Learning with norm-based neural networks: model capacity, function spaces, and computational-statistical gaps

Fanghui Liu

fanghui.liu@warwick.ac.uk

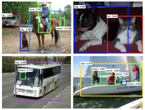
Department of Computer Science, University of Warwick, UK

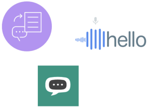
Centre for Discrete Mathematics and its Applications (DIMAP), Warwick

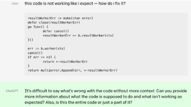
[joint work with Leello Dadi, Zhenyu Zhu, Volkan Cevher (EPFL)]

at Department of Computer Science, University of Wisconsin-Madison

In the era of deep learning



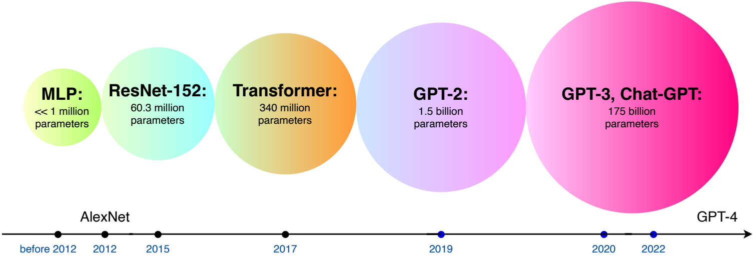



User this code is not working like i expect — how do i fix it?

```
resultWorkerErr := make(chan error)
defer close(resultWorkerErr)
go func() {
        defer cancel()
        resultWorkerErr <- b.resultWorker(ctx)
}()

err := b.worker(ctx)
cancel()
if err == nil {
        return <-resultWorkerErr
}
return multierror.Append(err, <-resultWorkerErr)
```

ChatGPT It's difficult to say what's wrong with the code without more context. Can you provide more information about what the code is supposed to do and what isn't working as expected? Also, is this the entire code or just a part of it?



Learning paradigm transition in the past twenty years

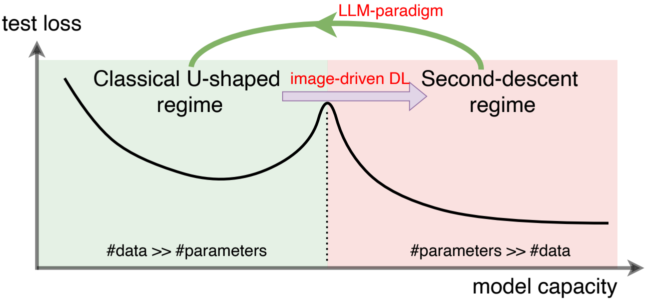


Figure 1: Paradigm among test loss, data, and model capacity.

- double descent [5] (Belkin, Hsu, Ma, Mandal, 2019)

scaling law [15]

test loss = A $\times$ Model Size^{-a} + B $\times$ Data Size^{-b} + C

Learning paradigm transition in the past twenty years

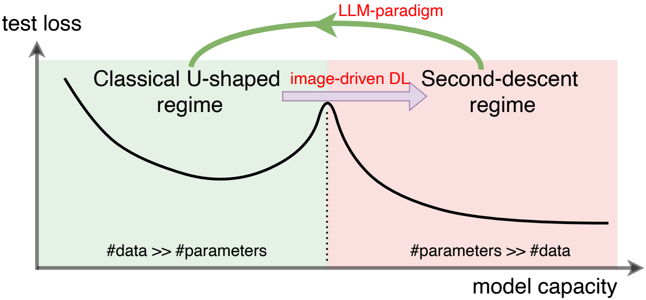


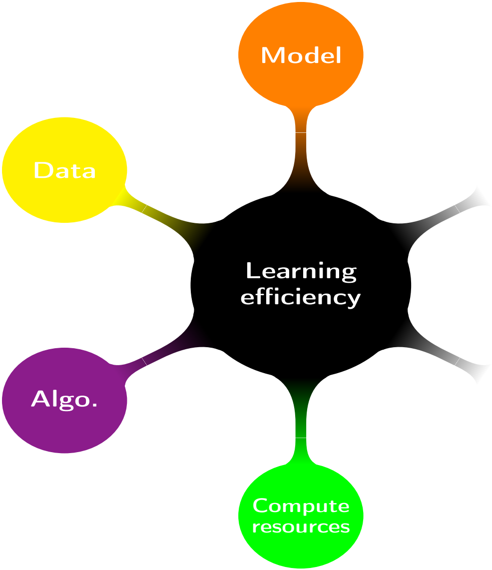
Figure 1: Paradigm among test loss, data, and model capacity.

- double descent [5] (Belkin, Hsu, Ma, Mandal, 2019)

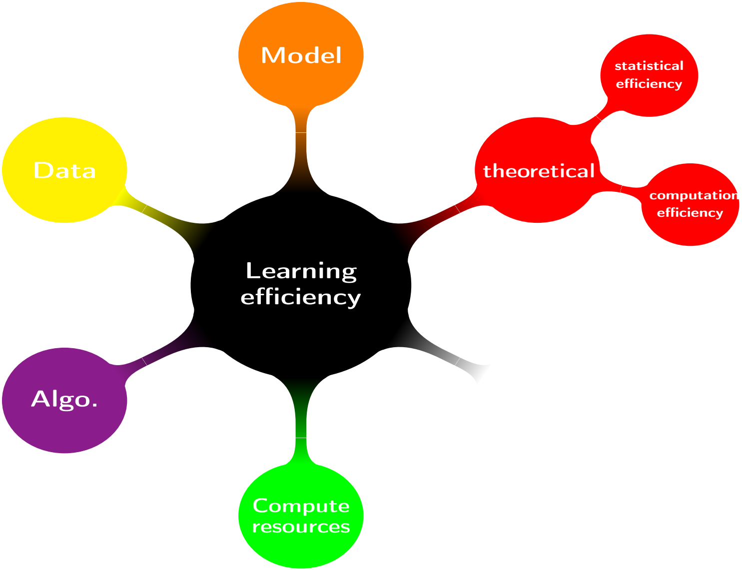
scaling law [15]

test loss = A $\times$ Model Size^{-a} + B $\times$ Data Size^{-b} + C

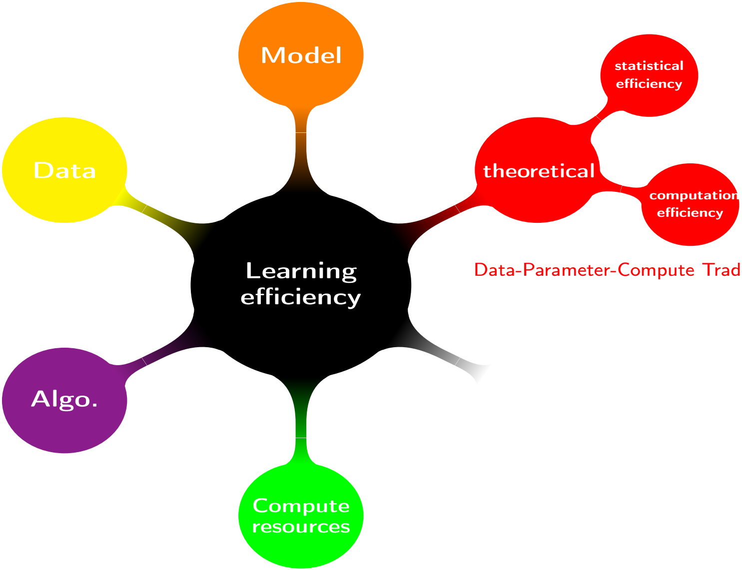
Pipeline: Learning efficiency - curse of dimensionality



Pipeline: Learning efficiency - Theory

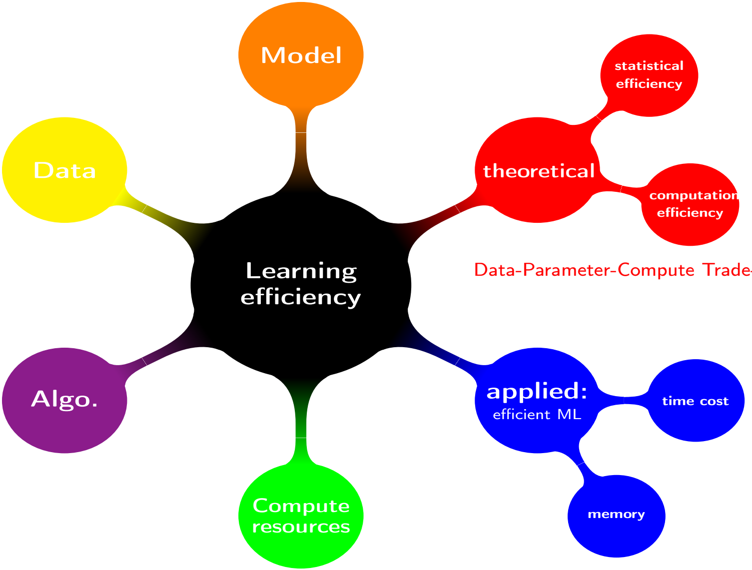


Pipeline: Learning efficiency - Theory



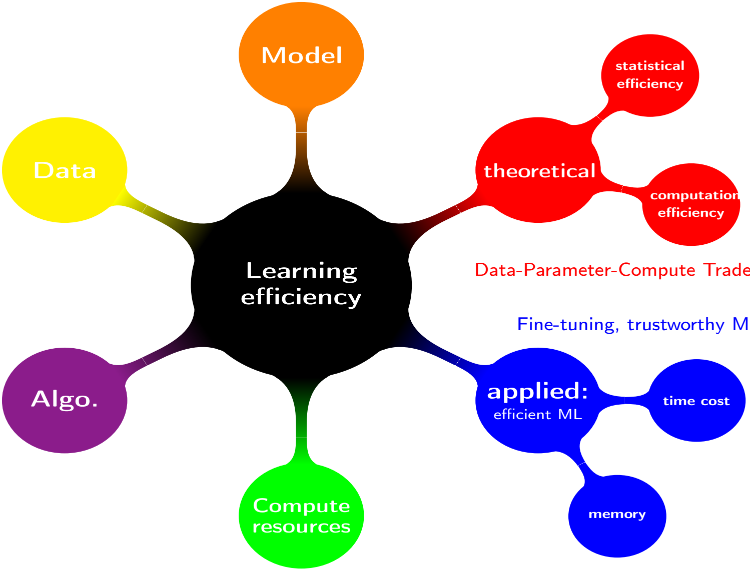
Data-Parameter-Compute Trade-offs!

Pipeline: Learning efficiency - Application



Data-Parameter-Compute Trade-offs!

Pipeline: Learning efficiency - Application



Data-Parameter-Compute Trade-offs!

Fine-tuning, trustworthy ML etc

Outline

- model size is a good metric?
- Learning with path-norm: the perspective of model capacity
- Learning with Barron spaces: the perspective of function space
- statistical/computational learning efficiency

Target

Characterize the "right" model capacity as well as the induced "right" *function spaces* for ML models and track statistical/computational learning efficiency.

Outline

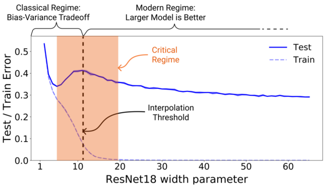
- model size is a good metric?
- Learning with path-norm: the perspective of model capacity
- Learning with Barron spaces: the perspective of function space
- statistical/computational learning efficiency

Target

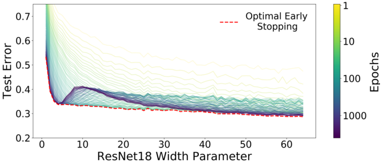
Characterize the "right" model capacity as well as the induced "right" *function spaces* for ML models and track statistical/computational learning efficiency.

Model size is a “right” complexity?

- double descent can disappear!



(a) Results on ResNet18 [20]

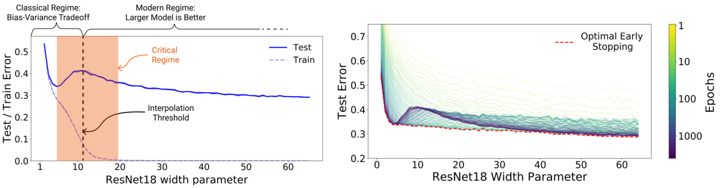


(b) Optimal early stopping [20].

- Empirically: neural network pruning [18], lottery ticket hypothesis [13], fine-tuning with large dropout [31]
- Theoretically: how much over-parameterization is sufficient [8, 28]

Model size is a “right” complexity?

- double descent can disappear!



(a) Results on ResNet18 [20] **(b)** Optimal early stopping [20].

- Empirically: neural network pruning [18], lottery ticket hypothesis [13], fine-tuning with large dropout [31]
- Theoretically: how much over-parameterization is sufficient [8, 28]

What is the "right" model complexity?

◦ Complexity of a prediction rule, e.g.,

- number of parameters
- norm of parameters

[3] (Bartlett, 1998)

The size of the weights is more important than the size of the network!

Norm-based capacity:[21, 25, 22, 10]

name	definition	rank correlation
Parameter Frobenius norm	$\sum_{i=1}^{L} \|\boldsymbol{W}_i\|_F^2$	0.073
Frobenius distance to initialization [19]	$\sum_{i=1}^{L} \|\boldsymbol{W}_i - \boldsymbol{W}_i^0\|_{\mathrm{F}}^2$	−0.263
Spectral complexity [4]	$\prod_{i=1}^{L} \|\boldsymbol{W}_i\| \left(\sum_{i=1}^{L} \frac{\|\boldsymbol{W}_i\|_{2,1}^{3/2}}{\|\boldsymbol{W}_i\|^{3/2}}\right)^{2/3}$	−0.537
Fisher-Rao [16]	$\frac{(L+1)^2}{n} \sum_{i=1}^{n} \langle \boldsymbol{W}, \nabla_{\boldsymbol{W}} \ell(h_{\boldsymbol{W}}(\boldsymbol{x}_i), y_i) \rangle$	0.078
Path-norm [21]	$\sum_{(i_0,\dots,i_L)} \prod_{j=1}^{L} \left(\boldsymbol{W}_{i_j,i_{j-1}}\right)^2$	0.373

Table 1: Complexity measures compared in the empirical study [14], and their correlation with generalization

What is the “right” model complexity?

◦ Complexity of a prediction rule, e.g.,

- number of parameters
- norm of parameters

[3] (Bartlett, 1998)

The size of the weights is more important than the size of the network!

Norm-based capacity:[21, 25, 22, 10]

name	definition	rank correlation
Parameter Frobenius norm	$\sum_{i=1}^{L} \|W_i\|_F^2$	0.073
Frobenius distance to initialization [19]	$\sum_{i=1}^{L} \|W_i - W_i^0\|_F^2$	−0.263
Spectral complexity [4]	$\prod_{i=1}^{L} \|W_i\| \left(\sum_{i=1}^{L} \frac{\|W_i\|_{2,1}^{3/2}}{\|W_i\|^{3/2}} \right)^{2/3}$	−0.537
Fisher-Rao [16]	$\frac{(L+1)^2}{n} \sum_{i=1}^{n} \langle W, \nabla_W \ell(h_W(x_i), y_i) \rangle$	0.078
Path-norm [21]	$\sum_{(i_0,\ldots,i_L)} \prod_{j=1}^{L} \left(W_{i_j, i_{j-1}} \right)^2$	0.373

Table 1: Complexity measures compared in the empirical study [14], and their correlation with generalization

What is the “right” model complexity?

◦ Complexity of a prediction rule, e.g.,

- number of parameters
- norm of parameters

[3] (Bartlett, 1998)

The size of the weights is more important than the size of the network!

Norm-based capacity:[21, 25, 22, 10]

name	definition	rank correlation
Parameter Frobenius norm	$\sum_{i=1}^{L} \|W_i\|_F^2$	0.073
Frobenius distance to initialization [19]	$\sum_{i=1}^{L} \|W_i - W_i^0\|_F^2$	-0.263
Spectral complexity [4]	$\prod_{i=1}^{L} \|W_i\| \left(\sum_{i=1}^{L} \frac{\|W_i\|_{2,1}^{3/2}}{\|W_i\|^{3/2}} \right)^{2/3}$	-0.537
Fisher-Rao [16]	$\frac{(L+1)^2}{n} \sum_{i=1}^{n} \langle W, \nabla_W \ell(h_W(x_i), y_i) \rangle$	0.078
Path-norm [21]	$\sum_{(i_0,\dots,i_L)} \prod_{j=1}^{L} \left(W_{i_j, i_{j-1}} \right)^2$	0.373

Table 1: Complexity measures compared in the empirical study [14], and their correlation with generalization

What is the “right” model complexity?

∘ Complexity of a prediction rule, e.g.,

- number of parameters
- norm of parameters

[3] (Bartlett, 1998)

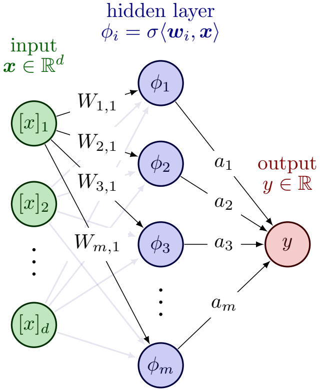
The size of the weights is more important than the size of the network!

Norm-based capacity:[21, 25, 22, 10]

name	definition	rank correlation
Parameter Frobenius norm	$\sum_{i=1}^{L} \lVert W_i \rVert_F^2$	0.073
Frobenius distance to initialization [19]	$\sum_{i=1}^{L} \lVert W_i - W_i^0 \rVert_{\mathrm{F}}^2$	−0.263
Spectral complexity [4]	$\prod_{i=1}^{L} \lVert W_i \rVert \left(\sum_{i=1}^{L} \frac{\lVert W_i \rVert_{2,1}^{3/2}}{\lVert W_i \rVert^{3/2}} \right)^{2/3}$	−0.537
Fisher-Rao [16]	$\frac{(L+1)^2}{n} \sum_{i=1}^{n} \langle W, \nabla_W \ell(h_W(x_i), y_i) \rangle$	0.078
Path-norm [21]	$\sum_{(i_0,\dots,i_L)} \prod_{j=1}^{L} \left(W_{i_j, i_{j-1}} \right)^2$	0.373

Table 1: Complexity measures compared in the empirical study [14], and their correlation with generalization

Two-layer neural networks, path norm

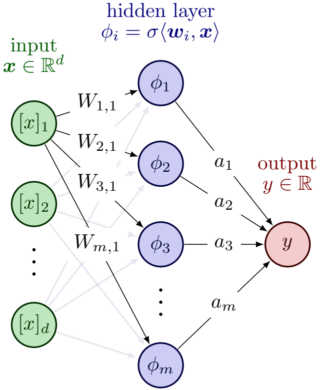


$$\mathcal{P}_m = \left\{ f_\theta(\cdot) := \frac{1}{m} \sum_{k=1}^m a_k \phi\big(\langle \boldsymbol{w}_k, \cdot\rangle\big) \right\}$$

ℓ_1-path norm

$$\|\boldsymbol{\theta}\|_{\mathcal{P}} := \frac{1}{m} \sum_{k=1}^m |a_k| \|\boldsymbol{w}_k\|_1$$

- semi-norm
- relations to Barron spaces $\mathcal{B}$ [2, 12]
  $\|f\|_{\mathcal{B}} \leq \|\boldsymbol{\theta}\|_{\mathcal{P}} \leq 2\|f\|_{\mathcal{B}}$



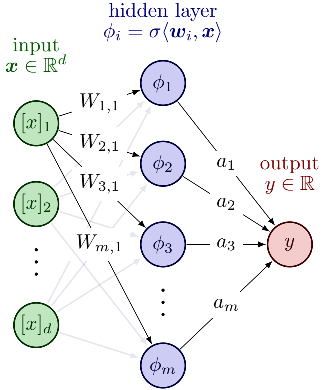
$$\mathcal{P}_m = \left\{ f_{\boldsymbol{\theta}}(\cdot) := \frac{1}{m} \sum_{k=1}^m a_k \phi\big(\langle \boldsymbol{w}_k, \cdot \rangle\big) \right\}$$

ℓ_1-path norm

$$\|\boldsymbol{\theta}\|_{\mathcal{P}} := \frac{1}{m} \sum_{k=1}^m |a_k| \|\boldsymbol{w}_k\|_1$$

- semi-norm
- relations to Barron spaces $\mathcal{B}$ [2, 12]
  $\|f\|_{\mathcal{B}} \leq \|\boldsymbol{\theta}\|_{\mathcal{P}} \leq 2\|f\|_{\mathcal{B}}$

Two-layer neural networks, path norm



$$\mathcal{P}_m = \left\{ f_{\boldsymbol{\theta}}(\cdot) := \frac{1}{m} \sum_{k=1}^m a_k \phi\big(\langle \boldsymbol{w}_k, \cdot \rangle\big) \right\}$$

ℓ_1-path norm

$$\|\boldsymbol{\theta}\|_{\mathcal{P}} := \frac{1}{m} \sum_{k=1}^m |a_k| \|\boldsymbol{w}_k\|_1$$

- semi-norm
- relations to Barron spaces $\mathcal{B}$ [2, 12]
  $\|f\|_{\mathcal{B}} \leq \|\boldsymbol{\theta}\|_{\mathcal{P}} \leq 2\|f\|_{\mathcal{B}}$

*Path norm, Barron spaces, RKHS [11, 6]

Consider a random features model (RFM) [23, 17]

- first layer: $\boldsymbol{w} \overset{iid}{\sim} \mu \in \mathcal{P}(\mathcal{W})$; only train the second layer

infinite many features $f_a(\boldsymbol{x}) = \int_{\mathcal{W}} a(\boldsymbol{w}) \phi(\boldsymbol{x}, \boldsymbol{w}) \mathrm{d}\mu(\boldsymbol{w})$

$$\mathcal{F}_{p,\mu} := \{ f_a : \|\boldsymbol{a}\|_{L^p(\mu)} < \infty \}, \quad \|f\|_{\mathcal{F}_{p,\mu}} := \inf_{f = f_a} \|\boldsymbol{a}\|_{L^p(\mu)}$$

- RFMs $\equiv$ kernel methods by taking $p = 2$ using Representer theorem [24]
- RFMs $\not\equiv$ kernel methods if $p < 2$
- function space: $\mathcal{F}_{\infty,\mu} \subseteq \mathcal{F}_{p,\mu} \subseteq \mathcal{F}_{q,\mu} \subseteq \mathcal{F}_{1,\mu}$ if $p \geq q$

For any $1 \leq p \leq \infty$, define

$$\mathcal{B} = \cup_{\mu \in \mathcal{P}(\mathcal{W})} \mathcal{F}_{p,\mu}\,, \quad \|f\|_{\mathcal{B}} = \inf_{\mu \in \mathcal{P}(\mathcal{W})} \|f\|_{\mathcal{F}_{p,\mu}}$$

- largest
- data-adaptive

*Path norm, Barron spaces, RKHS [11, 6]

Consider a random features model (RFM) [23, 17]

- first layer: $\boldsymbol{w} \overset{iid}{\sim} \mu \in \mathcal{P}(\mathcal{W})$; only train the second layer

infinite many features $f_a(\boldsymbol{x}) = \int_{\mathcal{W}} a(\boldsymbol{w})\phi(\boldsymbol{x}, \boldsymbol{w})\mathrm{d}\mu(\boldsymbol{w})$

$$\mathcal{F}_{p,\mu} := \{f_a : \|\boldsymbol{a}\|_{L^p(\mu)} < \infty\}, \quad \|f\|_{\mathcal{F}_{p,\mu}} := \inf_{f=f_a} \|\boldsymbol{a}\|_{L^p(\mu)}$$

- RFMs $\equiv$ kernel methods by taking $p = 2$ using Representer theorem [24]
- RFMs $\neq$ kernel methods if $p < 2$
- function space: $\mathcal{F}_{\infty,\mu} \subseteq \mathcal{F}_{p,\mu} \subseteq \mathcal{F}_{q,\mu} \subseteq \mathcal{F}_{1,\mu}$ if $p \geq q$

For any $1 \leq p \leq \infty$, define

$$\mathcal{B} = \cup_{\mu \in \mathcal{P}(\mathcal{W})} \mathcal{F}_{p,\mu}, \quad \|f\|_{\mathcal{B}} = \inf_{\mu \in \mathcal{P}(\mathcal{W})} \|f\|_{\mathcal{F}_{p,\mu}}$$

- ○ largest
- ○ data-adaptive

*Path norm, Barron spaces, RKHS [11, 6]

Consider a random features model (RFM) [23, 17]

- first layer: $\boldsymbol{w} \overset{iid}{\sim} \mu \in \mathcal{P}(\mathcal{W})$; only train the second layer

infinite many features $f_a(\boldsymbol{x}) = \int_{\mathcal{W}} a(\boldsymbol{w})\phi(\boldsymbol{x}, \boldsymbol{w})\mathrm{d}\mu(\boldsymbol{w})$

$$\mathcal{F}_{p,\mu} := \{f_a : \|\boldsymbol{a}\|_{L^p(\mu)} < \infty\}, \quad \|f\|_{\mathcal{F}_{p,\mu}} := \inf_{f=f_a} \|\boldsymbol{a}\|_{L^p(\mu)}$$

- RFMs $\equiv$ kernel methods by taking $p = 2$ using Representer theorem [24]
- RFMs $\not\equiv$ kernel methods if $p < 2$
- function space: $\mathcal{F}_{\infty,\mu} \subseteq \mathcal{F}_{p,\mu} \subseteq \mathcal{F}_{q,\mu} \subseteq \mathcal{F}_{1,\mu}$ if $p \geq q$

For any $1 \leq p \leq \infty$, define

$$\mathcal{B} = \cup_{\mu \in \mathcal{P}(\mathcal{W})} \mathcal{F}_{p,\mu}, \quad \|f\|_{\mathcal{B}} = \inf_{\mu \in \mathcal{P}(\mathcal{W})} \|f\|_{\mathcal{F}_{p,\mu}}$$

- largest
- data-adaptive

*Path norm, Barron spaces, RKHS [11, 6]

Consider a random features model (RFM) [23, 17]

- first layer: $\boldsymbol{w} \overset{iid}{\sim} \mu \in \mathcal{P}(\mathcal{W})$; only train the second layer

infinite many features $f_a(\boldsymbol{x}) = \int_{\mathcal{W}} a(\boldsymbol{w})\phi(\boldsymbol{x}, \boldsymbol{w})\mathrm{d}\mu(\boldsymbol{w})$

$$\mathcal{F}_{p,\mu} := \{f_a : \|\boldsymbol{a}\|_{L^p(\mu)} < \infty\}, \quad \|f\|_{\mathcal{F}_{p,\mu}} := \inf_{f=f_a} \|\boldsymbol{a}\|_{L^p(\mu)}$$

- RFMs $\equiv$ kernel methods by taking $p = 2$ using Representer theorem [24]
- RFMs $\not\equiv$ kernel methods if $p < 2$
- function space: $\mathcal{F}_{\infty,\mu} \subseteq \mathcal{F}_{p,\mu} \subseteq \mathcal{F}_{q,\mu} \subseteq \mathcal{F}_{1,\mu}$ if $p \geq q$

For any $1 \leq p \leq \infty$, define

$$\mathcal{B} = \cup_{\mu \in \mathcal{P}(\mathcal{W})} \mathcal{F}_{p,\mu}, \quad \|f\|_{\mathcal{B}} = \inf_{\mu \in \mathcal{P}(\mathcal{W})} \|f\|_{\mathcal{F}_{p,\mu}}$$

- largest
- data-adaptive

*Path norm, Barron spaces, RKHS [11, 6]

Consider a random features model (RFM) [23, 17]

- first layer: $\boldsymbol{w} \overset{iid}{\sim} \mu \in \mathcal{P}(\mathcal{W})$; only train the second layer

infinite many features $f_a(\boldsymbol{x}) = \int_{\mathcal{W}} a(\boldsymbol{w})\phi(\boldsymbol{x}, \boldsymbol{w})\mathrm{d}\mu(\boldsymbol{w})$

$$\mathcal{F}_{p,\mu} := \{f_a : \|\boldsymbol{a}\|_{L^p(\mu)} < \infty\}, \quad \|f\|_{\mathcal{F}_{p,\mu}} := \inf_{f=f_a} \|\boldsymbol{a}\|_{L^p(\mu)}$$

- RFMs $\equiv$ kernel methods by taking $p = 2$ using Representer theorem [24]
- RFMs $\not\equiv$ kernel methods if $p < 2$
- function space: $\mathcal{F}_{\infty,\mu} \subseteq \mathcal{F}_{p,\mu} \subseteq \mathcal{F}_{q,\mu} \subseteq \mathcal{F}_{1,\mu}$ if $p \geq q$

For any $1 \leq p \leq \infty$, define

$$\mathcal{B} = \cup_{\mu \in \mathcal{P}(\mathcal{W})} \mathcal{F}_{p,\mu}\,, \quad \|f\|_{\mathcal{B}} = \inf_{\mu \in \mathcal{P}(\mathcal{W})} \|f\|_{\mathcal{F}_{p,\mu}}$$

- largest
- data-adaptive

*Path norm, Barron spaces, RKHS [11, 6]

Consider a random features model (RFM) [23, 17]

- first layer: $\boldsymbol{w} \overset{iid}{\sim} \mu \in \mathcal{P}(\mathcal{W})$; only train the second layer

infinite many features $f_a(\boldsymbol{x}) = \int_{\mathcal{W}} a(\boldsymbol{w})\phi(\boldsymbol{x}, \boldsymbol{w})\mathrm{d}\mu(\boldsymbol{w})$

$$\mathcal{F}_{p,\mu} := \{f_a : \|\boldsymbol{a}\|_{L^p(\mu)} < \infty\}, \quad \|f\|_{\mathcal{F}_{p,\mu}} := \inf_{f=f_a} \|\boldsymbol{a}\|_{L^p(\mu)}$$

- RFMs $\equiv$ kernel methods by taking $p = 2$ using Representer theorem [24]
- RFMs $\not\equiv$ kernel methods if $p < 2$
- function space: $\mathcal{F}_{\infty,\mu} \subseteq \mathcal{F}_{p,\mu} \subseteq \mathcal{F}_{q,\mu} \subseteq \mathcal{F}_{1,\mu}$ if $p \geq q$

For any $1 \leq p \leq \infty$, define

$$\mathcal{B} = \cup_{\mu \in \mathcal{P}(\mathcal{W})} \mathcal{F}_{p,\mu}\,, \quad \|f\|_{\mathcal{B}} = \inf_{\mu \in \mathcal{P}(\mathcal{W})} \|f\|_{\mathcal{F}_{p,\mu}}$$

◦ largest

◦ data-adaptive

Our results: statistical guarantees

For the class of two-layer neural networks $\mathcal{G}_R = \{f_{\boldsymbol{\theta}} \in \mathcal{P}_m : \|\boldsymbol{\theta}\|_{\mathcal{P}} \leqslant R\}$

$$\widehat{f_{\boldsymbol{\theta}}} := \operatorname*{argmin}_{f_{\boldsymbol{\theta}} \in \mathcal{G}_R} \frac{1}{n} \sum_{i=1}^{n} (y_i - f_{\boldsymbol{\theta}}(\boldsymbol{x}_i))^2 .$$

Theorem (Liu, Dadi, Cevher, JMLR 2024)

Under standard assumptions (bounded data, $f^\star \in \mathcal{B}$), for two-layer over-parameterized neural networks, we have

$$\left\| \widehat{f_{\boldsymbol{\theta}}} - f^\star \right\|^2_{L^2_{\rho_X}} \lesssim \frac{R^2}{m} + R^2 d^{\frac{1}{3}} n^{-\frac{d+2}{2d+2}} \qquad w.h.p.$$

$n^{-\frac{d+2}{2d+2}}$ is always faster than $n^{-\frac{1}{2}}$: No curse of dimensionality!

Our results: statistical guarantees

For the class of two-layer neural networks $\mathcal{G}_R = \{f_{\boldsymbol{\theta}} \in \mathcal{P}_m : \|\boldsymbol{\theta}\|_{\mathcal{P}} \leqslant R\}$

$$\widehat{f_{\boldsymbol{\theta}}} := \operatorname*{argmin}_{f_{\boldsymbol{\theta}} \in \mathcal{G}_R} \frac{1}{n} \sum_{i=1}^{n} (y_i - f_{\boldsymbol{\theta}}(\boldsymbol{x}_i))^2 \,.$$

Theorem (Liu, Dadi, Cevher, JMLR 2024)

Under standard assumptions (bounded data, $f^\star \in \mathcal{B}$), for two-layer over-parameterized neural networks, we have

$$\left\| \widehat{f_{\boldsymbol{\theta}}} - f^\star \right\|^2_{L^2_{\rho_X}} \lesssim \frac{R^2}{m} + R^2 d^{\frac{1}{3}} n^{-\frac{d+2}{2d+2}} \qquad w.h.p.$$

$n^{-\frac{d+2}{2d+2}}$ is always faster than $n^{-\frac{1}{2}}$: No curse of dimensionality!

Our results: statistical guarantees

For the class of two-layer neural networks $\mathcal{G}_R = \{f_{\boldsymbol{\theta}} \in \mathcal{P}_m : \|\boldsymbol{\theta}\|_{\mathcal{P}} \leqslant R\}$

$$\widehat{f_{\boldsymbol{\theta}}} := \operatorname*{argmin}_{f_{\boldsymbol{\theta}} \in \mathcal{G}_R} \frac{1}{n} \sum_{i=1}^{n} (y_i - f_{\boldsymbol{\theta}}(\boldsymbol{x}_i))^2 \,.$$

Theorem (Liu, Dadi, Cevher, JMLR 2024)

Under standard assumptions (bounded data, $f^\star \in \mathcal{B}$), for two-layer over-parameterized neural networks, we have

$$\left\| \widehat{f_{\boldsymbol{\theta}}} - f^\star \right\|^2_{L^2_{\rho_X}} \lesssim \frac{R^2}{m} + R^2 d^{\frac{1}{3}} n^{-\frac{d+2}{2d+2}} \qquad w.h.p.$$

$n^{-\frac{d+2}{2d+2}}$ is always faster than $n^{-\frac{1}{2}}$: No curse of dimensionality!

Metric entropy: minimax?

Proposition (metric entropy)

For bounded data $\|\boldsymbol{x}\|_\infty \leq 1$, *denote* $\mathcal{G}_R = \{f_{\boldsymbol{\theta}} \in \mathcal{P}_m : \|\boldsymbol{\theta}\|_{\mathcal{P}} \leqslant R\}$, *the metric entropy of* $\mathcal{G}_1$ *can be bounded by*

$$\log \mathcal{N}_2(\mathcal{G}_1, \epsilon) \leqslant C d \epsilon^{-\frac{2d}{d+2}}\,, \quad \forall \epsilon > 0 \quad \text{and} \quad d \geq 5\,,$$

with some universal constant C *independent of* d.

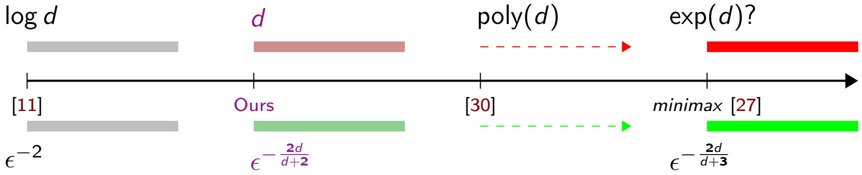
Metric entropy: minimax?

Proposition (metric entropy)

For bounded data $\|\boldsymbol{x}\|_\infty \leq 1$, denote $\mathcal{G}_R = \{f_{\boldsymbol{\theta}} \in \mathcal{P}_m : \|\boldsymbol{\theta}\|_{\mathcal{P}} \leqslant R\}$, the metric entropy of $\mathcal{G}_1$ can be bounded by

$$\log \mathcal{N}_2(\mathcal{G}_1, \epsilon) \leqslant C d \epsilon^{-\frac{2d}{d+2}}\,, \quad \forall \epsilon > 0 \quad \text{and} \quad d \geq 5\,,$$

with some universal constant C independent of d.



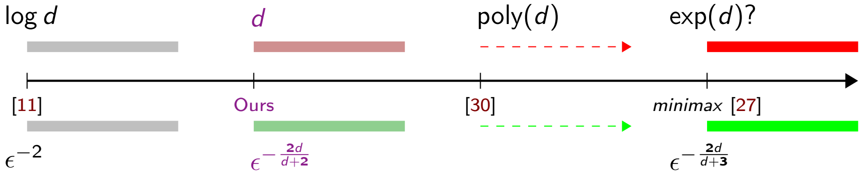
Metric entropy: minimax?

Proposition (metric entropy)

For bounded data $\|\boldsymbol{x}\|_\infty \leq 1$, denote $\mathcal{G}_R = \{f_{\boldsymbol{\theta}} \in \mathcal{P}_m : \|\boldsymbol{\theta}\|_{\mathcal{P}} \leqslant R\}$, the metric entropy of $\mathcal{G}_1$ can be bounded by

$$\log \mathcal{N}_2(\mathcal{G}_1, \epsilon) \leqslant C d \epsilon^{-\frac{2d}{d+2}}, \quad \forall \epsilon > 0 \quad \textit{and} \quad d \geq 5,$$

with some universal constant C independent of d.



The “best” trade-off between ϵ and d.

Proof sketch: convex hull technique and its constant!

- Consider the following function space

$$\mathcal{F} = \{\sigma(\langle \widetilde{\boldsymbol{w}}, \cdot \rangle) : \widetilde{\boldsymbol{w}} \in \mathcal{W}\} \cup \{0\} \cup \{-\sigma(\langle \widetilde{\boldsymbol{w}}, \cdot \rangle) : \widetilde{\boldsymbol{w}} \in \mathbb{S}_1^{d-1} \text{ with the } \ell_1 \text{ ball}\}$$

- the convex hull of $\mathcal{F}$ is

$$\overline{\mathrm{conv}}\mathcal{F} = \left\{ \sum_{i=1}^m \alpha_i f_i \,\middle|\, f_i \in \mathcal{F}, \sum_{i=1}^m \alpha_i = 1, \alpha_i \geqslant 0, m \in \mathbb{N} \right\}.$$

- convex hull technique [29, Theorem 2.6.9]

$$\log \mathcal{N}_2(\mathcal{G}_1, \epsilon) \leqslant \log \mathcal{N}_2(\overline{\mathrm{conv}}\mathcal{F}, \epsilon, \mu) \leqslant C \left(\frac{1}{\epsilon}\right)^{\frac{2d}{d+2}}.$$

- control the constant C

$$C := \underbrace{D_k}_{=\Theta(d)} [\underbrace{C_k}_{=\Theta(1)} (2^{d+1}+1)^{\frac{1}{d}}]^{\frac{2d}{d+2}} \leq 10^7 d \quad \text{if } d > 5$$

Proof sketch: convex hull technique and its constant!

- Consider the following function space

  $\mathcal{F} = \{\sigma(\langle \widetilde{\boldsymbol{w}}, \cdot \rangle) : \widetilde{\boldsymbol{w}} \in \mathcal{W}\} \cup \{0\} \cup \{-\sigma(\langle \widetilde{\boldsymbol{w}}, \cdot \rangle) : \widetilde{\boldsymbol{w}} \in \mathbb{S}_1^{d-1}$ with the ℓ_1 ball$\}$

- the convex hull of $\mathcal{F}$ is

$$\overline{\mathrm{conv}}\mathcal{F} = \left\{ \sum_{i=1}^{m} \alpha_i f_i \middle| f_i \in \mathcal{F}, \sum_{i=1}^{m} \alpha_i = 1, \alpha_i \geqslant 0, m \in \mathbb{N} \right\}.$$

- convex hull technique [29, Theorem 2.6.9]

$$\log \mathcal{N}_2(\mathcal{G}_1, \epsilon) \leqslant \log \mathcal{N}_2(\overline{\mathrm{conv}}\mathcal{F}, \epsilon, \mu) \leqslant C \left(\frac{1}{\epsilon}\right)^{\frac{2d}{d+2}}.$$

- control the constant C

$$C := \underbrace{D_k}_{=\Theta(d)} [\underbrace{C_k}_{=\Theta(1)} (2^{d+1} + 1)^{\frac{1}{d}}]^{\frac{2d}{d+2}} \leq 10^7 d \quad \text{if } d > 5$$

Proof sketch: convex hull technique and its constant!

- Consider the following function space

$$\mathcal{F} = \{\sigma(\langle \widetilde{\boldsymbol{w}}, \cdot \rangle) : \widetilde{\boldsymbol{w}} \in \mathcal{W}\} \cup \{0\} \cup \{-\sigma(\langle \widetilde{\boldsymbol{w}}, \cdot \rangle) : \widetilde{\boldsymbol{w}} \in \mathbb{S}_1^{d-1} \text{ with the } \ell_1 \text{ ball}\}$$

- the convex hull of $\mathcal{F}$ is

$$\overline{\mathrm{conv}}\mathcal{F} = \left\{ \sum_{i=1}^{m} \alpha_i f_i \middle| f_i \in \mathcal{F}, \sum_{i=1}^{m} \alpha_i = 1, \alpha_i \geqslant 0, m \in \mathbb{N} \right\}.$$

- convex hull technique [29, Theorem 2.6.9]

$$\log \mathcal{N}_2(\mathcal{G}_1, \epsilon) \leqslant \log \mathcal{N}_2(\overline{\mathrm{conv}}\mathcal{F}, \epsilon, \mu) \leqslant C \left(\frac{1}{\epsilon}\right)^{\frac{2d}{d+2}}.$$

- control the constant C

$$C := \underbrace{D_k}_{=\Theta(d)} \Big[\underbrace{C_k}_{=\Theta(1)} (2^{d+1}+1)^{\frac{1}{d}} \Big]^{\frac{2d}{d+2}} \leq 10^7 d \quad \text{if } d > 5$$

Proof sketch: convex hull technique and its constant!

- Consider the following function space

$$\mathcal{F} = \{\sigma(\langle \widetilde{\boldsymbol{w}}, \cdot\rangle) : \widetilde{\boldsymbol{w}} \in \mathcal{W}\} \cup \{0\} \cup \{-\sigma(\langle \widetilde{\boldsymbol{w}}, \cdot\rangle) : \widetilde{\boldsymbol{w}} \in \mathbb{S}_1^{d-1} \text{ with the } \ell_1 \text{ ball}\}$$

- the convex hull of $\mathcal{F}$ is

$$\overline{\mathrm{conv}}\mathcal{F} = \left\{ \sum_{i=1}^{m} \alpha_i f_i \middle| f_i \in \mathcal{F}, \sum_{i=1}^{m} \alpha_i = 1, \alpha_i \geqslant 0, m \in \mathbb{N} \right\}.$$

- convex hull technique [29, Theorem 2.6.9]

$$\log \mathcal{N}_2(\mathcal{G}_1, \epsilon) \leqslant \log \mathcal{N}_2(\overline{\mathrm{conv}}\mathcal{F}, \epsilon, \mu) \leqslant C \left(\frac{1}{\epsilon}\right)^{\frac{2d}{d+2}}.$$

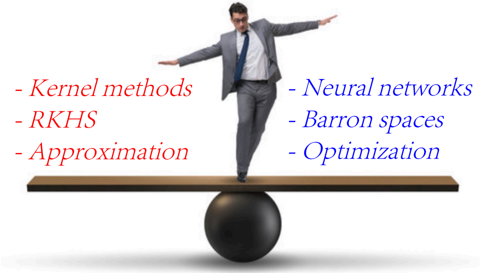
- control the constant C

$$C := \underbrace{D_k}_{=\Theta(d)} [\underbrace{C_k}_{=\Theta(1)} (2^{d+1} + 1)^{\frac{1}{d}}]^{\frac{2d}{d+2}} \leq 10^7 d \quad \text{if } d > 5$$

Computational-statistical gaps

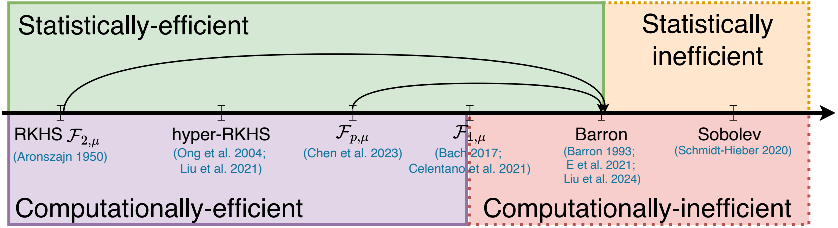
Optimization in Barron spaces is NP hard: curse of dimensionality! [1]

Optimization in Barron spaces is NP hard: curse of dimensionality! [1]



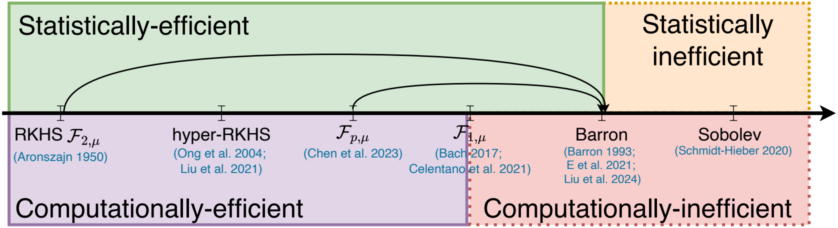
Computational-statistical gaps

Optimization in Barron spaces is NP hard: curse of dimensionality! [1]



Computational-statistical gaps

Optimization in Barron spaces is NP hard: curse of dimensionality! [1]



Do some Barron functions can be learned by two-layer NNs, both statistically and computationally efficient?

Learning with multiple ReLU neurons

Can we learn multiple ReLU neurons by two-layer NNs, both statistically and computationally efficient?

$$f^\star(\boldsymbol{x}) = \sum_{j=1}^{k} a_j \sigma(\langle \boldsymbol{v}_j, \boldsymbol{x} \rangle), k = \mathcal{O}(1)$$

$\|\hat{f} - f^\star\|_{L^2(\mathrm{d}\mu)} \leq \epsilon$ from $\{\boldsymbol{x}_i, f^\star(\boldsymbol{x}_i)\}_{i=1}^n$ with $\boldsymbol{x}_i \sim \mathcal{N}(0, \boldsymbol{I}_d)$

Theorem ([7] PAC learning $f^\star$ under Gaussian measure)

There exists an algorithm that requires time/samples at $(d/\epsilon)^{\mathcal{O}(k^2)}$

- correlational statistical query (CSQ): $|\tilde{q} - \mathbb{E}_{\boldsymbol{x},y}[\psi(\boldsymbol{x})y]| \leq \tau$
- the lower bound for any CSQ algorithm is $d^{\Omega(k)}$ [9]

Learning with multiple ReLU neurons

Can we learn multiple ReLU neurons by two-layer NNs, both statistically and computationally efficient?

$$f^\star(\boldsymbol{x}) = \sum_{j=1}^{k} a_j \sigma(\langle \boldsymbol{v}_j, \boldsymbol{x}\rangle), k = \mathcal{O}(1)$$

$\|\hat{f} - f^\star\|_{L^2(\mathrm{d}\mu)} \leq \epsilon$ from $\{\boldsymbol{x}_i, f^\star(\boldsymbol{x}_i)\}_{i=1}^n$ with $\boldsymbol{x}_i \sim \mathcal{N}(0, \boldsymbol{I}_d)$

Theorem ([7] PAC learning $f^\star$ under Gaussian measure)

There exists an algorithm that requires time/samples at $(d/\epsilon)^{\mathcal{O}(k^2)}$

- correlational statistical query (CSQ): $|\tilde{q} - \mathbb{E}_{\boldsymbol{x},y}[\psi(\boldsymbol{x})y]| \leq \tau$
- the lower bound for any CSQ algorithm is $d^{\Omega(k)}$ [9]

Learning with multiple ReLU neurons

Can we learn multiple ReLU neurons by two-layer NNs, both statistically and computationally efficient?

$$f^\star(\boldsymbol{x}) = \sum_{j=1}^{k} a_j \sigma(\langle \boldsymbol{v}_j, \boldsymbol{x} \rangle), k = \mathcal{O}(1)$$

$\|\hat{f} - f^\star\|_{L^2(\mathrm{d}\mu)} \leq \epsilon$ from $\{\boldsymbol{x}_i, f^\star(\boldsymbol{x}_i)\}_{i=1}^n$ with $\boldsymbol{x}_i \sim \mathcal{N}(0, \boldsymbol{I}_d)$

Theorem ([7] PAC learning $f^\star$ under Gaussian measure)

There exists an algorithm that requires time/samples at $(d/\epsilon)^{\mathcal{O}(k^2)}$

- correlational statistical query (CSQ): $|\tilde{q} - \mathbb{E}_{\boldsymbol{x},y}[\psi(\boldsymbol{x})y]| \leq \tau$
- the lower bound for any CSQ algorithm is $d^{\Omega(k)}$ [9]

Learning with multiple ReLU neurons

Can we learn multiple ReLU neurons by two-layer NNs, both statistically and computationally efficient?

$$f^\star(\boldsymbol{x}) = \sum_{j=1}^{k} a_j \sigma(\langle \boldsymbol{v}_j, \boldsymbol{x} \rangle), k = \mathcal{O}(1)$$

$\|\hat{f} - f^\star\|_{L^2(\mathrm{d}\mu)} \leq \epsilon$ from $\{\boldsymbol{x}_i, f^\star(\boldsymbol{x}_i)\}_{i=1}^n$ with $\boldsymbol{x}_i \sim \mathcal{N}(0, \boldsymbol{I}_d)$

Theorem ([7] PAC learning $f^\star$ under Gaussian measure)

There exists an algorithm that requires time/samples at $(d/\epsilon)^{\mathcal{O}(k^2)}$

- correlational statistical query (CSQ): $|\tilde{q} - \mathbb{E}_{\boldsymbol{x},y}[\psi(\boldsymbol{x})y]| \leq \tau$
- the lower bound for any CSQ algorithm is $d^{\Omega(k)}$ [9]

Learning with multiple ReLU neurons

Can we learn multiple ReLU neurons by two-layer NNs, both statistically and computationally efficient?

$$f^\star(\boldsymbol{x}) = \sum_{j=1}^{k} a_j \sigma(\langle \boldsymbol{v}_j, \boldsymbol{x} \rangle), k = \mathcal{O}(1)$$

$\|\hat{f} - f^\star\|_{L^2(\mathrm{d}\mu)} \leq \epsilon$ from $\{\boldsymbol{x}_i, f^\star(\boldsymbol{x}_i)\}_{i=1}^n$ with $\boldsymbol{x}_i \sim \mathcal{N}(0, \boldsymbol{I}_d)$

Theorem ([7] PAC learning $f^\star$ under Gaussian measure)

There exists an algorithm that requires time/samples at $(d/\epsilon)^{\mathcal{O}(k^2)}$

- correlational statistical query (CSQ): $|\tilde{q} - \mathbb{E}_{\boldsymbol{x},y}[\psi(\boldsymbol{x})y]| \leq \tau$
- the lower bound for any CSQ algorithm is $d^{\Omega(k)}$ [9]

How to do it by gradient descent

Learning multi ReLU neurons by two-layer NN via GD

$$L(\boldsymbol{W}) = \frac{1}{2}\mathbb{E}_{\boldsymbol{x}\sim\mathcal{N}(0,\boldsymbol{I}_d)}\left(\sum_{i=1}^{m}\sigma(\langle \boldsymbol{w}_i, \boldsymbol{x}\rangle) - f^\star(\boldsymbol{x})\right)^2$$

- Gaussian initialization for student neurons: $\boldsymbol{w}_i \sim \mathcal{N}(0, \sigma^2 \boldsymbol{I}_d)$
- angle: $\theta_{ij} \triangleq \angle(\boldsymbol{w}_i, \boldsymbol{v}_j)$, $\varphi_{ij} \triangleq \angle(\boldsymbol{w}_i, \boldsymbol{w}_j)$

Assumption

- *diverse teacher neurons: $\{\boldsymbol{v}_j\}_{j=1}^k$ are orthogonal and $\|\boldsymbol{v}_j\|_2 = \texttt{const}$*
- *warm start: the smallest angle not close to orthogonal*
  - *weak recovery: $\langle \boldsymbol{w}_i, \boldsymbol{v}_{i^\star}\rangle \gg \langle \boldsymbol{w}_i, \boldsymbol{v}_j\rangle$*

How to do it by gradient descent

Learning multi ReLU neurons by two-layer NN via GD

$$L(\boldsymbol{W}) = \frac{1}{2}\mathbb{E}_{\boldsymbol{x}\sim\mathcal{N}(0,\boldsymbol{I}_d)}\left(\sum_{i=1}^{m}\sigma(\langle \boldsymbol{w}_i, \boldsymbol{x}\rangle) - f^\star(\boldsymbol{x})\right)^2$$

- Gaussian initialization for student neurons: $\boldsymbol{w}_i \sim \mathcal{N}(0, \sigma^2 \boldsymbol{I}_d)$
- angle: $\theta_{ij} \triangleq \angle(\boldsymbol{w}_i, \boldsymbol{v}_j)$, $\varphi_{ij} \triangleq \angle(\boldsymbol{w}_i, \boldsymbol{w}_j)$

Assumption

- *diverse teacher neurons:* $\{\boldsymbol{v}_j\}_{j=1}^k$ *are orthogonal and* $\|\boldsymbol{v}_j\|_2 = \mathtt{const}$
- *warm start: the smallest angle not close to orthogonal*
  - *weak recovery:* $\langle \boldsymbol{w}_i, \boldsymbol{v}_{i^\star}\rangle \gg \langle \boldsymbol{w}_i, \boldsymbol{v}_j\rangle$

How to do it by gradient descent

Learning multi ReLU neurons by two-layer NN via GD

$$L(\boldsymbol{W}) = \frac{1}{2}\mathbb{E}_{\boldsymbol{x}\sim\mathcal{N}(0,\boldsymbol{I}_d)}\left(\sum_{i=1}^{m}\sigma(\langle \boldsymbol{w}_i, \boldsymbol{x}\rangle) - f^\star(\boldsymbol{x})\right)^2$$

- Gaussian initialization for student neurons: $\boldsymbol{w}_i \sim \mathcal{N}(0, \sigma^2 \boldsymbol{I}_d)$
- angle: $\theta_{ij} \triangleq \angle(\boldsymbol{w}_i, \boldsymbol{v}_j)$, $\varphi_{ij} \triangleq \angle(\boldsymbol{w}_i, \boldsymbol{w}_j)$

Assumption

- *diverse teacher neurons: $\{\boldsymbol{v}_j\}_{j=1}^{k}$ are orthogonal and $\|\boldsymbol{v}_j\|_2 = \texttt{const}$*
- *warm start: the smallest angle not close to orthogonal*
    - *weak recovery: $\langle \boldsymbol{w}_i, \boldsymbol{v}_{i^\star}\rangle \gg \langle \boldsymbol{w}_i, \boldsymbol{v}_j\rangle$*

How to do it by gradient descent

Learning multi ReLU neurons by two-layer NN via GD

$$L(\boldsymbol{W}) = \frac{1}{2}\mathbb{E}_{\boldsymbol{x}\sim\mathcal{N}(0,\boldsymbol{I}_d)}\left(\sum_{i=1}^{m}\sigma(\langle \boldsymbol{w}_i, \boldsymbol{x}\rangle) - f^\star(\boldsymbol{x})\right)^2$$

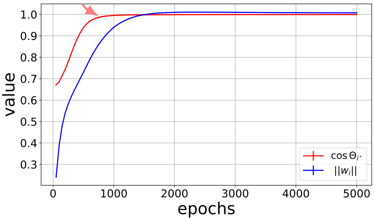
- Gaussian initialization for student neurons: $\boldsymbol{w}_i \sim \mathcal{N}(0, \sigma^2 \boldsymbol{I}_d)$
- angle: $\theta_{ij} \triangleq \angle(\boldsymbol{w}_i, \boldsymbol{v}_j)$, $\varphi_{ij} \triangleq \angle(\boldsymbol{w}_i, \boldsymbol{w}_j)$

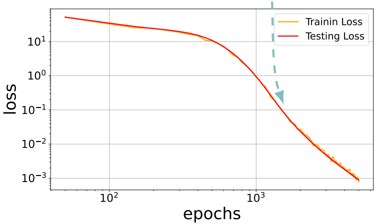
Assumption

- *diverse teacher neurons:* $\{\boldsymbol{v}_j\}_{j=1}^k$ *are orthogonal and* $\|\boldsymbol{v}_j\|_2 = \texttt{const}$
- *warm start: the smallest angle not close to orthogonal*
  - *weak recovery:* $\langle \boldsymbol{w}_i, \boldsymbol{v}_{i^\star}\rangle \gg \langle \boldsymbol{w}_i, \boldsymbol{v}_j\rangle$

How does GD learn features and recover teacher neurons?

- align $\theta_{i^*} \to 0$ norm converge then fit





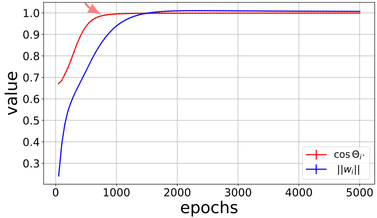
Theorem (Zhu, Liu, Cevher, 2024)

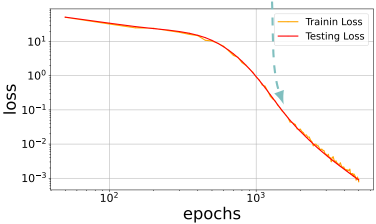
For sufficiently small initialization and step-size $\sigma, \eta = o(m^{-k^2})$, then after weak recovery, there exists a time $T_0 = \frac{1}{\eta}$ such that $\forall T \in \mathbb{N}$ and $i \in [m]$,

$$L(\mathbf{W}(T+T_0)) \le \mathcal{O}\left(\frac{1}{T^3}\right), \|\mathbf{w}_i(T+T_0)\|_2 = \Theta\left(\frac{k\|\mathbf{v}\|_2}{m}\right) \quad w.h.p.$$

How does GD learn features and recover teacher neurons?

- align $\theta_{i^*} \to 0$ norm converge then fit





Theorem (Zhu, Liu, Cevher, 2024)

For sufficiently small initialization and step-size $\sigma, \eta = o(m^{-k^2})$, then after weak recovery, there exists a time $T_0 = \frac{1}{\eta}$ such that $\forall T \in \mathbb{N}$ and $i \in [m]$,

$$L(\boldsymbol{W}(T+T_0)) \leq \mathcal{O}\left(\frac{1}{T^3}\right), \|\boldsymbol{w}_i(T+T_0)\|_2 = \Theta\left(\frac{k\|\boldsymbol{v}\|_2}{m}\right) \; w.h.p.$$

Proof Sketch

- gradient expression $\nabla_i(t)$: a function of $\|\boldsymbol{w}_i\|, \|\boldsymbol{v}_i\|, \theta_{il}, \varphi_{ij}$
- Phase 1 (amplify alignment)
  - $\langle \boldsymbol{w}_i(t), \nabla_i(t)\rangle \leq 0 \Rightarrow \|\boldsymbol{w}_i(t)\|$ increasing
  - track $\theta_{i^\star}(t)$: linear convergence of $\sin\theta_{i^\star} \Rightarrow \theta_{i^\star}(t) \leq \epsilon_1$
- Phase 2 (norm convergence)
  - norm-balanced

$$\frac{\|\boldsymbol{w}_i(t)\|}{\|\boldsymbol{w}_j(t)\|} = \Theta(1)\,, \forall i,j \in [m],\ \ T_1 \leq t \leq T_2\,.$$

  - track $\theta_{i^\star}(t)$ and obtain $L(\boldsymbol{W}(T_2)) \leq \epsilon_2$
- Phase 3 (local convergence)

  gradient lower bound + local smoothness of the loss L

  $\Rightarrow$ classical optimization results

Proof Sketch

- gradient expression $\nabla_i(t)$: a function of $\|\boldsymbol{w}_i\|, \|\boldsymbol{v}_i\|, \theta_{il}, \varphi_{ij}$
- Phase 1 (amplify alignment)
  - $\langle \boldsymbol{w}_i(t), \nabla_i(t) \rangle \leq 0 \Rightarrow \|\boldsymbol{w}_i(t)\|$ increasing
  - track $\theta_{i^\star}(t)$: linear convergence of $\sin\theta_{i^\star} \Rightarrow \theta_{i^\star}(t) \leq \epsilon_1$
- Phase 2 (norm convergence)
  - norm-balanced

$$\frac{\|\boldsymbol{w}_i(t)\|}{\|\boldsymbol{w}_j(t)\|} = \Theta(1)\,, \forall i,j \in [m], \; T_1 \leq t \leq T_2\,.$$

  - track $\theta_{i^\star}(t)$ and obtain $L(\boldsymbol{W}(T_2)) \leq \epsilon_2$
- Phase 3 (local convergence)

gradient lower bound + local smoothness of the loss L

$\Rightarrow$ classical optimization results

Proof Sketch

- gradient expression $\nabla_i(t)$: a function of $\|\boldsymbol{w}_i\|, \|\boldsymbol{v}_i\|, \theta_{il}, \varphi_{ij}$
- Phase 1 (amplify alignment)
  - $\langle \boldsymbol{w}_i(t), \nabla_i(t) \rangle \leq 0 \Rightarrow \|\boldsymbol{w}_i(t)\|$ increasing
  - track $\theta_{i^\star}(t)$: linear convergence of $\sin\theta_{i^\star} \Rightarrow \theta_{i^\star}(t) \leq \epsilon_1$
- Phase 2 (norm convergence)
  - norm-balanced

$$\frac{\|\boldsymbol{w}_i(t)\|}{\|\boldsymbol{w}_j(t)\|} = \Theta(1)\,, \forall i, j \in [m], \;\; T_1 \leq t \leq T_2\,.$$

  - track $\theta_{i^\star}(t)$ and obtain $L(\boldsymbol{W}(T_2)) \leq \epsilon_2$
- Phase 3 (local convergence)

gradient lower bound + local smoothness of the loss L

$\Rightarrow$ classical optimization results

Proof Sketch

- gradient expression $\nabla_i(t)$: a function of $\|\boldsymbol{w}_i\|, \|\boldsymbol{v}_i\|, \theta_{il}, \varphi_{ij}$
- Phase 1 (amplify alignment)
  - $\langle \boldsymbol{w}_i(t), \nabla_i(t)\rangle \leq 0 \Rightarrow \|\boldsymbol{w}_i(t)\|$ increasing
  - track $\theta_{i^\star}(t)$: linear convergence of $\sin\theta_{i^\star} \Rightarrow \theta_{i^\star}(t) \leq \epsilon_1$
- Phase 2 (norm convergence)
  - norm-balanced
  $$\frac{\|\boldsymbol{w}_i(t)\|}{\|\boldsymbol{w}_j(t)\|} = \Theta(1)\,, \forall i, j \in [m],\ \ T_1 \leq t \leq T_2\,.$$
  - track $\theta_{i^\star}(t)$ and obtain $L(\boldsymbol{W}(T_2)) \leq \epsilon_2$
- Phase 3 (local convergence)

  gradient lower bound + local smoothness of the loss L

  $\Rightarrow$ classical optimization results

Take-away messages

- model size -> size of weights -> path norm -> Barron spaces
- statistical guarantees with improved sample complexity
- computational-statistical gap -> learning with multiple ReLU neurons

- robust overfitting from the perspective of **approximation** [26]

well-separated data + target function is smooth enough + perturbation is small enough

$\Rightarrow$ **Avoid robust overfitting!**

- LoRA with low-rank dynamics: from **statistical** to **computational**

Take-away messages

- model size -> size of weights -> path norm -> Barron spaces
- statistical guarantees with improved sample complexity
- computational-statistical gap -> learning with multiple ReLU neurons

- robust overfitting from the perspective of **approximation** [26]

well-separated data + target function is smooth enough + perturbation is small enough

$\Rightarrow$ **Avoid robust overfitting!**

- LoRA with low-rank dynamics: from **statistical** to **computational**

Take-away messages

- model size -> size of weights -> path norm -> Barron spaces
- statistical guarantees with improved sample complexity
- computational-statistical gap -> learning with multiple ReLU neurons

- robust overfitting from the perspective of **approximation** [26]

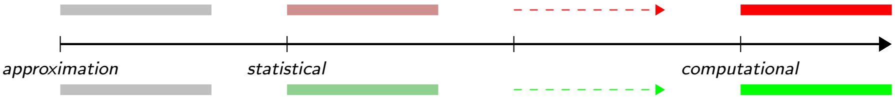
well-separated data + target function is smooth enough + perturbation is small enough

$\Rightarrow$ **Avoid robust overfitting!**

- LoRA with low-rank dynamics: from **statistical** to **computational**

Take-away messages

- model size -> size of weights -> path norm -> Barron spaces
- statistical guarantees with improved sample complexity
- computational-statistical gap -> learning with multiple ReLU neurons



- robust overfitting from the perspective of **approximation** [26]

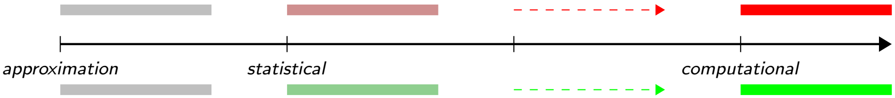
well-separated data + target function is smooth enough + perturbation is small enough

$\Rightarrow$ **Avoid robust overfitting!**

- LoRA with low-rank dynamics: from **statistical** to **computational**

Take-away messages

- model size -> size of weights -> path norm -> Barron spaces
- statistical guarantees with improved sample complexity
- computational-statistical gap -> learning with multiple ReLU neurons



- robust overfitting from the perspective of **approximation** [26]

well-separated data + target function is smooth enough + perturbation is small enough

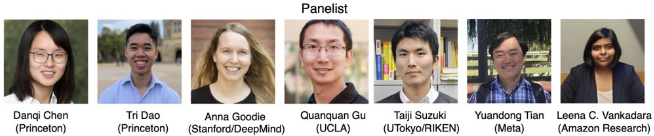
$\Rightarrow$ **Avoid robust overfitting!**

- LoRA with low-rank dynamics: from **statistical** to **computational**

We're organizing Fine-tuning workshop at NeurIPS 2024!

Fine-Tuning in Modern Machine Learning: Principles and Scalability

https://sites.google.com/view/neurips2024-ftw/home



Francis Bach.
Breaking the curse of dimensionality with convex neural networks.
Journal of Machine Learning Research, 18(1):629–681, 2017.

Andrew R Barron.
Universal approximation bounds for superpositions of a sigmoidal function.
IEEE Transactions on Information theory, 39(3):930–945, 1993.

Peter Bartlett.
The sample complexity of pattern classification with neural networks: the size of the weights is more important than the size of the network.
IEEE Transactions on Information Theory, 44(2):525–536, 1998.

Peter Bartlett, Dylan Foster, and Matus Telgarsky.
Spectrally-normalized margin bounds for neural networks.
In *Advances in Neural Information Processing Systems*, pages 6241–6250, 2017.

Mikhail Belkin, Daniel Hsu, Siyuan Ma, and Soumik Mandal.
Reconciling modern machine-learning practice and the classical bias–variance trade-off.
the National Academy of Sciences, 116(32):15849–15854, 2019.

Hongrui Chen, Jihao Long, and Lei Wu.
A duality framework for generalization analysis of random feature models and two-layer neural networks.
arXiv preprint arXiv:2305.05642, 2023.

Sitan Chen and Shyam Narayanan.
A faster and simpler algorithm for learning shallow networks.
arXiv preprint arXiv:2307.12496, 2023.

Zixiang Chen, Yuan Cao, Difan Zou, and Quanquan Gu.
How much over-parameterization is sufficient to learn deep relu networks?
In *International Conference on Learning Representations*, 2020.

Ilias Diakonikolas, Daniel M. Kane, Vasilis Kontonis, and Nikos Zarifis.
Algorithms and SQ lower bounds for PAC learning one-hidden-layer ReLU networks.
In *Thirty Third Conference on Learning Theory*, volume 125, pages 1514–1539. PMLR, 2020.

Carles Domingo-Enrich and Youssef Mroueh.
Tighter sparse approximation bounds for relu neural networks.
In *International Conference on Learning Representations*, 2022.

Weinan E, Chao Ma, and Lei Wu.
A priori estimates of the population risk for two-layer neural networks.
Communications in Mathematical Sciences, 17(5):1407–1425, 2019.

Weinan E, Chao Ma, and Lei Wu.
The barron space and the flow-induced function spaces for neural network models.
Constructive Approximation, pages 1–38, 2021.

Jonathan Frankle and Michael Carbin.
The lottery ticket hypothesis: Finding sparse, trainable neural networks.
In *International Conference on Learning Representations*, 2019.

Yiding Jiang, Behnam Neyshabur, Hossein Mobahi, Dilip Krishnan, and Samy Bengio.
Fantastic generalization measures and where to find them.
In *International Conference on Learning Representations*, 2020.

Jared Kaplan, Sam McCandlish, Tom Henighan, Tom B Brown, Benjamin Chess, Rewon Child, Scott Gray, Alec Radford, Jeffrey Wu, and Dario Amodei.
Scaling laws for neural language models.
arXiv preprint arXiv:2001.08361, 2020.

Tengyuan Liang, Tomaso Poggio, Alexander Rakhlin, and James Stokes.
Fisher-rao metric, geometry, and complexity of neural networks.
In *International conference on Artificial Intelligence and Statistics*, pages 888–896, 2019.

Fanghui Liu, Xiaolin Huang, Yudong Chen, and Johan AK Suykens.
Random features for kernel approximation: A survey on algorithms, theory, and beyond.
IEEE Transactions on Pattern Analysis and Machine Intelligence, 44(10):7128–7148, 2021.

Pavlo Molchanov, Stephen Tyree, Tero Karras, Timo Aila, and Jan Kautz.
Pruning convolutional neural networks for resource efficient inference.
In *International Conference on Learning Representations*, 2017.

Vaishnavh Nagarajan and J Zico Kolter.
Generalization in deep networks: The role of distance from initialization.
arXiv preprint arXiv:1901.01672, 2019.

Preetum Nakkiran, Gal Kaplun, Yamini Bansal, Tristan Yang, Boaz Barak, and Ilya Sutskever.
Deep double descent: Where bigger models and more data hurt.
In *International Conference on Learning Representations*, 2019.

Behnam Neyshabur, Ryota Tomioka, and Nathan Srebro.
Norm-based capacity control in neural networks.
In *Conference on Learning Theory*, pages 1376–1401. PMLR, 2015.

Greg Ongie, Rebecca Willett, Daniel Soudry, and Nathan Srebro.
A function space view of bounded norm infinite width relu nets: The multivariate case.
In *International Conference on Learning Representations*, 2020.

Ali Rahimi and Benjamin Recht.
Random features for large-scale kernel machines.
In *Advances in Neural Information Processing Systems*, pages 1177–1184, 2007.

Ali Rahimi and Benjamin Recht.
Uniform approximation of functions with random bases.
In *Annual Allerton Conference on Communication, Control, and Computing*, pages 555–561. IEEE, 2008.

Pedro Savarese, Itay Evron, Daniel Soudry, and Nathan Srebro.
How do infinite width bounded norm networks look in function space?
In *Conference on Learning Theory*, pages 2667–2690. PMLR, 2019.

Zhongjie Shi, Fanghui Liu, Yuan Cao, and Johan AK Suykens.
Can overfitted deep neural networks in adversarial training generalize?–an approximation viewpoint.
arXiv preprint arXiv:2401.13624, 2024.

References x

Jonathan W Siegel and Jinchao Xu.
Sharp bounds on the approximation rates, metric entropy, and n-widths of shallow neural networks.
arXiv preprint arXiv:2101.12365, 2021.

Taiji Suzuki, Denny Wu, Kazusato Oko, and Atsushi Nitanda.
Feature learning via mean-field langevin dynamics: classifying sparse parities and beyond.
In *Advances in Neural Information Processing Systems*, 2023.

Aad W Van Der Vaart, Adrianus Willem van der Vaart, Aad van der Vaart, and Jon Wellner.
***Weak convergence and empirical processes: with applications to statistics*.**
Springer Science & Business Media, 1996.

Lei Wu and Jihao Long.
A spectral-based analysis of the separation between two-layer neural networks and linear methods.
Journal of Machine Learning Research, 119:1–34, 2022.

Jianyu Zhang and Léon Bottou.
Fine-tuning with very large dropout.
arXiv preprint arXiv:2403.00946, 2024.

*Separation between robust and clean generalization

	#parameters	Upper bound
standard generalization	$\mathcal{O}(nd)$	$\widetilde{\mathcal{O}}\left(n^{-\frac{2\alpha}{2\alpha+d}}\right)$
robust generalization	$\mathcal{O}\left(nd + \delta^{-\frac{d}{2\alpha-2}} \log \frac{1}{\delta}\right)$	$\mathcal{O}(\sqrt{d}\delta)$

- more smooth, less #params
- Examples: $\delta < n^{-\frac{1}{d-1}}$
  - $\delta = \frac{1}{n}$: robust overfitting?

well-separated data + target function is smooth enough + perturbation is small enough

$\Rightarrow$ **Avoid robust overfitting!**

*Separation between robust and clean generalization

	#parameters	Upper bound
standard generalization	$\mathcal{O}(nd)$	$\widetilde{\mathcal{O}}\left(n^{-\frac{2\alpha}{2\alpha+d}}\right)$
robust generalization	$\mathcal{O}\left(nd + \delta^{-\frac{d}{2\alpha-2}} \log \frac{1}{\delta}\right)$	$\mathcal{O}(\sqrt{d}\delta)$

- more smooth, less #params
- Examples: $\delta < n^{-\frac{1}{d-1}}$
  - $\delta = \frac{1}{n}$: robust overfitting?

well-separated data + target function is smooth enough + perturbation is small enough

$\Rightarrow$ **Avoid robust overfitting!**