent finds global minimum (among many) that generalizes well?

From NTK regime to feature learning regime

- Neural Tangent Kernel (NTK) Regime [Jacot’18, Arora’19]: Extremely wide hidden layer, large initialization
  - Exponential convergence toward the global minimum
  - ≈“Kernel regression” with fixed kernel: Prevent feature learning

- From large to small init. scale: Kernel regime to rich regime
  - Implicit bias in L-layer diagonal linear networks [Woodworth’20]: l_2 regularization $\longrightarrow$ (decreasing init. scale) $\longrightarrow l_{2/L}$ regularization

- Inductive bias of small initialization
  - Diagonal linear networks [Woodworth’20, Vaskevicius’19]: Sparsity
  - Matrix factorization [Soltanolkotabi’21, Li’21]: Low-rankness
  - **This work**: **Two-layer ReLU networks**

Penn Engineering
UNIVERSITY of PENNSYLVANIA

Two-layer ReLU nets under small init.: Prior work

	Assumption on Training data	***Quantitative Analysis?***	***Requirement on hidden-layer width***
[Phuong'21]	μ-orthogonally separable + # of data ≥ dim of data	No	**Ω(1)**
[Boursier'22]	Mutually orthogonal data (# of data ≤ dim of data)	**Yes**	Ω(exp((# of data))
Our work	**$\mu-$orthogonally separable**	**Yes**	**Ω(1)**

Phuong, M. and Lampert, H. C. The inductive bias of relu networks on orthogonally separable data. ICLR 2021

Boursier, E., Pillaud-Vivien, L., and Flammarion, N. Gradient flow dynamics of shallow relu networks for square loss and orthogonal inputs. NeurIPS, 2022

Min, H., Mallda, E., and Vidal, R., Early Neuron Alignment in Two-layer ReLU Networks with Small Initialization. arXiv 2307.12851. 2023

Outline

- Motivation and prior work
- Problem setting and illustrative example
- Two-stage training with small initialization
- Neuron dynamics in alignment phase
- Conclusion

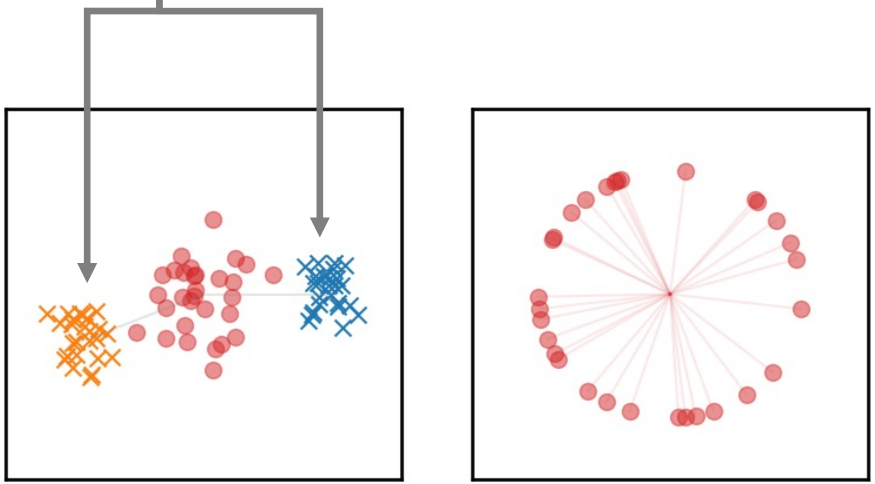
Penn Engineering
UNIVERSITY of PENNSYLVANIA

Problem setting

- (*Data*) Input: $x_i \in \mathbb{R}^D$ Label: $y_i \in \{+1, -1\}$

Negative data
$y_i = -1$

Positive data
$y_i = +1$



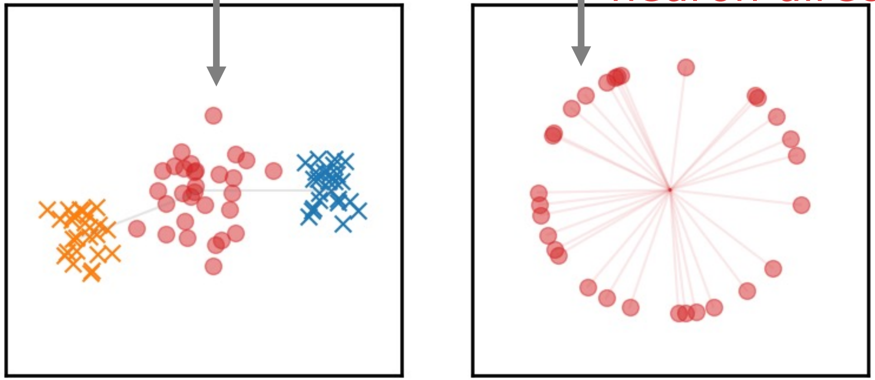
Toy example in $\mathbb{R}^2$

Problem setting

- *(Data)* Input: $x_i \in \mathbb{R}^D$ Label: $y_i \in \{+1, -1\}$
- *(ReLU Network)* $\mathrm{NN}\left(x; \{w_j, v_j\}_{j=1}^{h}\right) = \sum_{j=1}^{h} v_j \sigma(\langle x, w_j \rangle)$, $\sigma(u) = \max\{u, 0\}$

Neurons: $w_j \in \mathbb{R}^D$

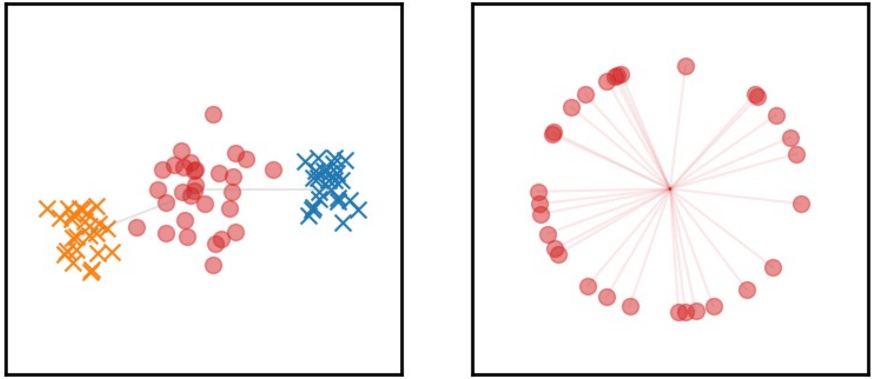
$\frac{w_j}{\|w_j\|}$: normalized neuron directions



Toy example in $\mathbb{R}^2$

Problem setting

- (*Data*) Input: $x_i \in \mathbb{R}^D$ Label: $y_i \in \{+1, -1\}$
- (*ReLU Network*) $\mathrm{NN}\left(x; \{w_j, v_j\}_{j=1}^h\right) = \sum_{j=1}^h v_j \sigma(\langle x, w_j \rangle)$, $\sigma(u) = \max\{u, 0\}$
- (*Exponential Loss*) $\mathcal{L}\left(\{w_j, v_j\}_{j=1}^h\right) = \sum_{i=1}^n \exp\left(-y_i \cdot \mathrm{NN}\left(x_i; \{w_j, v_j\}_{j=1}^h\right)\right)$

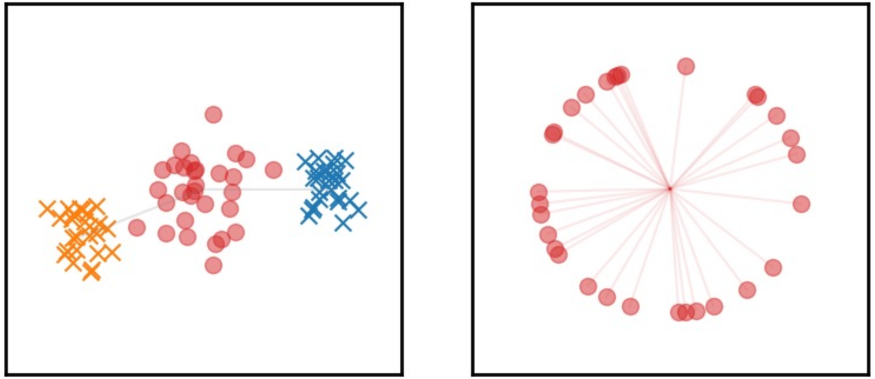


Toy example in $\mathbb{R}^2$

Problem setting

- (*Initialization*) $w_j(0) \sim \mathcal{N}(0, \epsilon^2 I)$ $v_j(0) \sim \mathcal{N}(0, \epsilon^2)$
- (*Training*) **gradient flow** under **small** init. scale ϵ

$$\frac{d}{dt} w_j = -\nabla_{w_j} \mathcal{L}, \qquad \frac{d}{dt} v_j = -\nabla_{v_j} \mathcal{L}$$



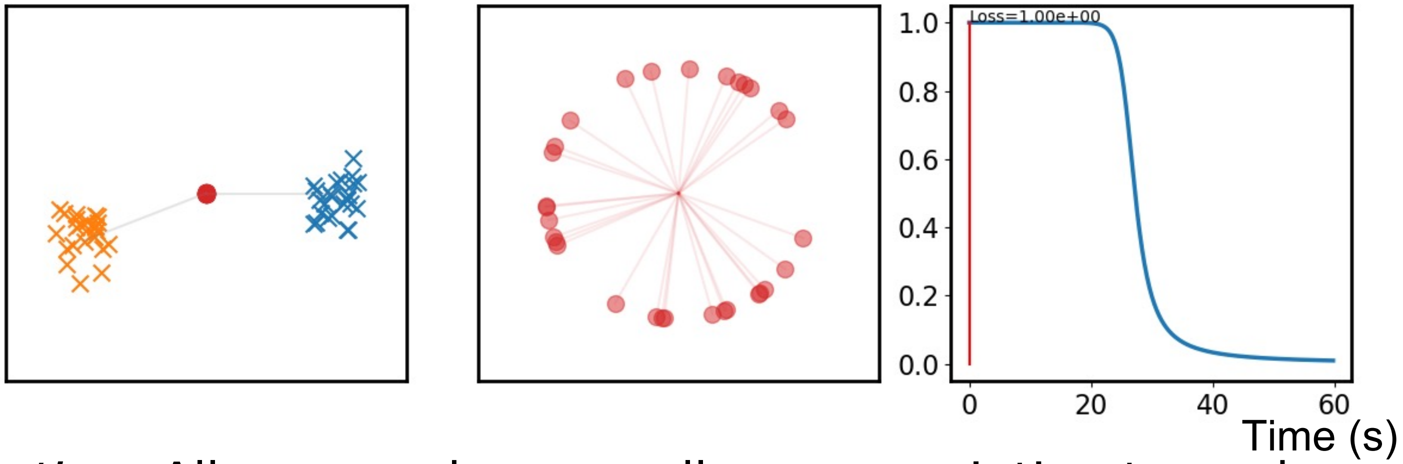
Toy example in $\mathbb{R}^2$

Training under small initialization

Small init. scale
$\epsilon = \mathbf{1{\times}10^{-6}}$

x: Positive data
x: Negative data
o: Neurons

Data/neuron positions | *Neuron directions* | *Training Loss*



- *At initialization*: All neurons have small norms, pointing toward random directions

Training under small initialization

Small init. scale

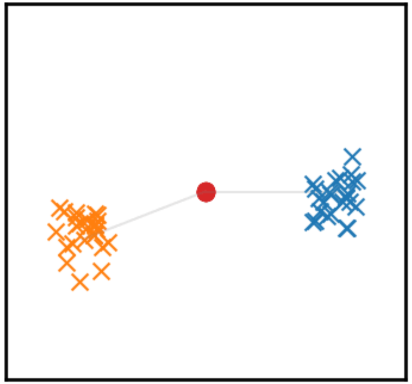
$\boldsymbol{\epsilon = 1\times10^{-6}}$

x: Positive data

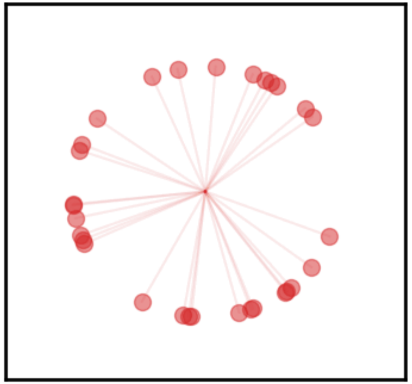
x: Negative data

o: Neurons

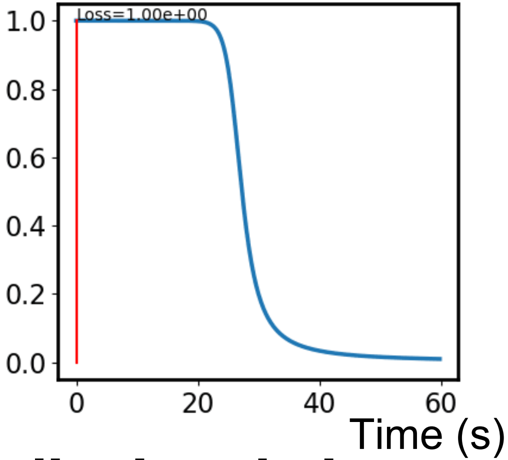
Data/neuron positions



Neuron directions



Training Loss



- ***First stage***: Neurons keep small norms while **aligning their directions** with input data; No significant decrease in loss

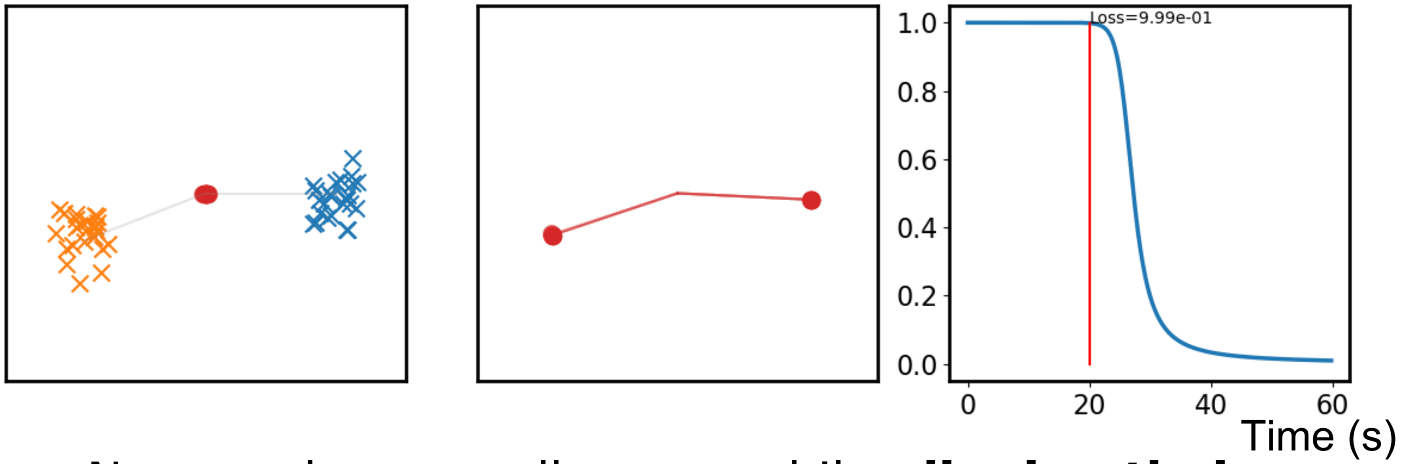
Penn Engineering
University of Pennsylvania

Training under small initialization

Small init. scale
$\epsilon = 1{\times}10^{-6}$

x: Positive data
x: Negative data
o: Neurons

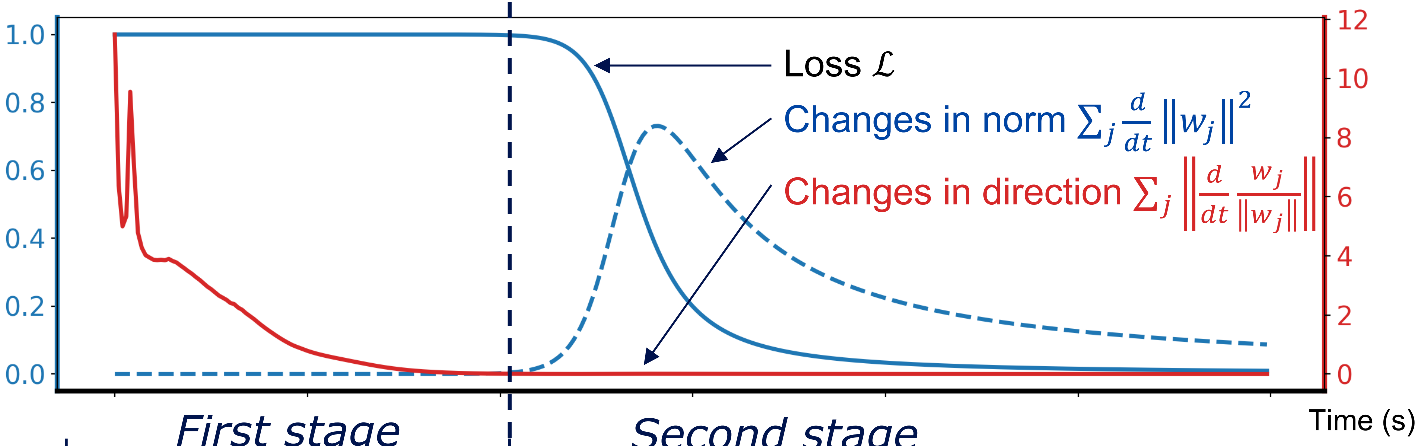
Data/neuron positions *Neuron directions* *Training Loss*



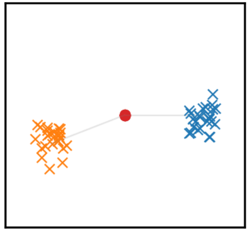
- ***First stage***: Neurons keep small norms while **aligning their directions** with input data; No significant decrease in loss
- ***Second stage***: Neurons **grow their norms**, and the loss decreases quickly

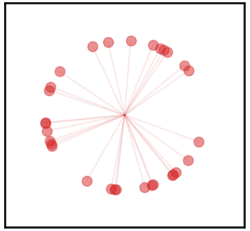
Penn Engineering
University of Pennsylvania

Two-stage training



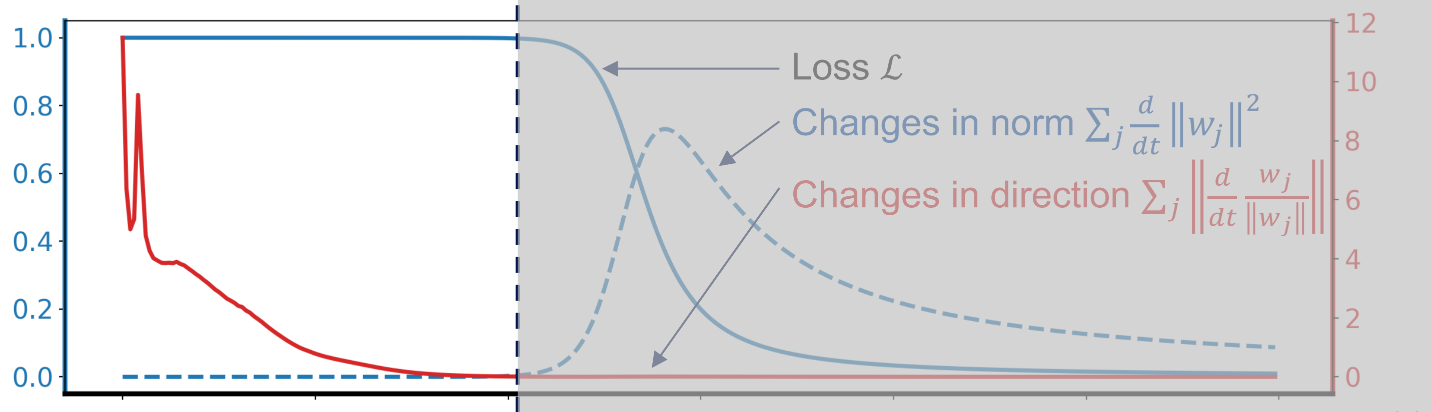
Neurons $w_j, j = 1, \cdots, h$	*First stage* ***Alignment Phase***	*Second stage* ***Final Convergence***
Changes in **norm**	**Small**	**Large** until loss is small
Changes in **direction**	**Large** until "good alignment"	**Small**



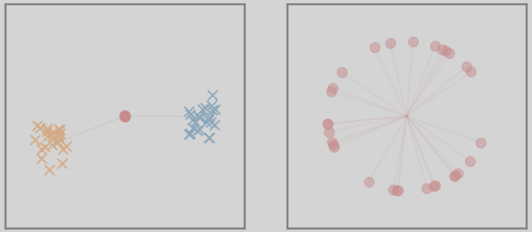


Penn Engineering
UNIVERSITY of PENNSYLVANIA

Two-stage training



Neurons $w_j, j = 1, \cdots, h$	*First stage* ***Alignment Phase***	*Second stage* ***Final Convergence***
Changes in **norm**	**Small**	**Large** until loss is small
Changes in **direction**	**Large** until "good alignment"	**Small**



Outline

- Motivation and prior work
- Problem setting and illustrative example
- Two-stage training with small initialization
- Neuron dynamics in alignment phase
- Conclusion

Decompose neuron dynamics in alignment phase

Gradient Flow dynamics

$$\frac{d}{dt} w_j = -\nabla_{w_j} \mathcal{L}$$

$$\frac{d}{dt} v_j = -\nabla_{v_j} \mathcal{L}$$

???

Neuron norm dynamics

$$\frac{d}{dt} \|w_j\|^2$$

Neuron angular dynamics

$$\frac{d}{dt} \frac{w_j}{\|w_j\|}$$

Decompose neuron dynamics in alignment phase

Gradient Flow dynamics

$$\frac{d}{dt} w_j = -\nabla_{w_j} \mathcal{L}$$

$$\frac{d}{dt} v_j = -\nabla_{v_j} \mathcal{L}$$

ϵ-small init. scale

Technical assumption: balanced weights

Decoupled Neuron dynamics

$$\frac{d}{dt} w_j \approx \operatorname{sign}\left(v_j(0)\right) \sum_{i:\langle x_i, w_j\rangle > 0} x_i\, y_i \|w_j\|$$

Neuron norm dynamics

$$\frac{d}{dt} \|w_j\|^2$$

Neuron angular dynamics

$$\frac{d}{dt} \frac{w_j}{\|w_j\|}$$

Decompose neuron dynamics in alignment phase

Gradient Flow dynamics

$$\frac{d}{dt} w_j = -\nabla_{w_j} \mathcal{L}$$

$$\frac{d}{dt} v_j = -\nabla_{v_j} \mathcal{L}$$

ϵ-small init. scale

Technical assumption: balanced weights

Decoupled Neuron dynamics

$$\frac{d}{dt} w_j \approx \sum_{i:\langle x_i, w_j \rangle > 0} x_i \, y_i \|w_j\|$$

Neuron norm dynamics

$$\frac{d}{dt} \|w_j\|^2$$

Neuron angular dynamics

$$\frac{d}{dt} \frac{w_j}{\|w_j\|}$$

Penn Engineering
University *of* Pennsylvania

Decompose neuron dynamics in alignment phase

Gradient Flow dynamics

$$\frac{d}{dt} w_j = -\nabla_{w_j}\mathcal{L}$$
$$\frac{d}{dt} v_j = -\nabla_{v_j}\mathcal{L}$$

ϵ-small init. scale

Technical assumption: balanced weights

Decoupled Neuron dynamics

$$\frac{d}{dt} w_j \approx \sum_{i:\langle x_i, w_j\rangle > 0} x_i\, y_i \|w_j\|$$

$$\frac{d}{dt}\|w_j\|^2 = 2\left\langle \frac{d}{dt} w_j, w_j \right\rangle$$

Neuron norm dynamics

$$\frac{d}{dt}\|w_j\|^2 \approx 2 \sum_{i:\langle x_i, w_j\rangle > 0} \langle x_i, w_j\rangle y_i \|w_j\|$$

Neuron angular dynamics

$$\frac{d}{dt}\frac{w_j}{\|w_j\|}$$

Decompose neuron dynamics in alignment phase

Gradient Flow dynamics

$$\frac{d}{dt} w_j = -\nabla_{w_j} \mathcal{L}$$

$$\frac{d}{dt} v_j = -\nabla_{v_j} \mathcal{L}$$

ϵ-small init. scale

Technical assumption: balanced weights

Decoupled Neuron dynamics

$$\frac{d}{dt} w_j \approx \sum_{i: \langle x_i, w_j \rangle > 0} x_i \, y_i \|w_j\|$$

$$\frac{d}{dt}\|w_j\|^2 = 2\left\langle \frac{d}{dt} w_j, w_j \right\rangle$$

Neuron norm dynamics

$$\frac{d}{dt}\|w_j\|^2 \lesssim C\|w_j\|^2$$

Neuron angular dynamics

$$\frac{d}{dt}\frac{w_j}{\|w_j\|}$$

Penn Engineering
University of Pennsylvania

Decompose neuron dynamics in alignment phase

Gradient Flow dynamics

$$\frac{d}{dt}w_j = -\nabla_{w_j}\mathcal{L}$$

$$\frac{d}{dt}v_j = -\nabla_{v_j}\mathcal{L}$$

ϵ-small init. scale

Technical assumption: balanced weights

Decoupled Neuron dynamics

$$\frac{d}{dt}w_j \approx \sum_{i:\langle x_i, w_j\rangle>0} x_i\, y_i \|w_j\|$$

$$\frac{d}{dt}\frac{w_j}{\|w_j\|} = \mathcal{P}^{\perp}_{w_j}\left(\frac{1}{\|w_j\|}\frac{d}{dt}w_j\right)$$

Neuron norm dynamics

$$\frac{d}{dt}\|w_j\|^2 \lesssim C\|w_j\|^2$$

Neuron angular dynamics

$$\frac{d}{dt}\frac{w_j}{\|w_j\|} = \mathcal{P}^{\perp}_{w_j}\left(\sum_{i:\langle x_i, w_j\rangle>0} x_i\, y_i\right)$$

Neuron dynamics in alignment phase

- *Neuron norms* are small at initialization, and so are derivatives. But they can only be small for a certain amount of time $\Theta\left(\log\frac{1}{\epsilon}\right)$

Neurons $w_j, j = 1, \cdots, h$	***Alignment Phase***
Changes in **norm**	**Small**
Changes in **direction**	**Large** until "good alignment"

Neuron norm dynamics

$$\frac{d}{dt}\|w_j\|^2 \lesssim C\|w_j\|^2$$

Neuron angular dynamics

$$\frac{d}{dt}\frac{w_j}{\|w_j\|} = \mathcal{P}^{\perp}_{w_j}\left(\sum_{i:\langle x_i, w_j\rangle>0} x_i\, y_i\right)$$

Neuron dynamics in alignment phase

- *Neuron norms* are small at initialization, and so are derivatives. But they can only be small for a certain amount of time $\Theta\left(\log\frac{1}{\epsilon}\right)$
- *Neurons* move their directions towards a **centroid**

$$x_a(w_j) = \sum_{i:\langle x_i, w_j\rangle>0} x_i\, y_i$$

Neurons $w_j, j = 1, \cdots, h$	***Alignment Phase***
Changes in **norm**	**Small**
Changes in **direction**	**Large** until "good alignment"

Neuron norm dynamics

$$\frac{d}{dt}\|w_j\|^2 \lesssim C\|w_j\|^2$$

Neuron angular dynamics

$$\frac{d}{dt}\frac{w_j}{\|w_j\|} = \mathcal{P}_{w_j}^{\perp}\left(\sum_{i:\langle x_i, w_j\rangle>0} x_i\, y_i\right)$$

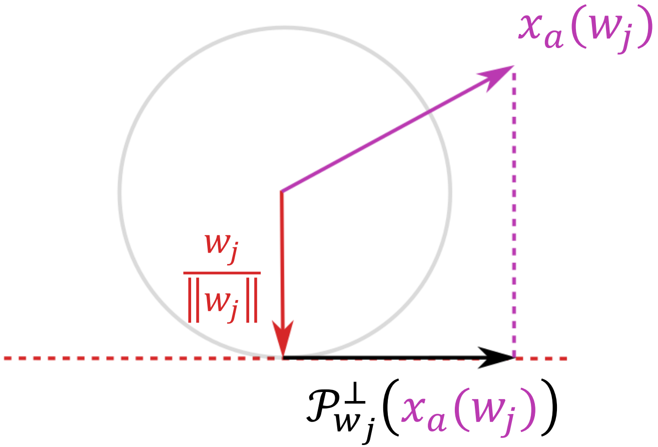
Penn Engineering
University of Pennsylvania

Neuron angular dynamics in alignment phase

Neuron angular dynamics

$$\frac{d}{dt}\frac{w_j}{\|w_j\|} = \mathcal{P}^{\perp}_{w_j}\big(x_a(w_j)\big), \qquad x_a(w_j) = \sum_{i:\langle x_i, w_j\rangle > 0} x_i\, y_i$$

If $x_a(w_j)$ is fixed, then neuron rotates towards $x_a(w_j)$



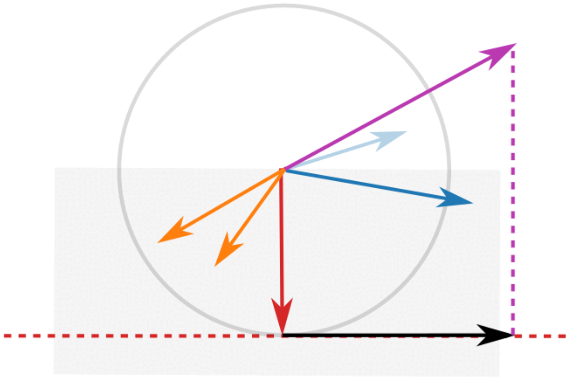
Neuron angular dynamics in alignment phase

Neuron angular dynamics

$$\frac{d}{dt}\frac{w_j}{\|w_j\|} = \mathcal{P}^{\perp}_{w_j}\big(x_a(w_j)\big), \qquad x_a(w_j) = \sum_{i:\langle x_i, w_j\rangle > 0} x_i\, y_i$$

$x_a(w_j)$ depends on direction of w_j, thus it is a **moving target** for the neuron

(Early Alignment)



Neuron angular dynamics in alignment phase

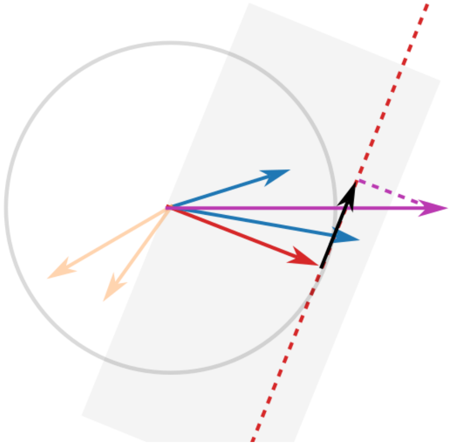
Neuron angular dynamics

$$\frac{d}{dt}\frac{w_j}{\|w_j\|} = \mathcal{P}^{\perp}_{w_j}\left(x_a(w_j)\right), \qquad x_a(w_j) = \sum_{i:\langle x_i, w_j\rangle > 0} x_i\, y_i$$

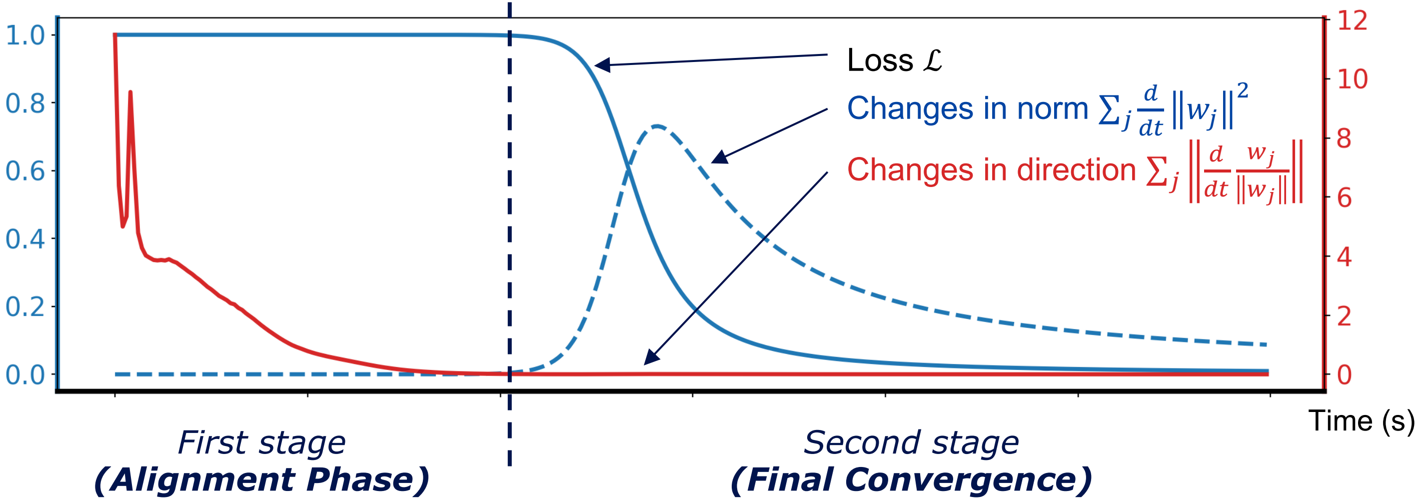
Once the neuron activates *all positive* data and *none of the negative* data, centroid $x_a(w)$ remains **fixed**:

(Positive data center) $x_a(w) = \sum_{i:\, y_i>0} x_i = \boldsymbol{x}_+$

(Refined Alignment)



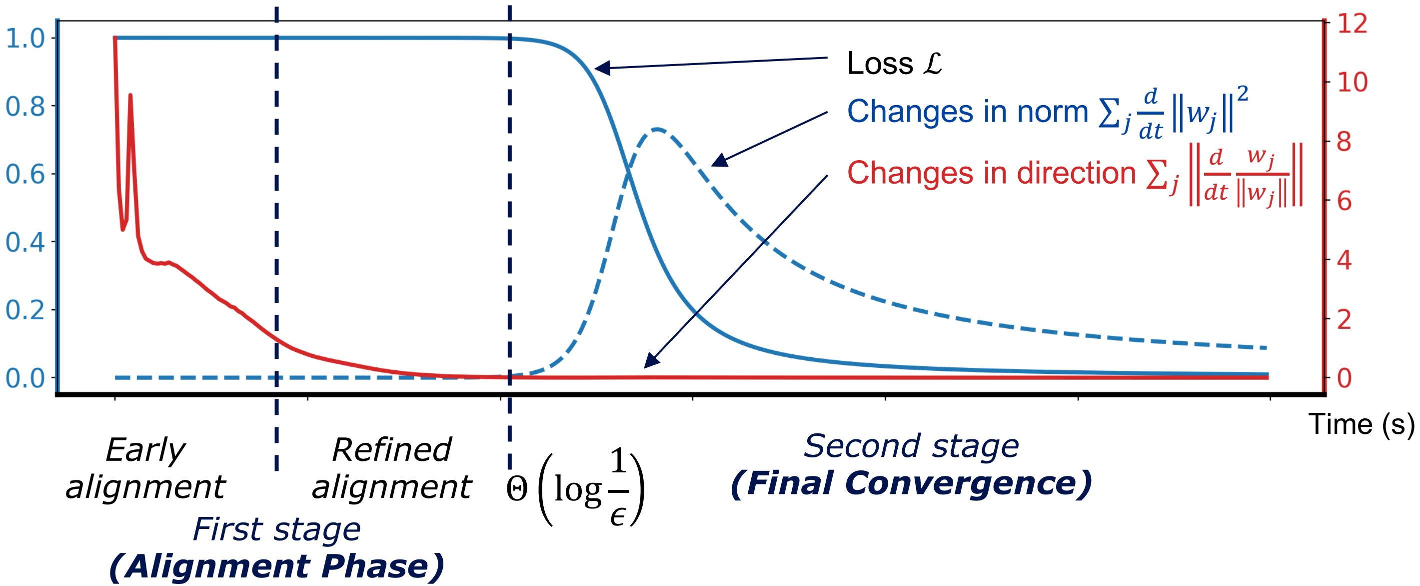
Break down alignment phase



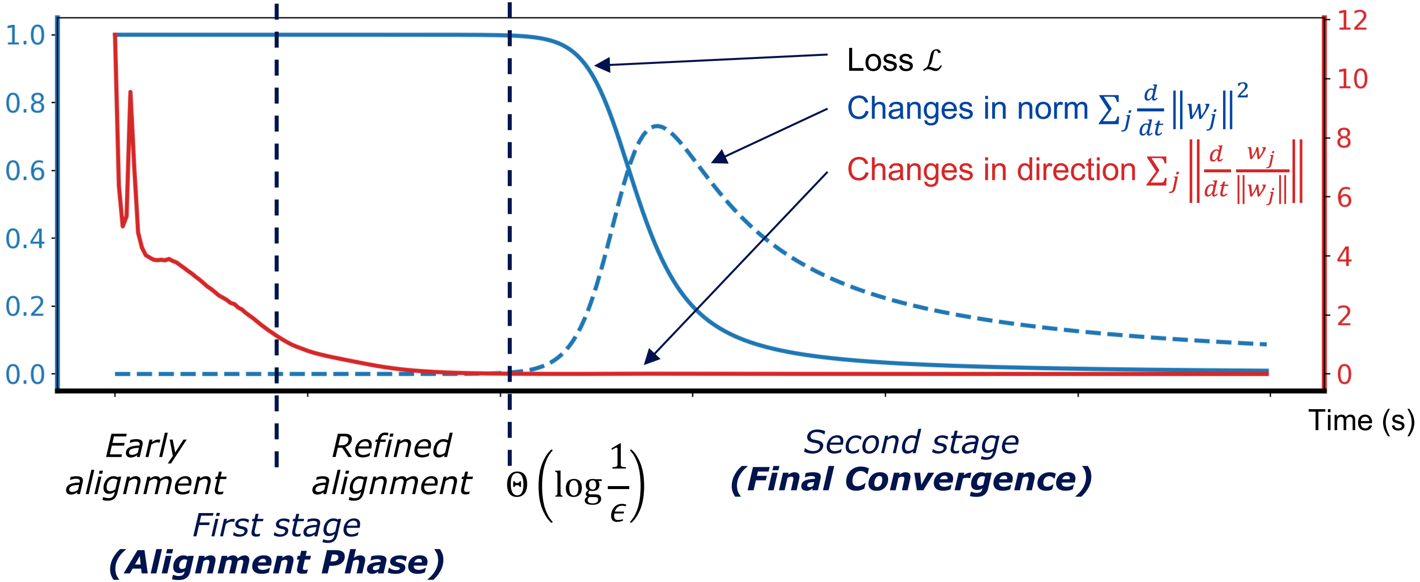
Alignment phase can be further broken down into:

- (*Early alignment*) each neuron w_j chases a moving target $x_a(w_j)$ until **activates all positive data and none of the negative data**
- (*Refined alignment*) each neuron w aligns with the positive data center

Break down alignment phase



Early alignment

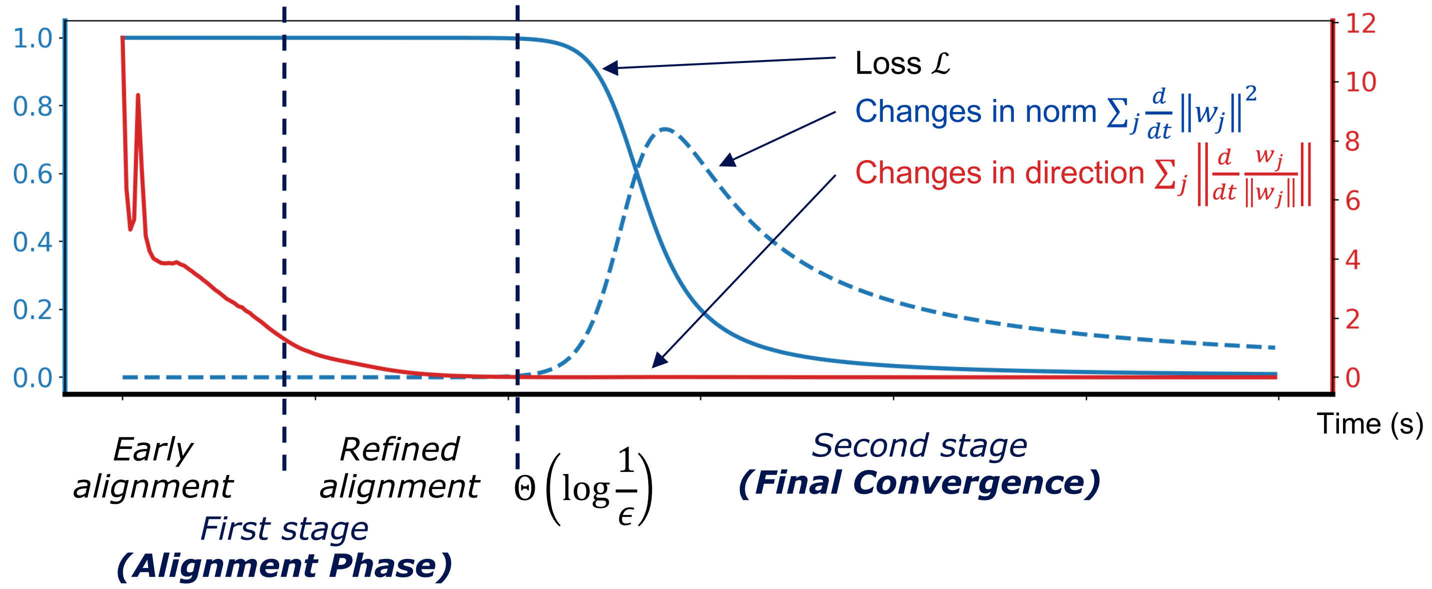


Theorem (Informal) Early alignment lasts **at most** $\mathcal{O}\left(\frac{\log n}{\sqrt{\mu}}\right)$ time

- n: # of data, μ: "data separability"
- Sufficient for final convergence

Penn Engineering
University of Pennsylvania

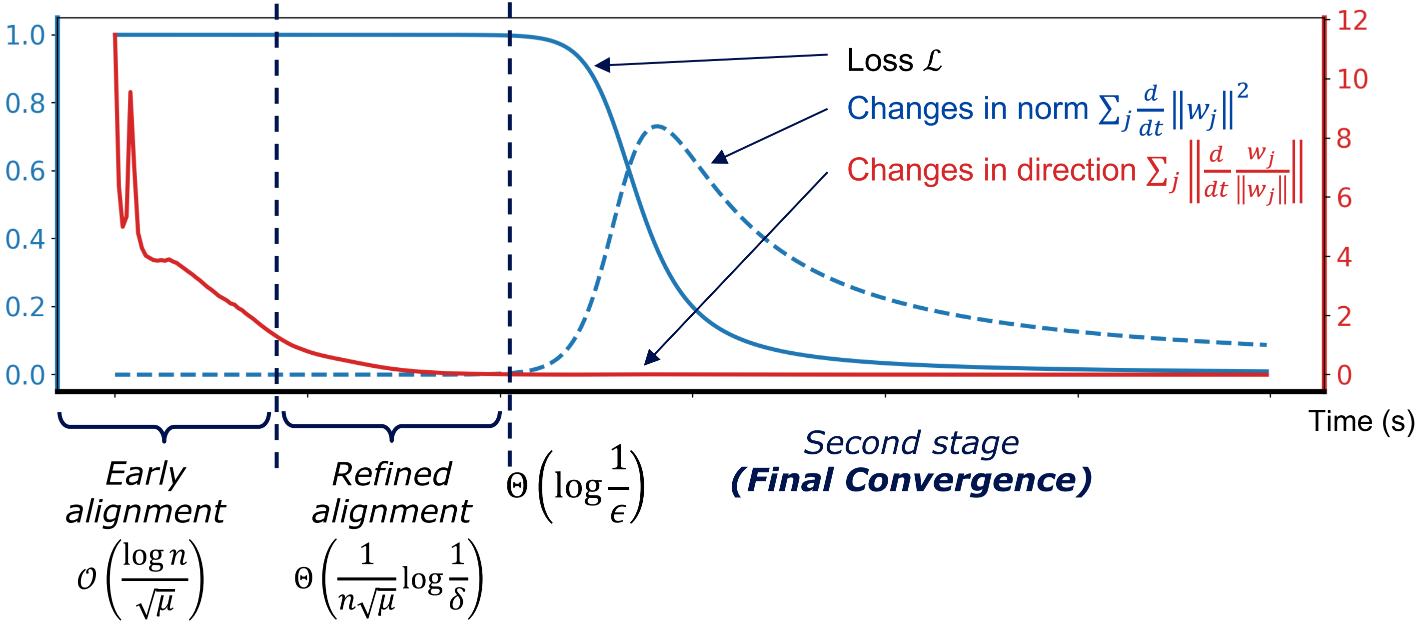
Refined alignment



Proposition (Informal) If refined alignment lasts $\Theta\left(\frac{1}{n\sqrt{\mu}}\log\frac{1}{\delta}\right)$ time, then all neurons are δ-close to positive/negative data center w.r.t. **cosine distance**

- Technical parts for showing this has been presented in [Boursier'22]

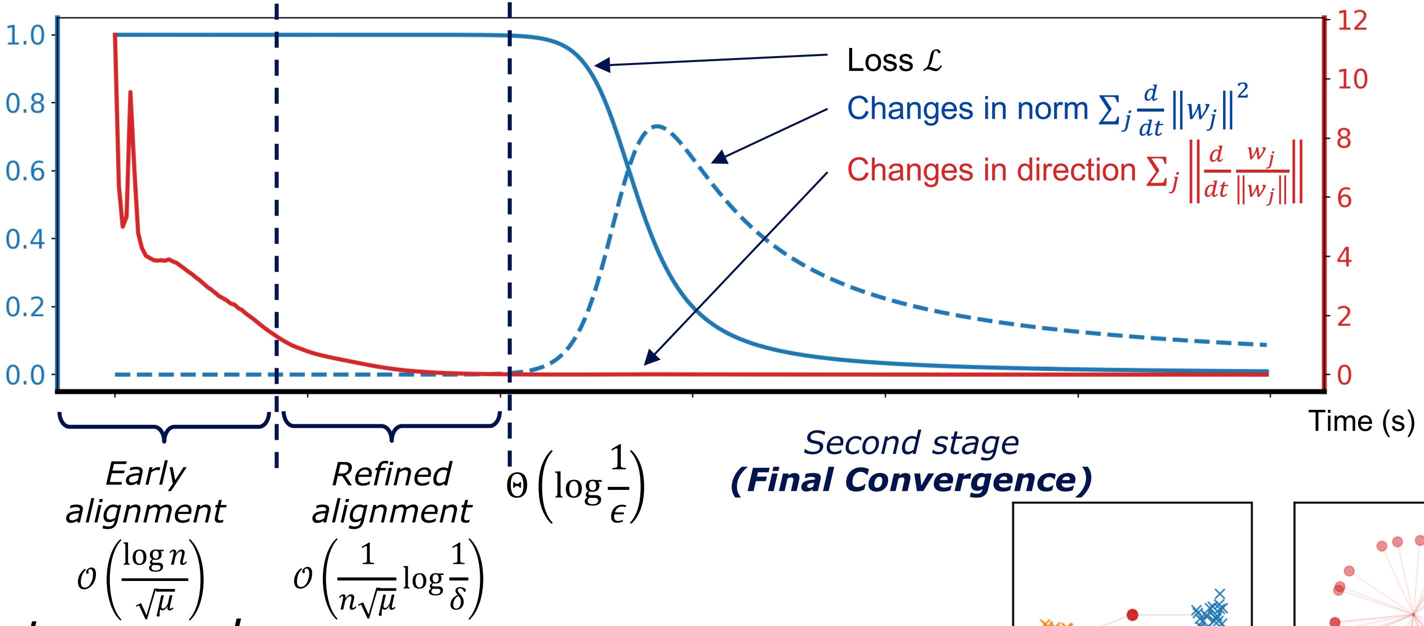
Sufficiently small init. scale

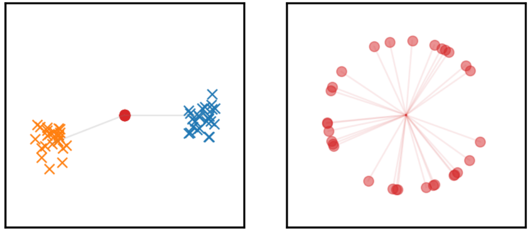


- For the theoretical results to hold, we require a sufficiently small ϵ

$$\mathcal{O}\left(\frac{\log n}{\sqrt{\mu}}\right) + \Theta\left(\frac{1}{n\sqrt{\mu}}\log\frac{1}{\delta}\right) \le \Theta\left(\log\frac{1}{\epsilon}\right) \Rightarrow \epsilon = \mathcal{O}\left(\exp\left(-\frac{1}{n\sqrt{\mu}}(n\log n + \log\frac{1}{\delta})\right)\right)$$

Conclusion





Future work:

- Extends the analysis to general data assumptions
- Deep ReLU networks