The Computational Advantage of Depth: Learning High-Dimensional Hierarchical Functions with Gradient Descent

The Computational Advantage of Depth: Learning High-Dimensional Hierarchical Functions with Gradient Descent

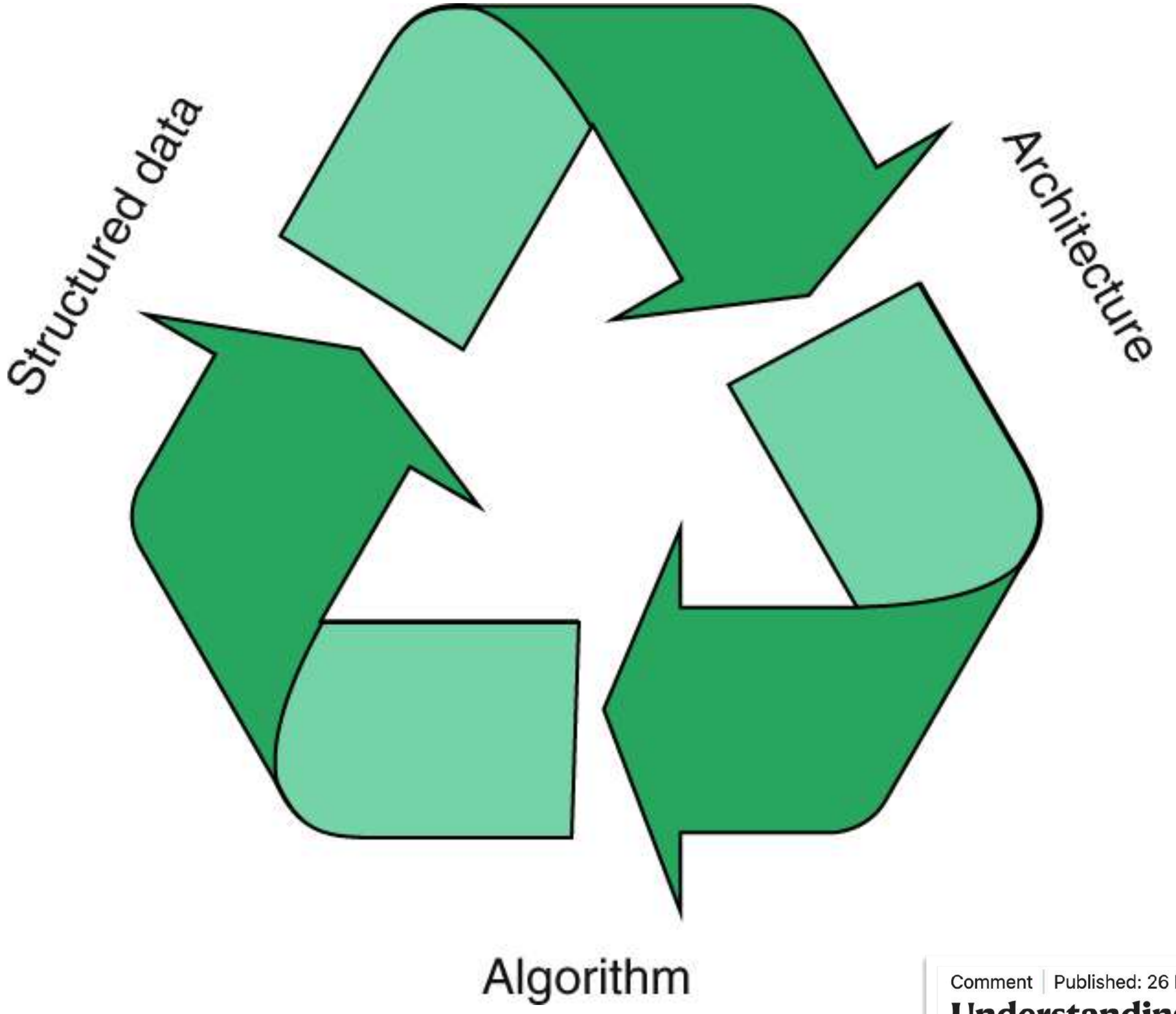
Yatin Dandi[1,2], Luca Pesce[1], Lenka Zdeborová[2], and Florent Krzakala[1]

[1]Ecole Polytechnique Fédérale de Lausanne, Information, Learning and Physics Laboratory. CH-1015 Lausanne, Switzerland.
[2]Ecole Polytechnique Fédérale de Lausanne, Statistical Physics of Computation Laboratory. CH-1015 Lausanne, Switzerland.

IdePHIX
INFORMATION, LEARNING & PHYSICS LAB.

SPOC
Statistical Physics Of Computation

Understanding deep learning is also a job*



Comment | Published: 26 May 2020

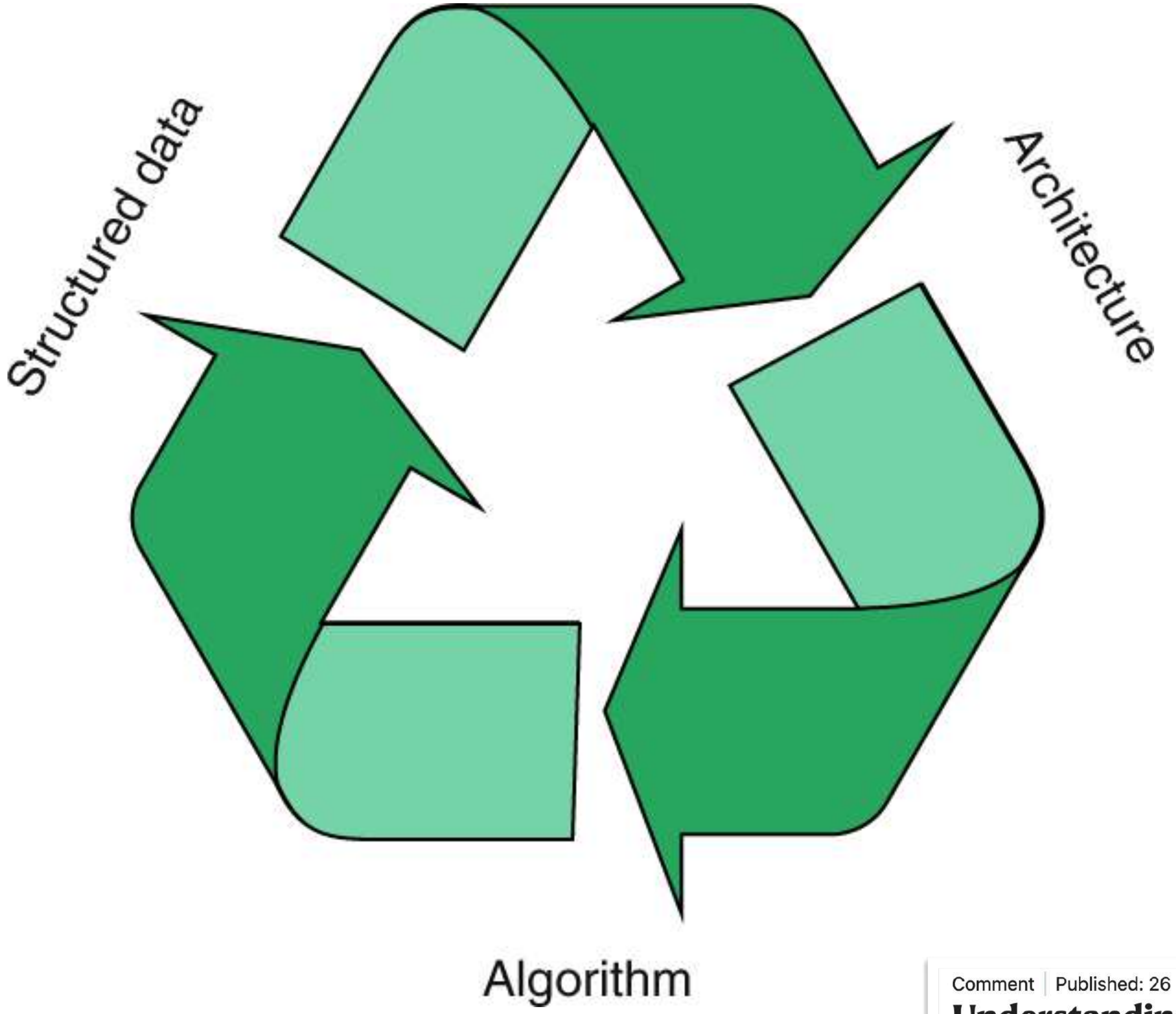
Understanding deep learning is also a job for physicists

Lenka Zdeborová

Nature Physics **16**, 602–604 (2020) | Cite this article

Understanding deep learning is also a job*

*Till AGI



Comment | Published: 26 May 2020

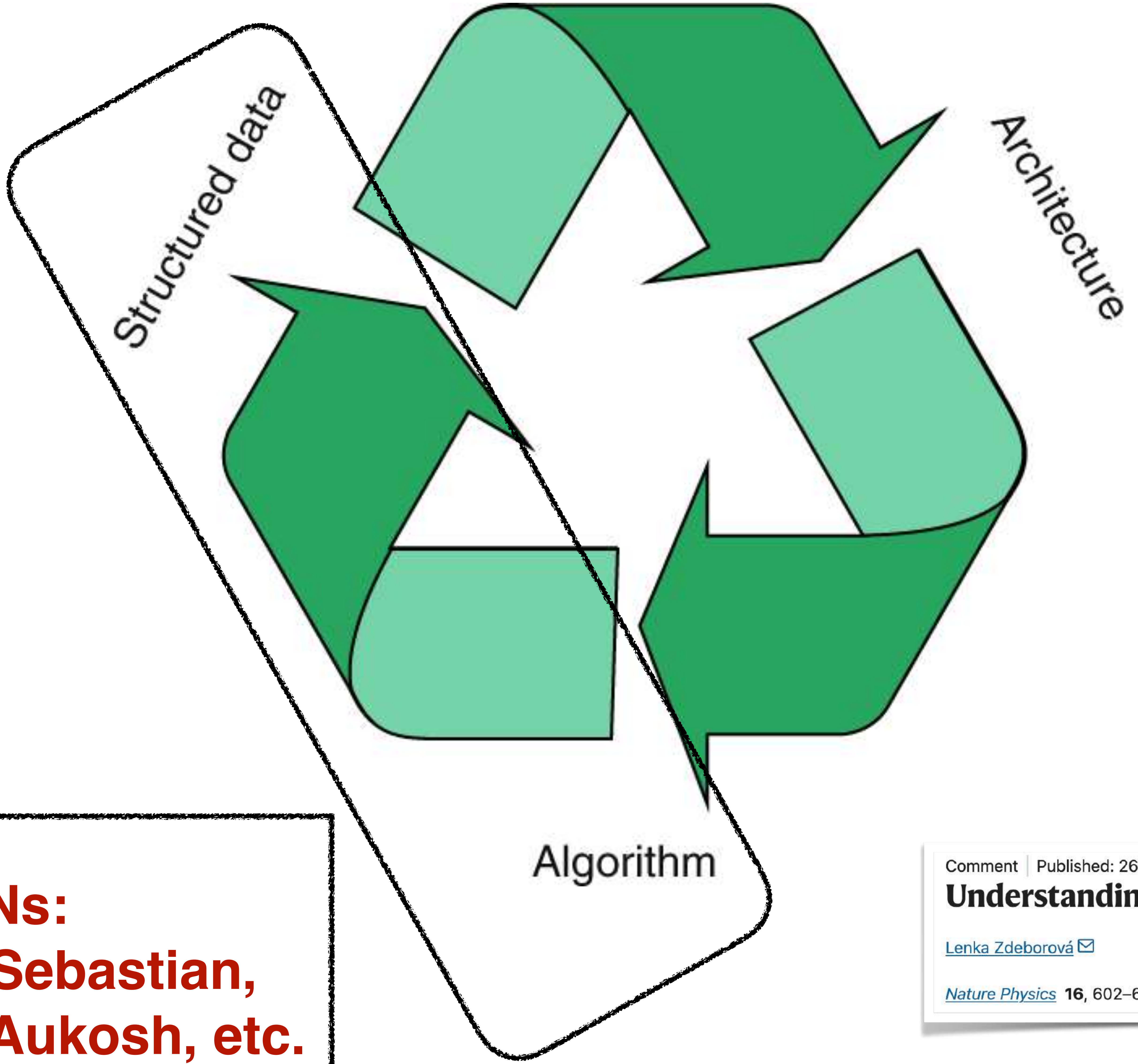
Understanding deep learning is also a job for physicists

Lenka Zdeborová

Nature Physics **16**, 602–604 (2020) | Cite this article

Understanding deep learning is also a job*

*Till AGI



2-layer NNs:
Bruno’s lecture, Sebastian, Joan, Inbar, Alex, Aukosh, etc.


Comment | Published: 26 May 2020

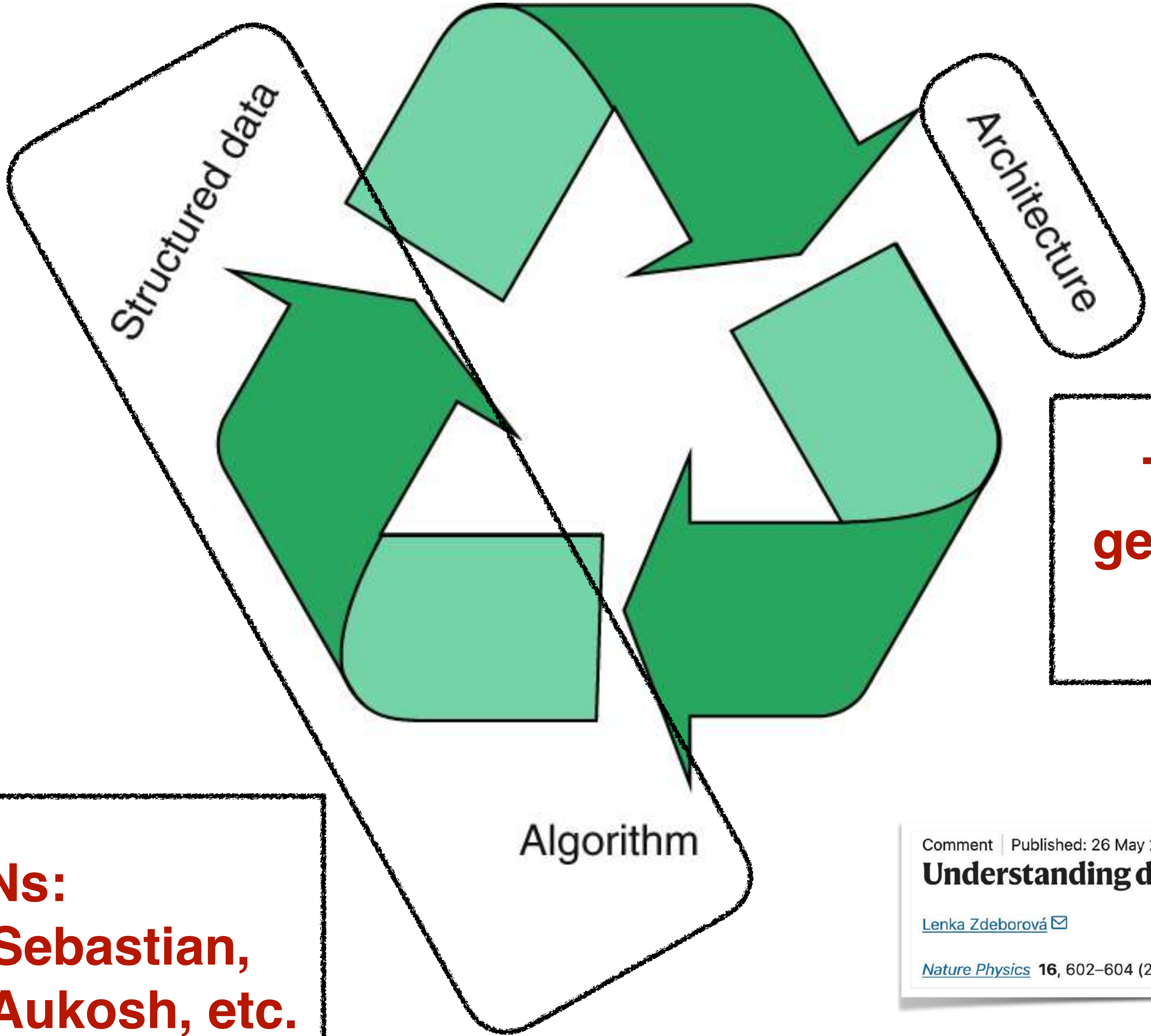
Understanding deep learning is also a job for physicists

Lenka Zdeborová

Nature Physics **16**, 602–604 (2020) | Cite this article

Understanding deep learning is also a job*

*Till AGI



Transformers, CNNs, geometric deep learning, etc.

2-layer NNs:
Bruno’s lecture, Sebastian, Joan, Inbar, Alex, Aukosh, etc.


Comment | Published: 26 May 2020

Understanding deep learning is also a job for physicists

Lenka Zdeborová

Nature Physics **16**, 602–604 (2020) | Cite this article

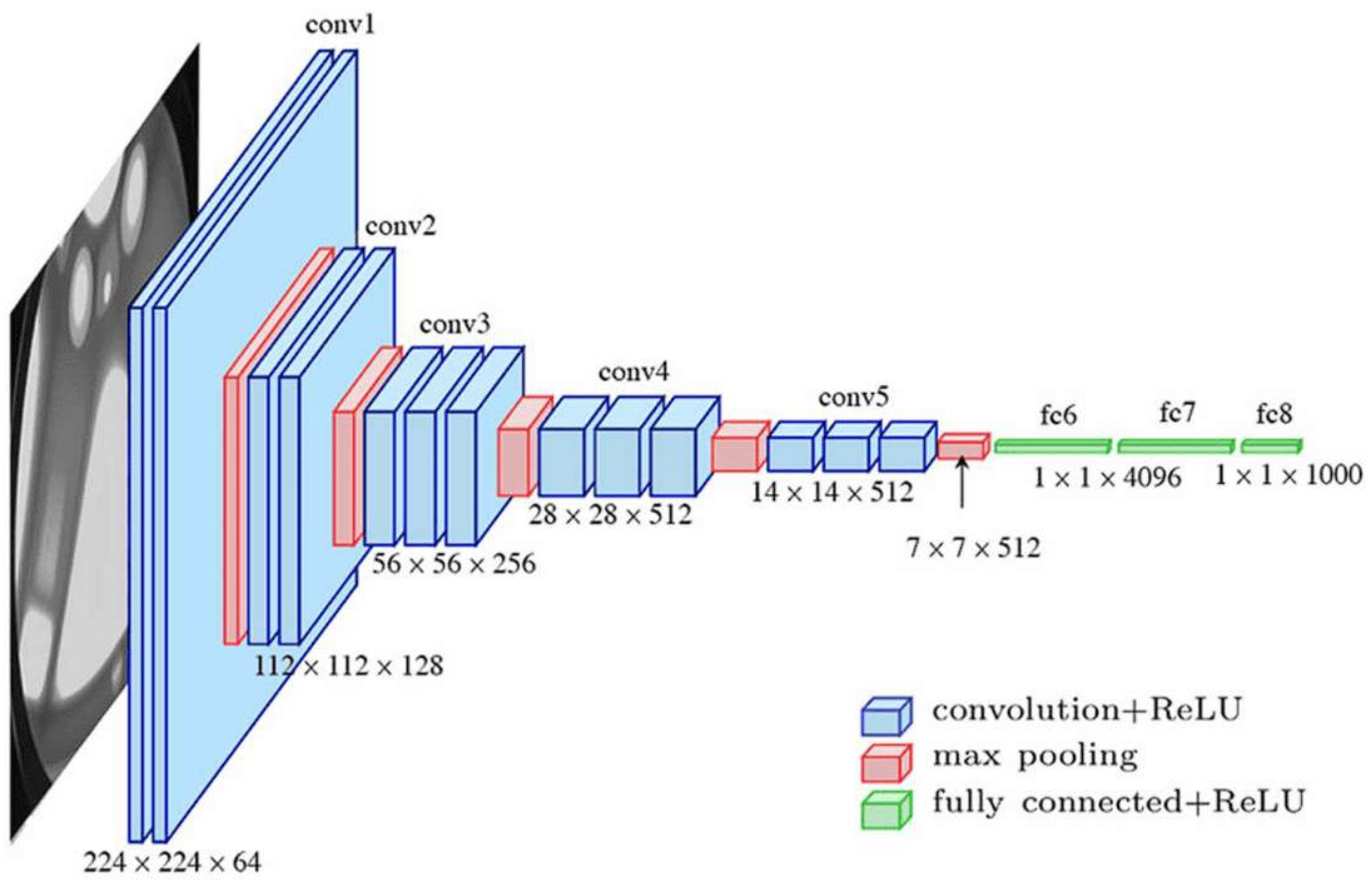
It's called *deep* learning for a reason.

It’s called *deep* learning for a reason.

VGG

VERY DEEP CONVOLUTIONAL NETWORKS FOR LARGE-SCALE IMAGE RECOGNITION

Karen Simonyan* **& Andrew Zisserman**+
Visual Geometry Group, Department of Engineering Science, University of Oxford
{karen,az}@robots.ox.ac.uk



Resnet

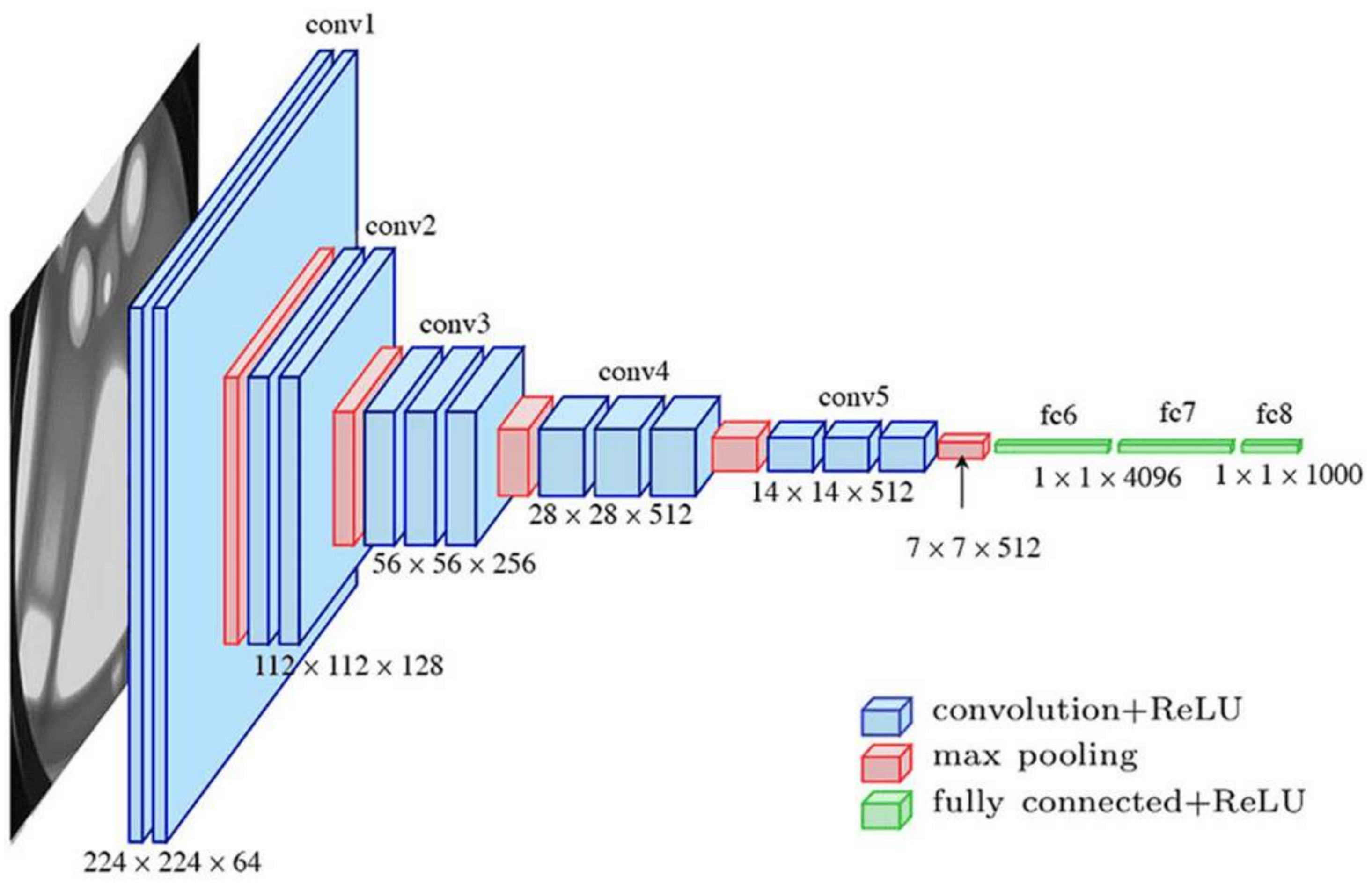
by the number of stacked layers (depth). Recent evidence [41, 44] reveals that network depth is of crucial importance, and the leading results [41, 44, 13, 16] on the challenging ImageNet dataset [36] all exploit “very deep” [41] models, with a depth of sixteen [41] to thirty [16]. Many other non-

It's called *deep* learning for a reason.

VGG

VERY DEEP CONVOLUTIONAL NETWORKS FOR LARGE-SCALE IMAGE RECOGNITION

Karen Simonyan* & Andrew Zisserman+
Visual Geometry Group, Department of Engineering Science, University of Oxford
{karen,az}@robots.ox.ac.uk



Resnet

by the number of stacked layers (depth). Recent evidence [41, 44] reveals that network depth is of crucial importance, and the leading results [41, 44, 13, 16] on the challenging ImageNet dataset [36] all exploit "very deep" [41] models, with a depth of sixteen [41] to thirty [16]. Many other non-

Exponential gains in approximation capacity

The Power of Depth for Feedforward Neural Networks

Ronen Eldan
Weizmann Institute of Science
ronen.eldan@weizmann.ac.il

Ohad Shamir
Weizmann Institute of Science
ohad.shamir@weizmann.ac.il

Abstract

We show that there is a simple (approximately radial) function on $\mathbb{R}^d$, expressible by a small 3-layer feedforward neural networks, which cannot be approximated by any 2-layer network, to more than a certain constant accuracy, unless its width is exponential in the dimension. The result holds for virtually

Deep vs. Shallow Networks: an Approximation Theory Perspective

by

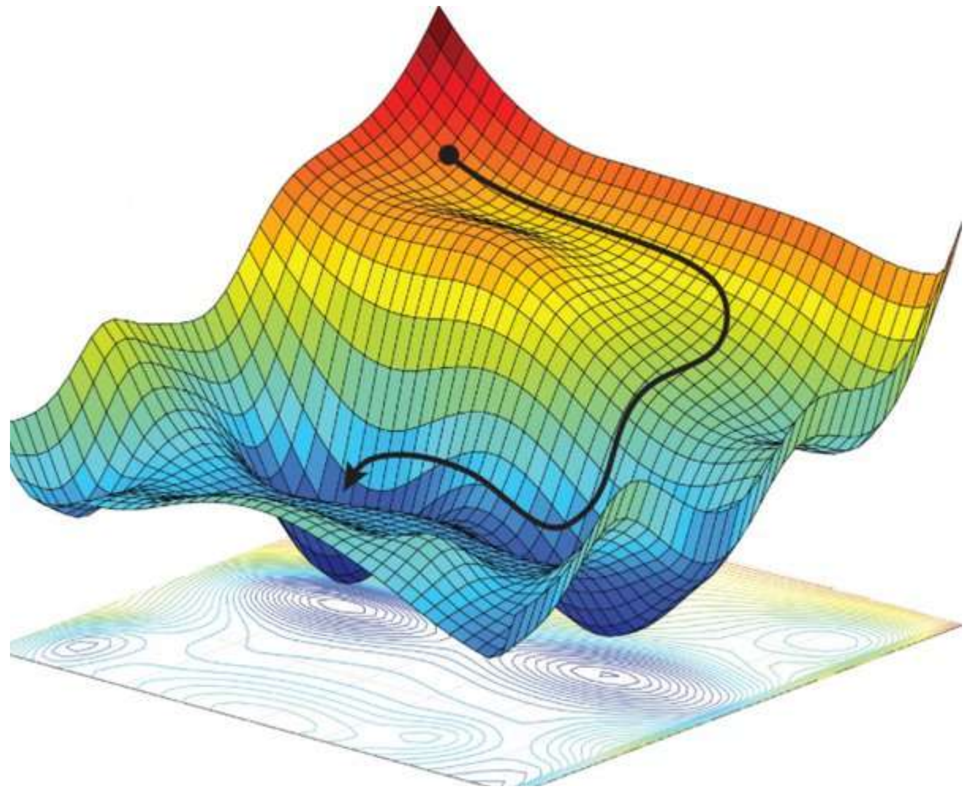
Hrushikesh N. Mhaskar[1] and Tomaso Poggio[2]
1. Department of Mathematics, California Institute of Technology, Pasadena, CA 91125
Institute of Mathematical Sciences, Claremont Graduate University, Claremont, CA 91711.
hrushikesh.mhaskar@cgu.edu
2. Center for Brains, Minds, and Machines, McGovern Institute for Brain Research, Massachusetts Institute of Technology, Cambridge, MA, 02139.
tp@mit.edu

What do we expect to show?

What do we expect to show?

The quest for adaptivity

Posted on June 17, 2021 by Francis Bach

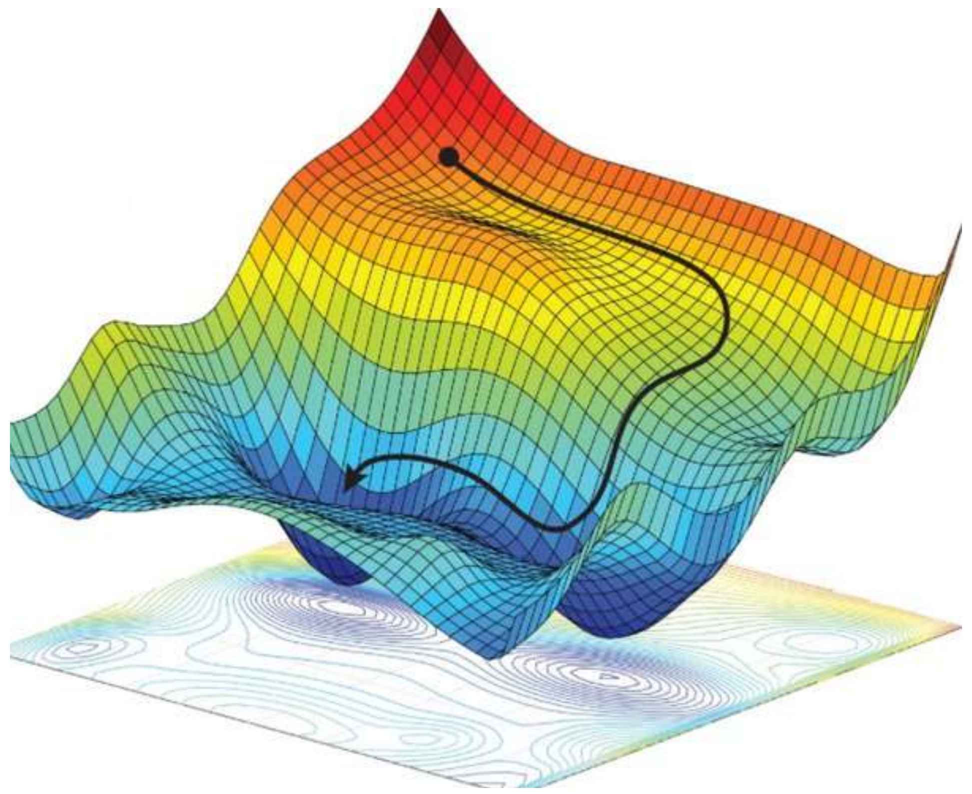


What do we expect to show?

The quest for adaptivity

Posted on June 17, 2021 by Francis Bach

Low-dimensional structure

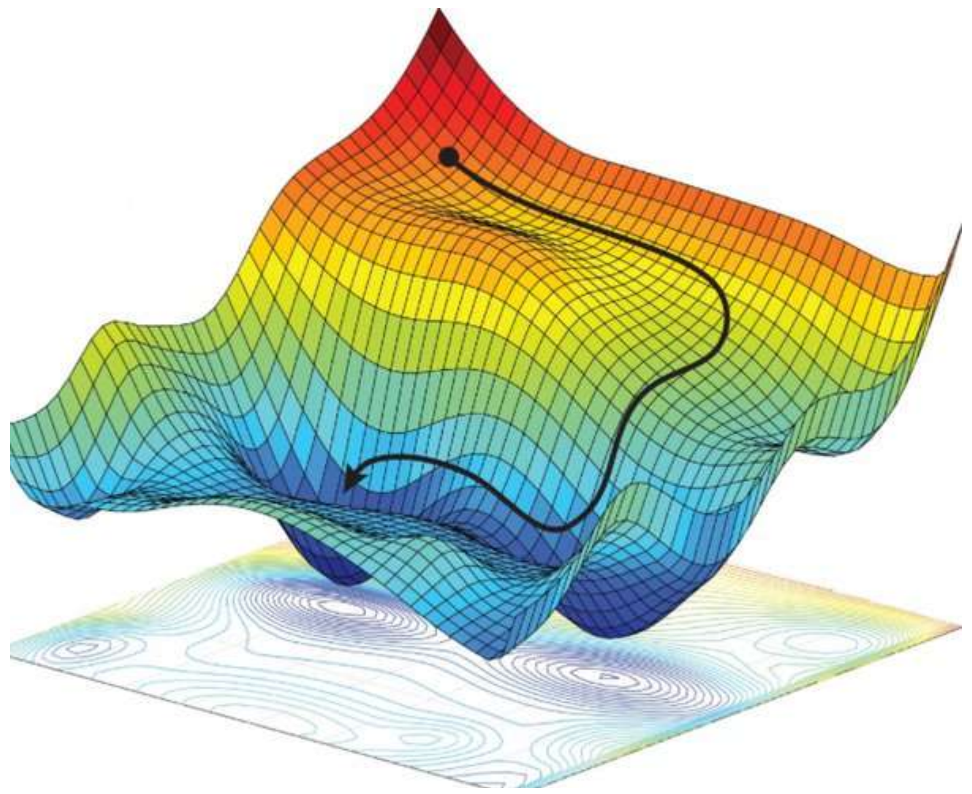


What do we expect to show?

The quest for adaptivity

Posted on June 17, 2021 by Francis Bach

Low-dimensional structure



What do we expect to show?

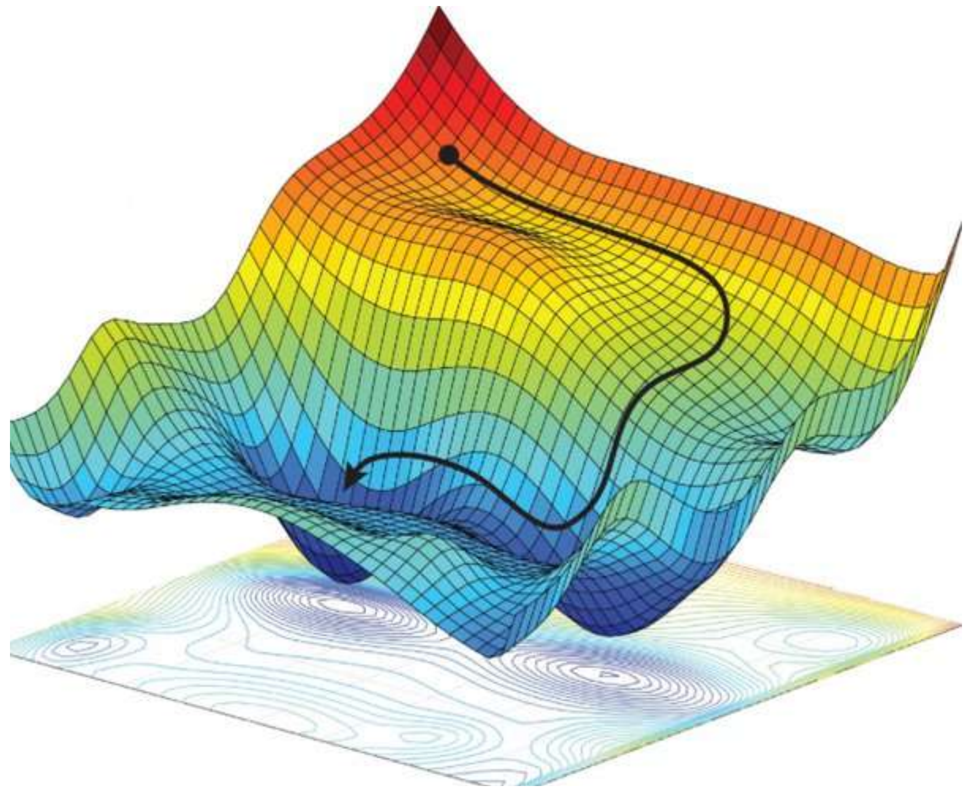
The quest for adaptivity

Posted on June 17, 2021 by Francis Bach

Low-dimensional structure

↓

Lower-sample Complexity with adaptive algorithms



What do we expect to show?

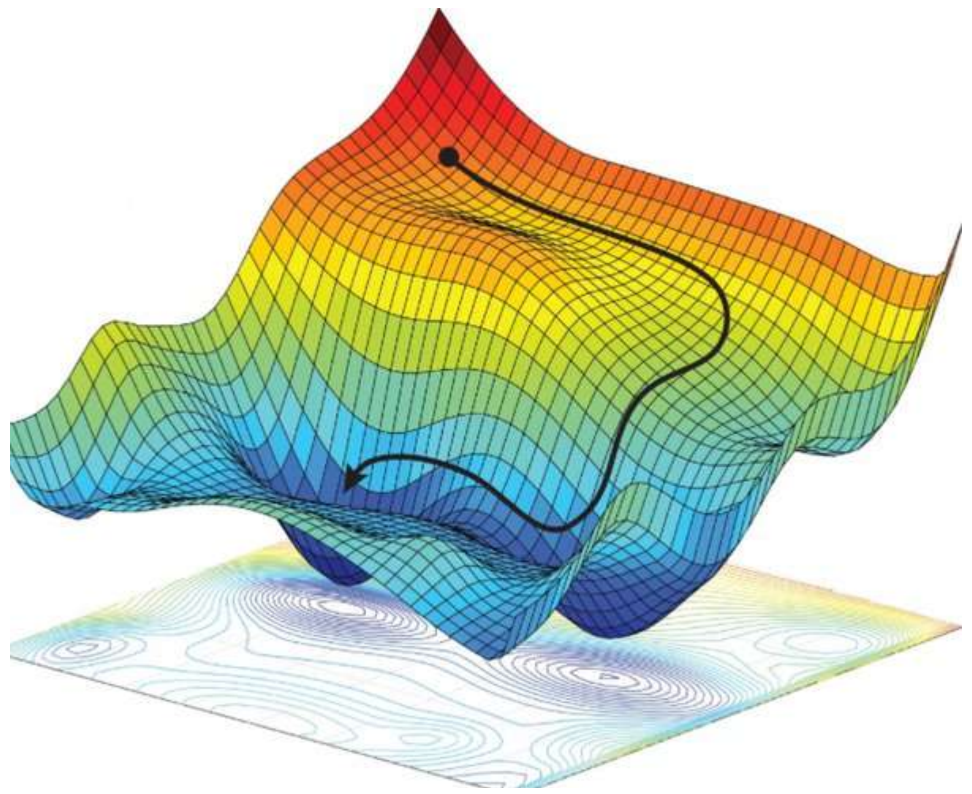
The quest for adaptivity

Posted on June 17, 2021 by Francis Bach

Low-dimensional structure

↓

Lower-sample Complexity with adaptive algorithms

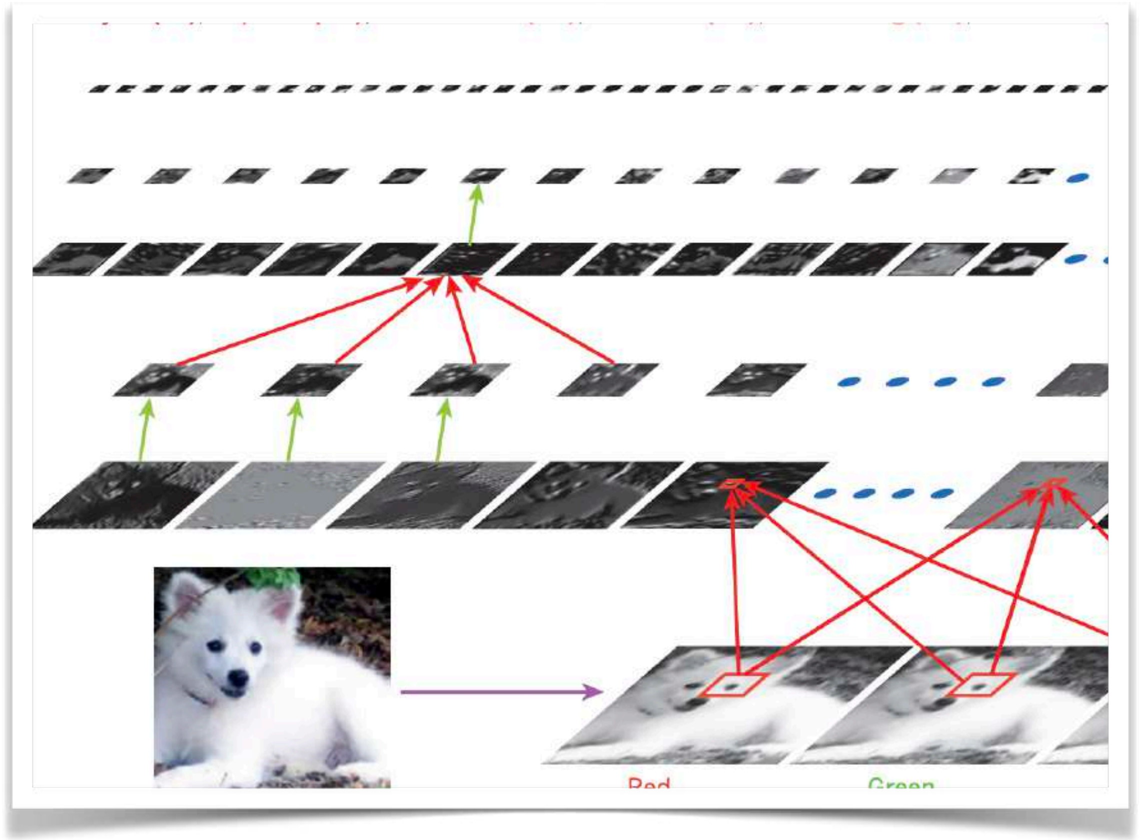


Gaussian data+ single, multi-index models

What do we expect to show?

The quest for adaptivity

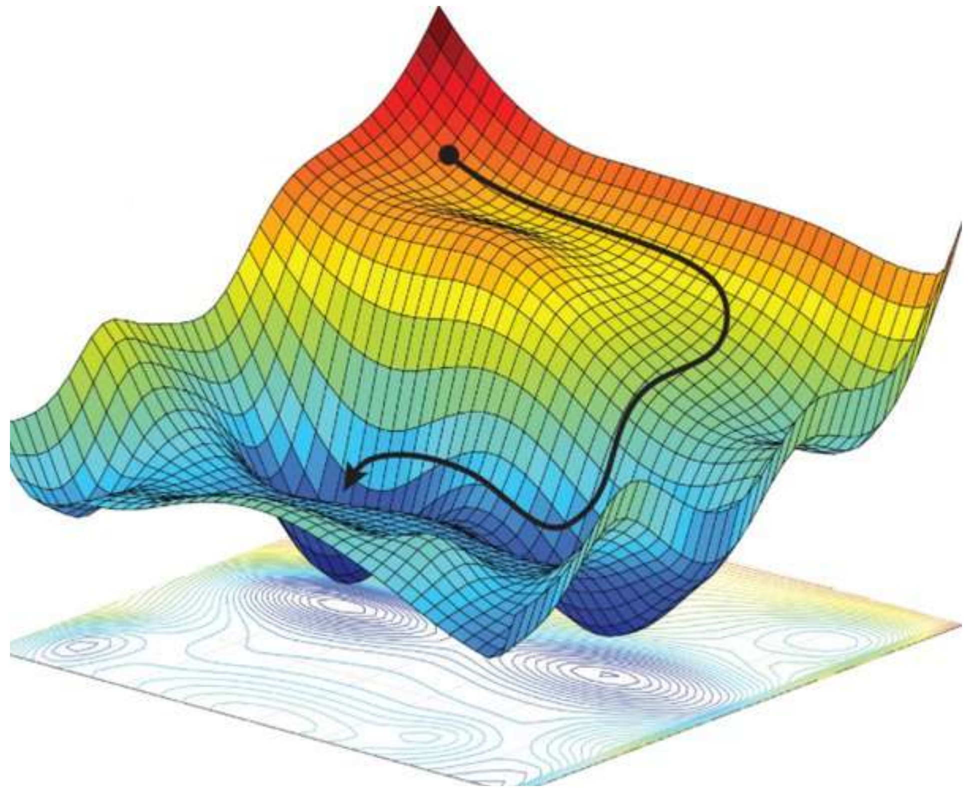
Posted on June 17, 2021 by Francis Bach



Low-dimensional structure

↓

Lower-sample Complexity with adaptive algorithms



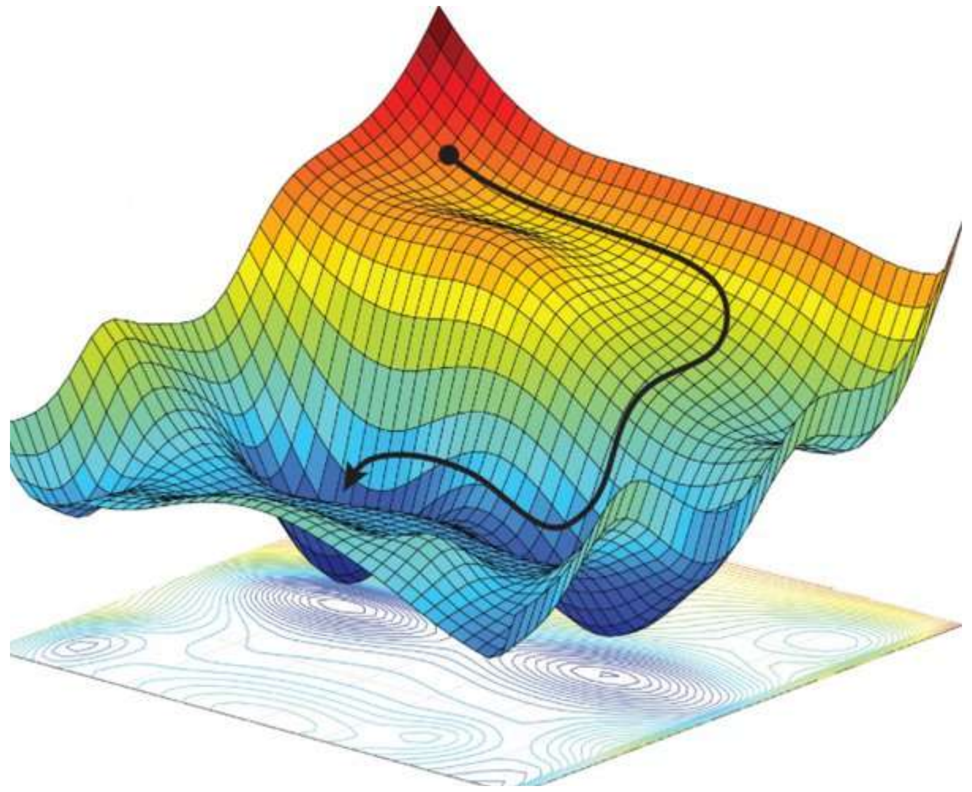
Hierarchical structure

Gaussian data+ single, multi-index models

What do we expect to show?

The quest for adaptivity

Posted on June 17, 2021 by Francis Bach

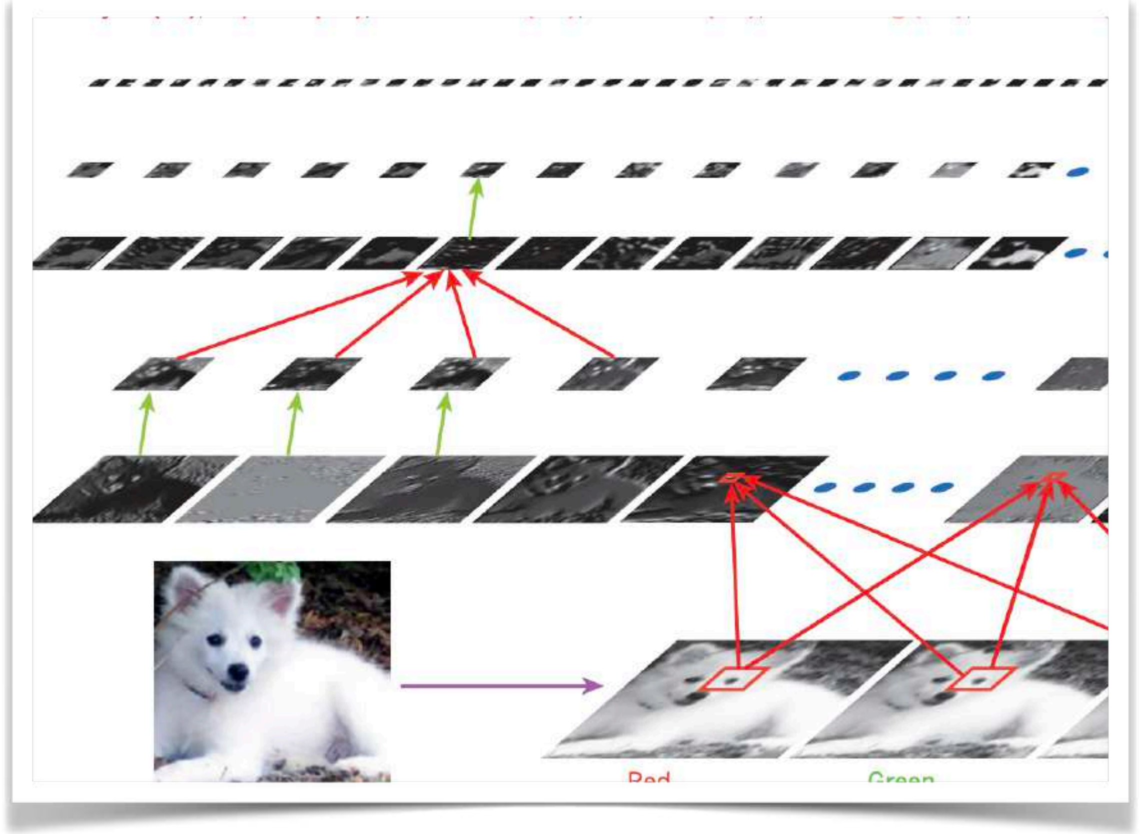


Low-dimensional structure

↓

Lower-sample Complexity with adaptive algorithms

Gaussian data+ single, multi-index models



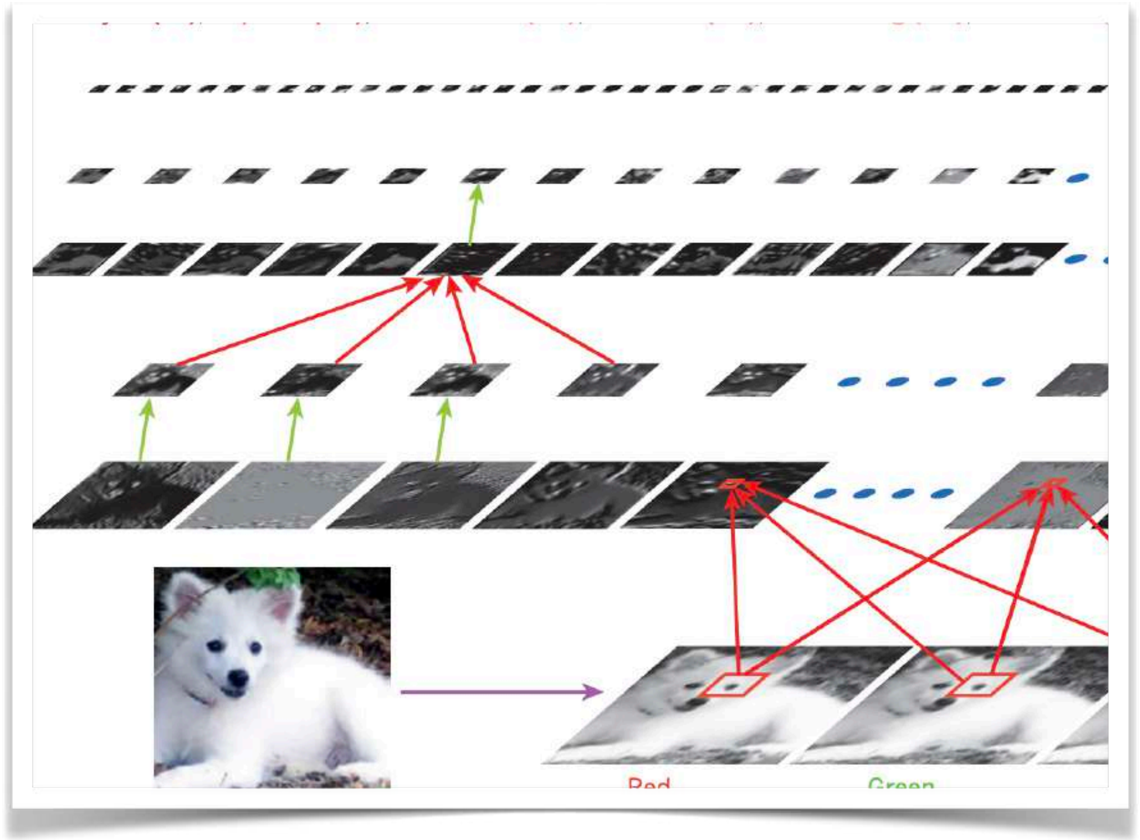
Hierarchical structure

↓

What do we expect to show?

The quest for adaptivity

Posted on June 17, 2021 by Francis Bach

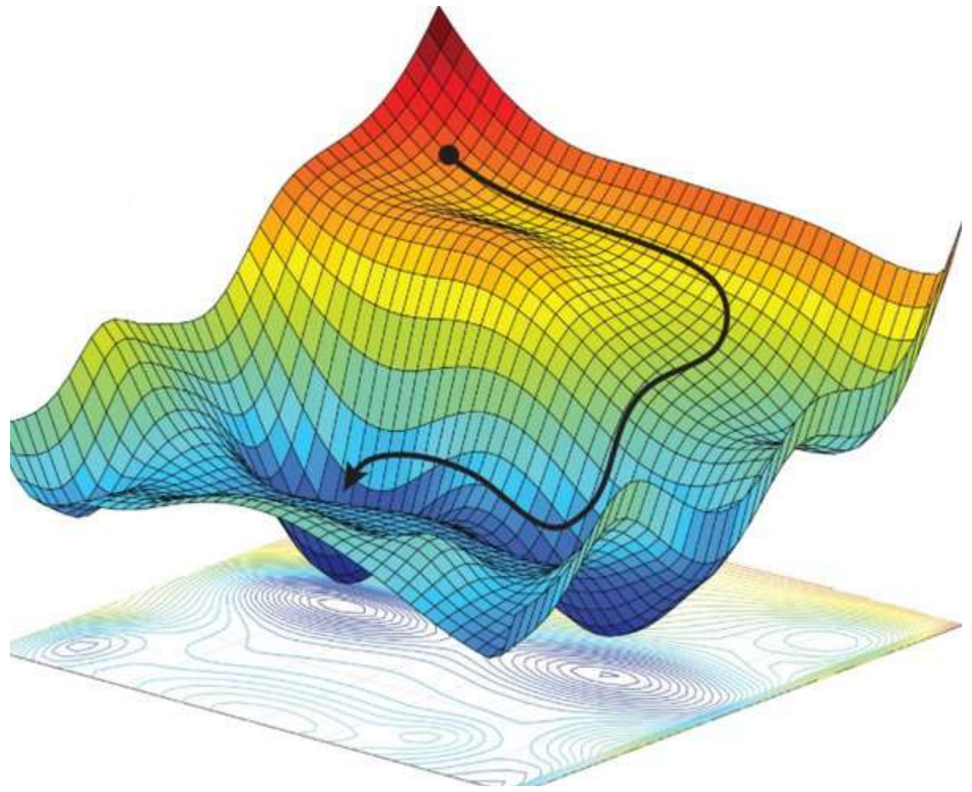


Low-dimensional structure

↓

Lower-sample Complexity with adaptive algorithms

Gaussian data+ single, multi-index models



Hierarchical structure

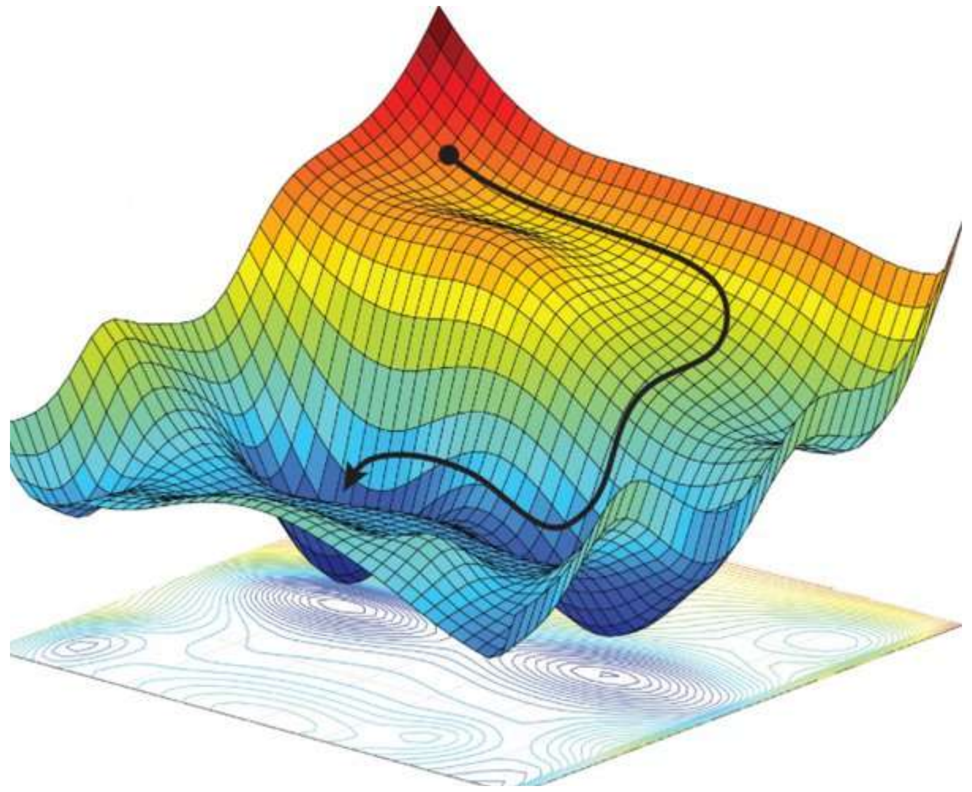
↓

Lower-sample Complexity With deep/multiple levels of adaptation.

What do we expect to show?

The quest for adaptivity

Posted on June 17, 2021 by Francis Bach

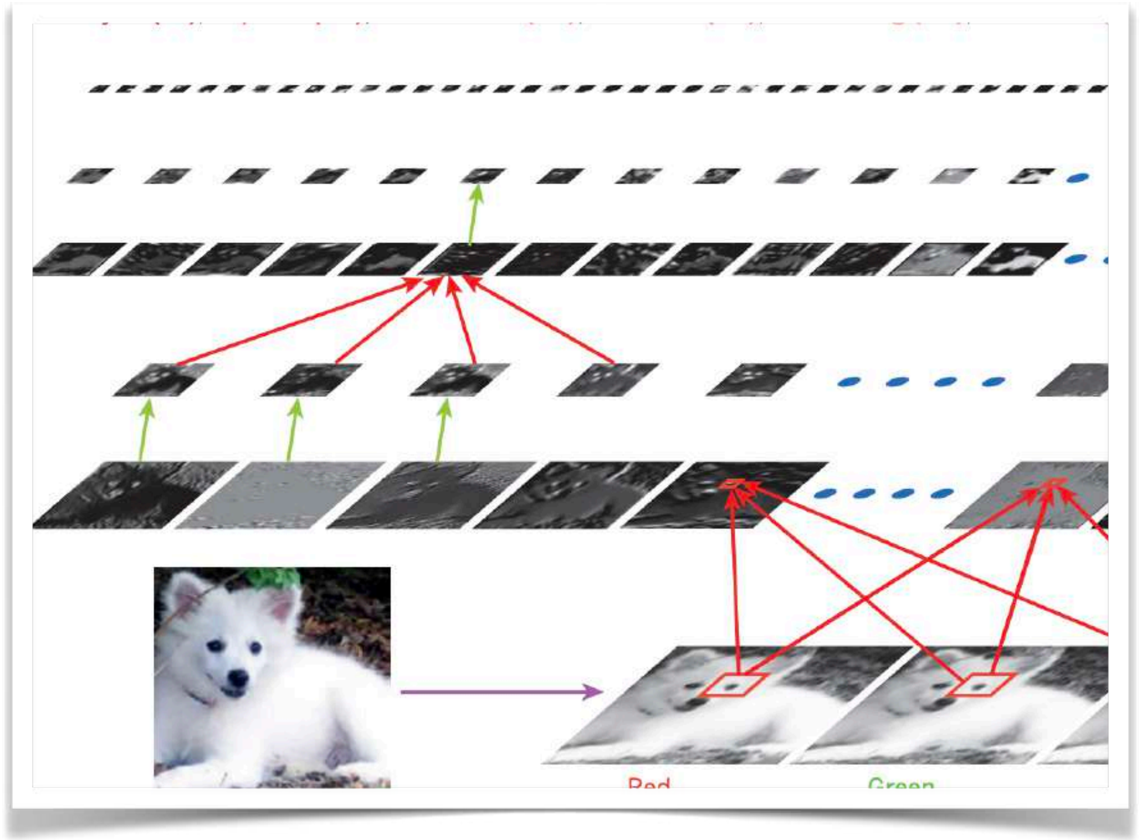


Low-dimensional structure

↓

Lower-sample Complexity with adaptive algorithms

Gaussian data+ single, multi-index models



Hierarchical structure

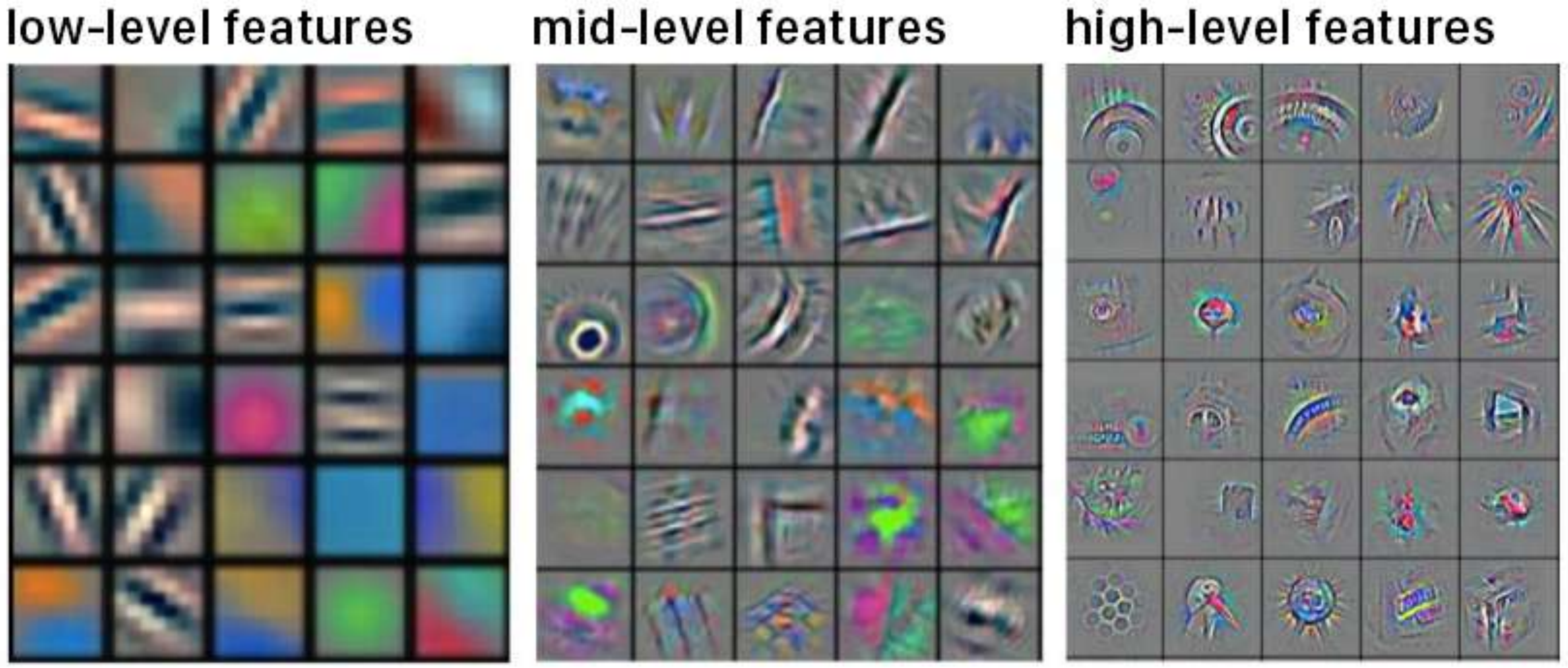
↓

Lower-sample Complexity With deep/multiple levels of adaptation.

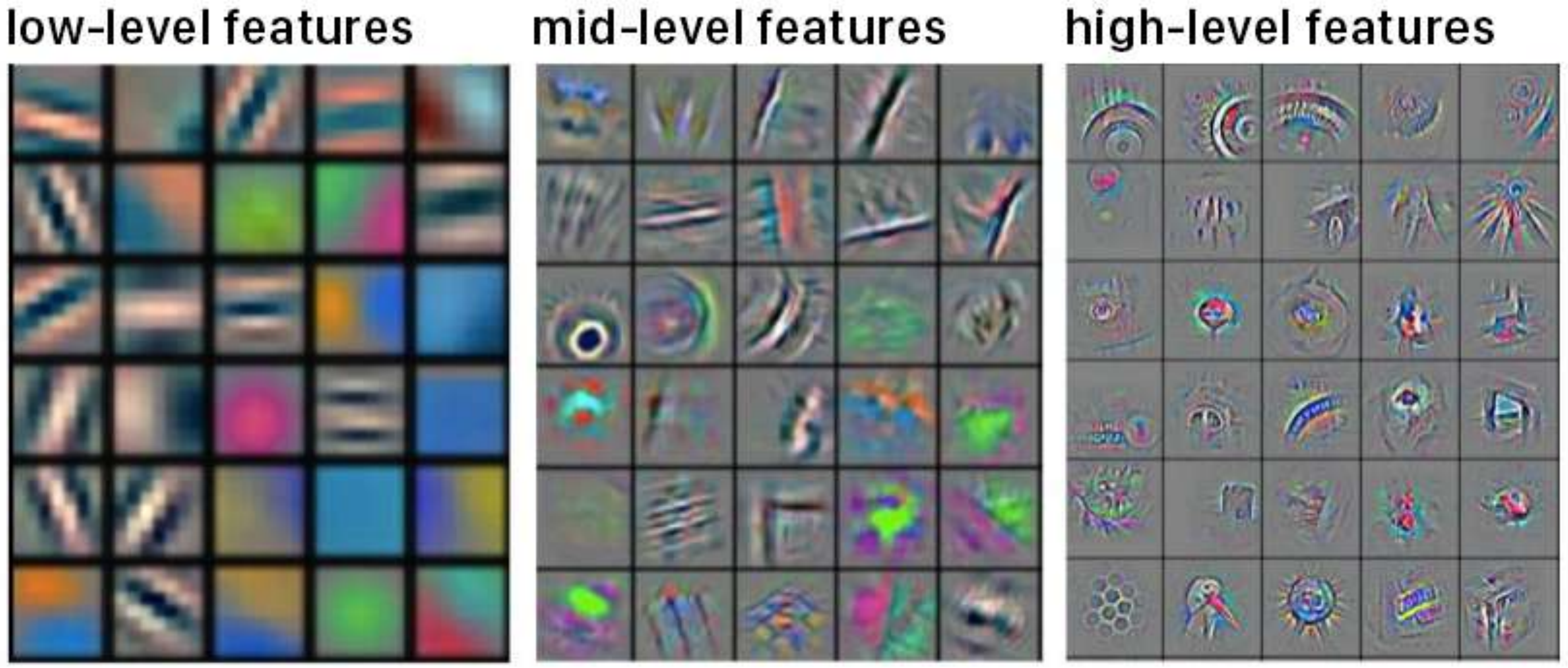
?

What’s “hierarchical”?

What's "hierarchical"?



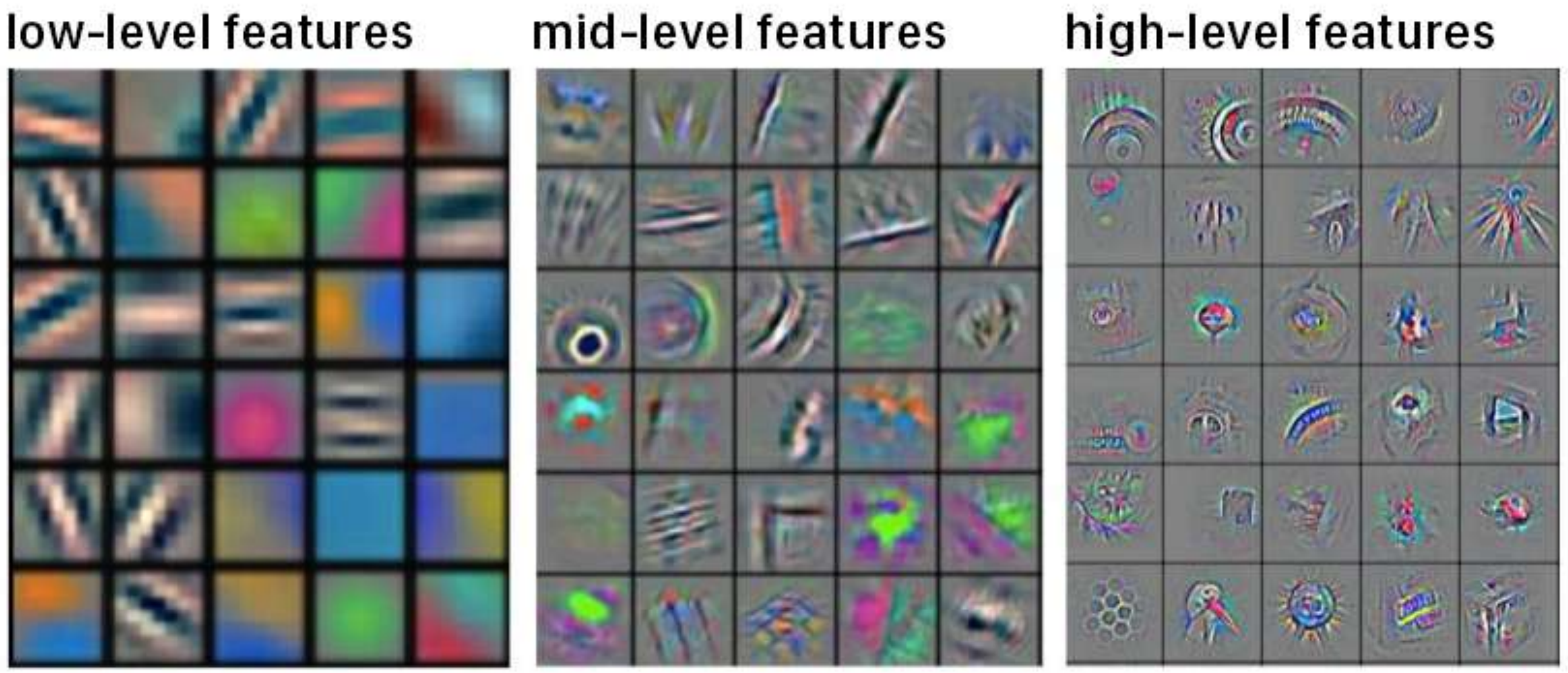
What's "hierarchical"?



Large search space for target

Large effective dimension

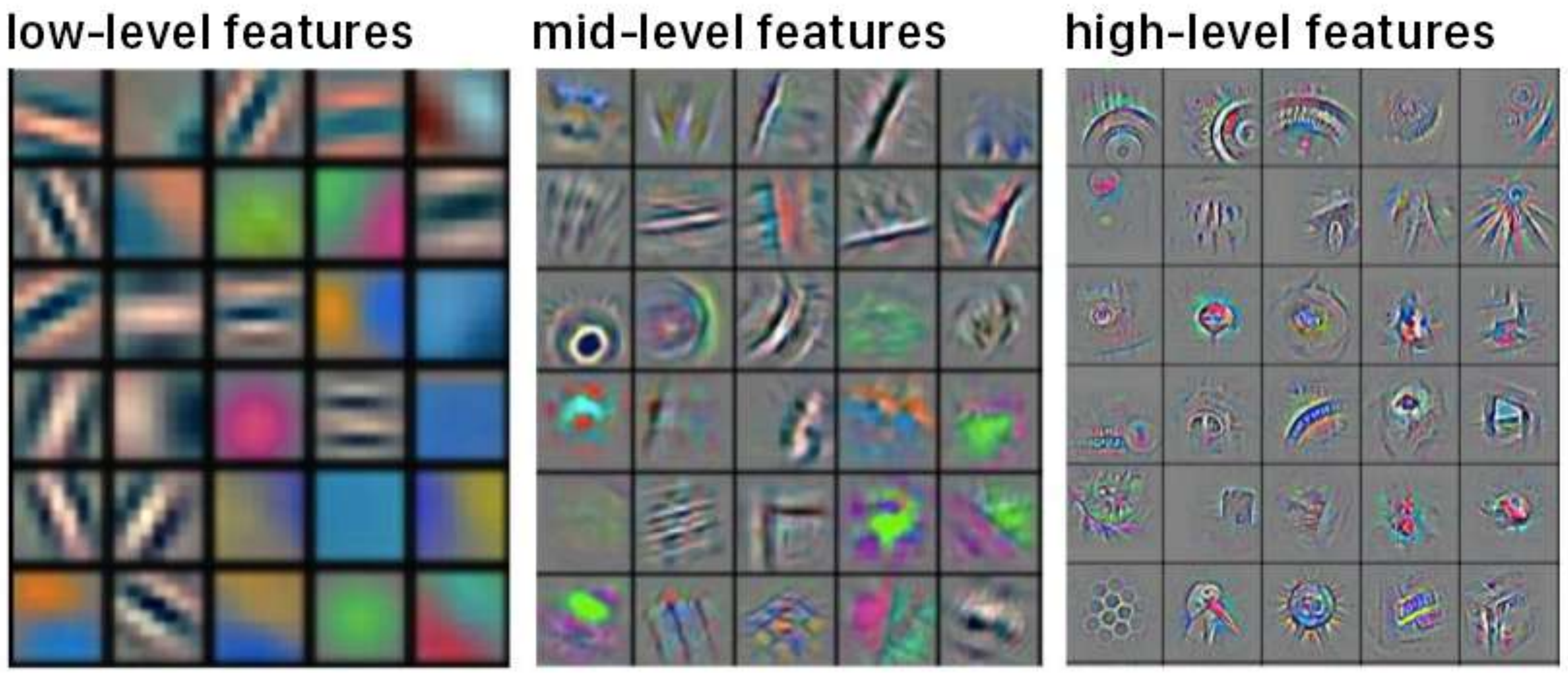
What's "hierarchical"?



Large search space for target

Large effective dimension

What's "hierarchical"?



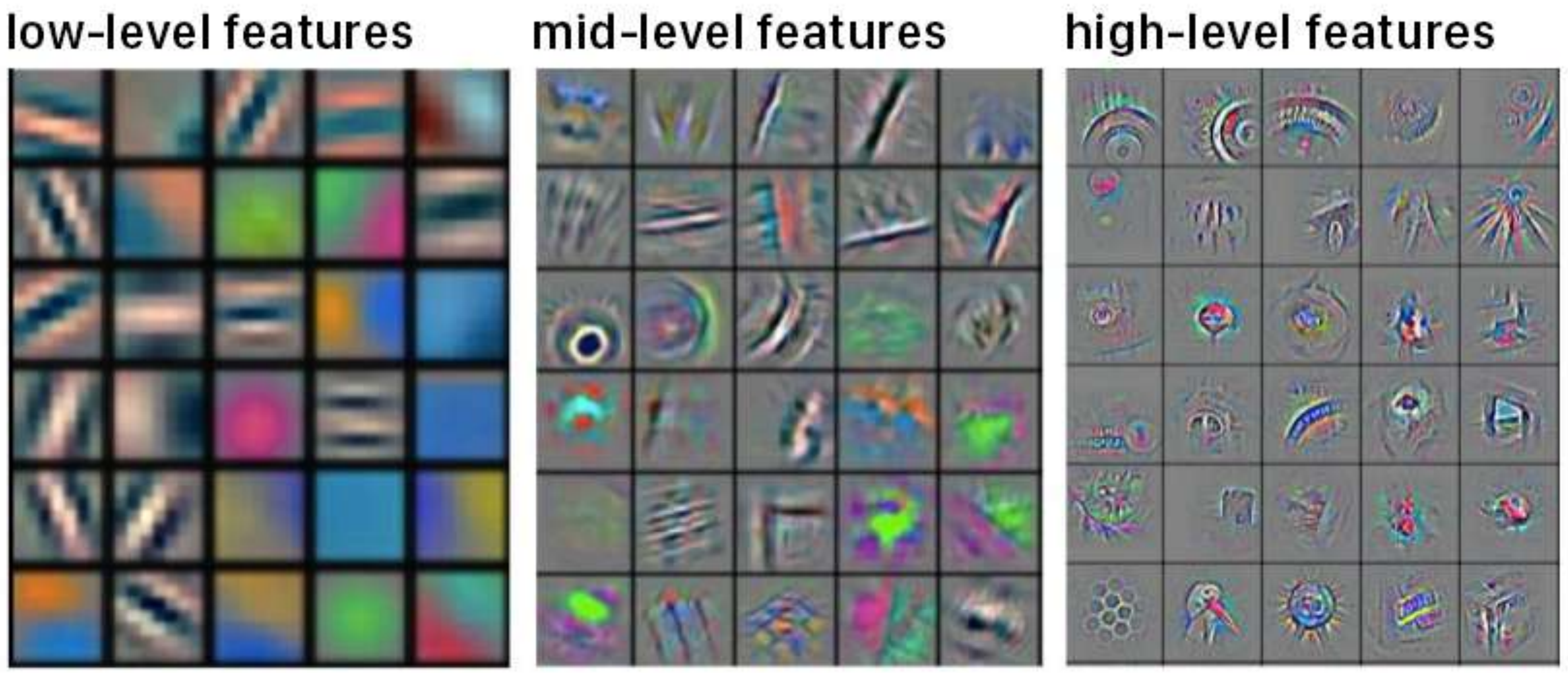
Large search space for target

Small search space for target

Large effective dimension

small effective dimension

What's "hierarchical"?



Large search space for target

Small search space for target

Large effective dimension

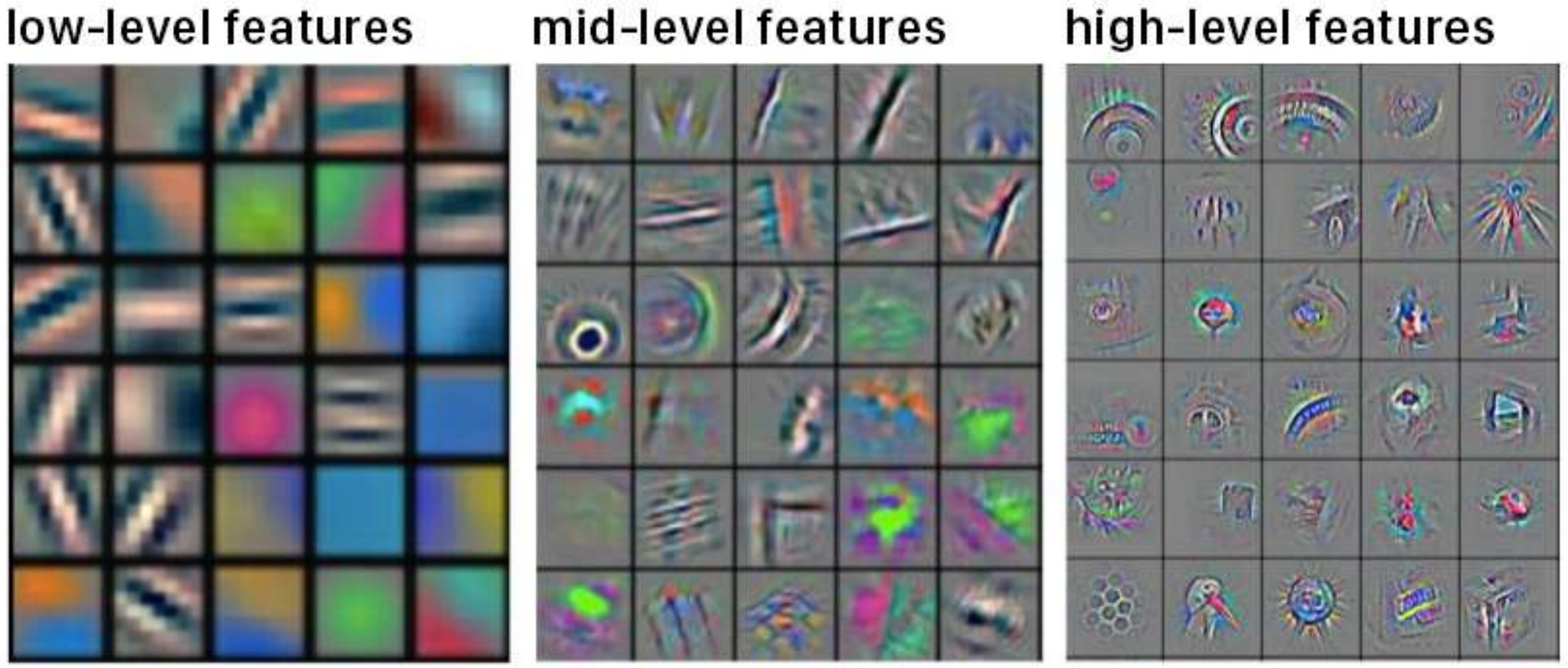
Filtering of features of increasing complexity

small effective dimension

Akin to Coarse-graining/Renormalization (Gulio's talk)

What’s “hierarchical”?

See also Cagnetta et al. 2025, Mossel et al., 2016, Bruna & Mallat 2013



Large search space for target

Small search space for target

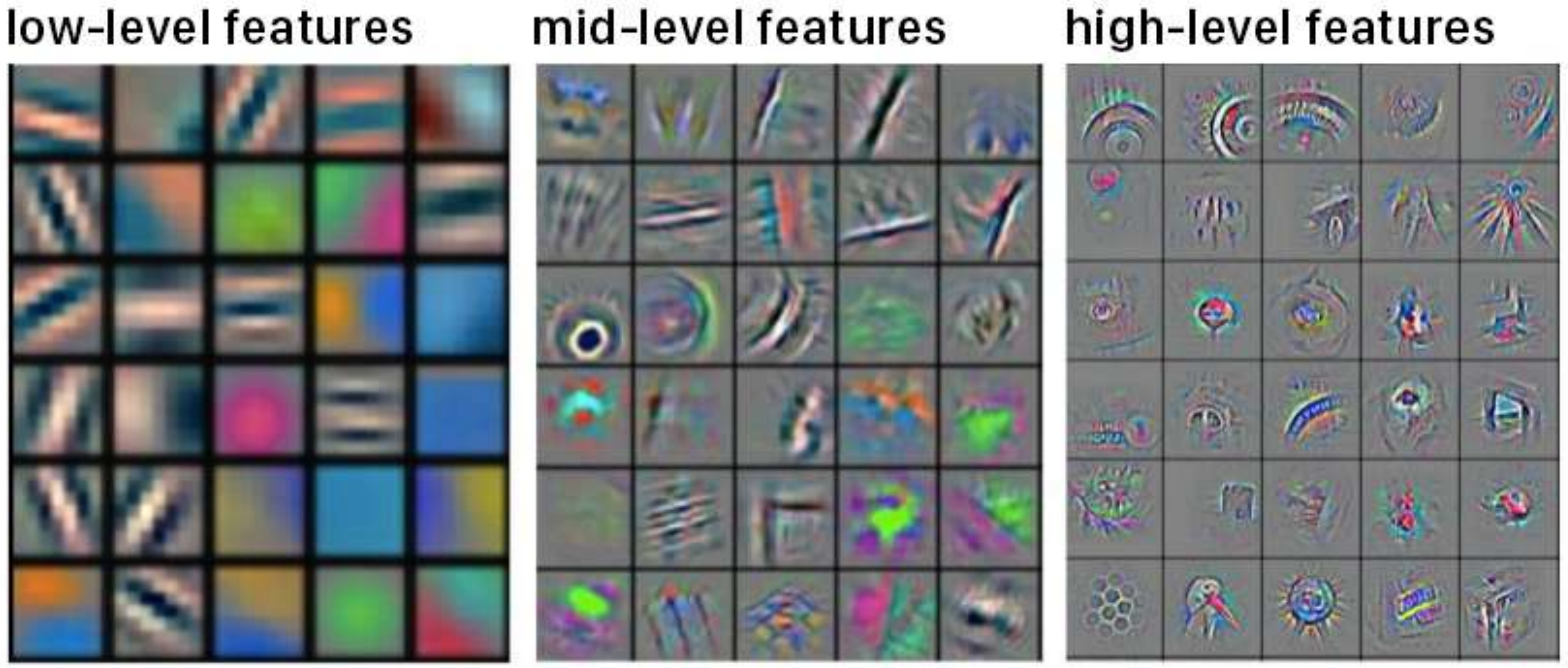
Large effective dimension

Filtering of features of increasing complexity

small effective dimension

Akin to Coarse-graining/Renormalization (Gulio’s talk)

What’s “hierarchical”?



See also Cagnetta et al. 2025, Mossel et al., 2016, Bruna & Mallat 2013

Large search space for target

Small search space for target

Large effective dimension

Filtering of features of increasing complexity

small effective dimension

Akin to Coarse-graining/Renormalization (Gulio’s talk)

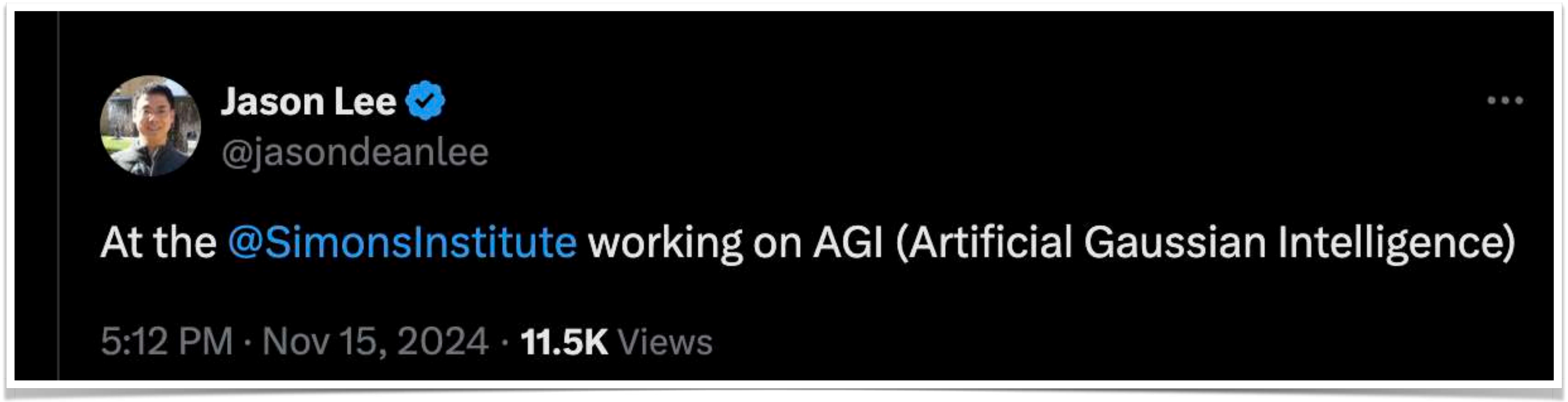
Can we understand this in some analyzable setting?

I

Recap of two-layer NNs for AGI

I

Recap of two-layer NNs for AGI



Neural Networks without Feature Learning

Random features

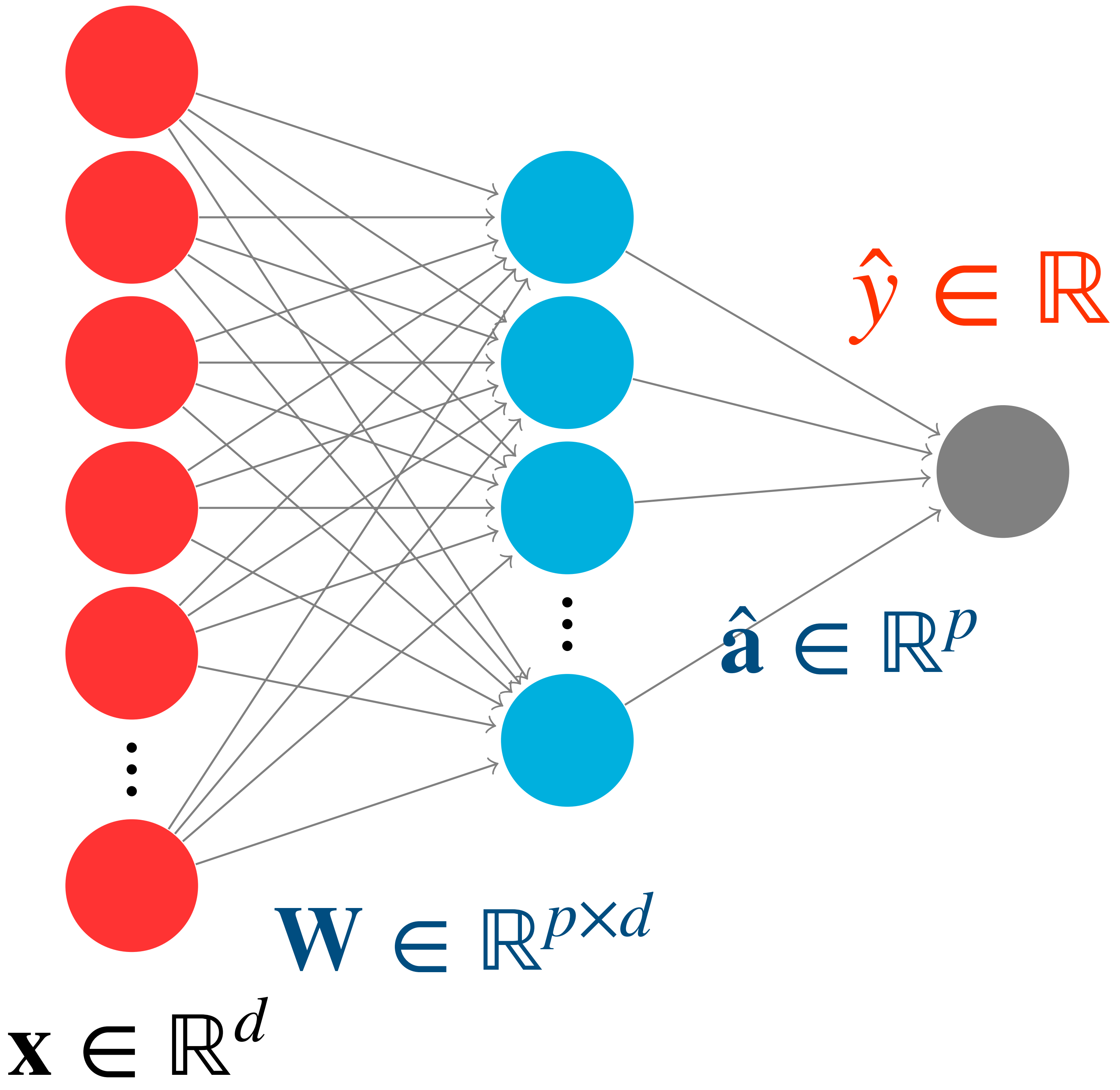
[Balcan,Blum, Vempala ’06, Rahimi-Recht ’17…]

$$\hat{y} = \hat{f}(\mathbf{x}) = \hat{\mathbf{a}} \cdot \sigma(W\mathbf{x})$$

No training of the first layer: W is fixed

$$\hat{y} = \hat{f}(\mathbf{x}) = \sum_{i=1}^{p} \hat{a}_i \sigma_i(\langle \mathbf{w}_i, \mathbf{x} \rangle) = \sum_{i=1}^{p} \hat{a}_i \Phi_{\mathrm{CK}}(\mathbf{x})$$

Computationally easy (linear regression)



Very popular setting among theoreticians

Equivalent to Neural Tangent Kernel/Lazy Regime/Kernel methods/ etc..

[Jacot, Gabriel, Hongler ’18; Lee, Jaehoon, et al. 18; Chizat, Bach ’19,…]

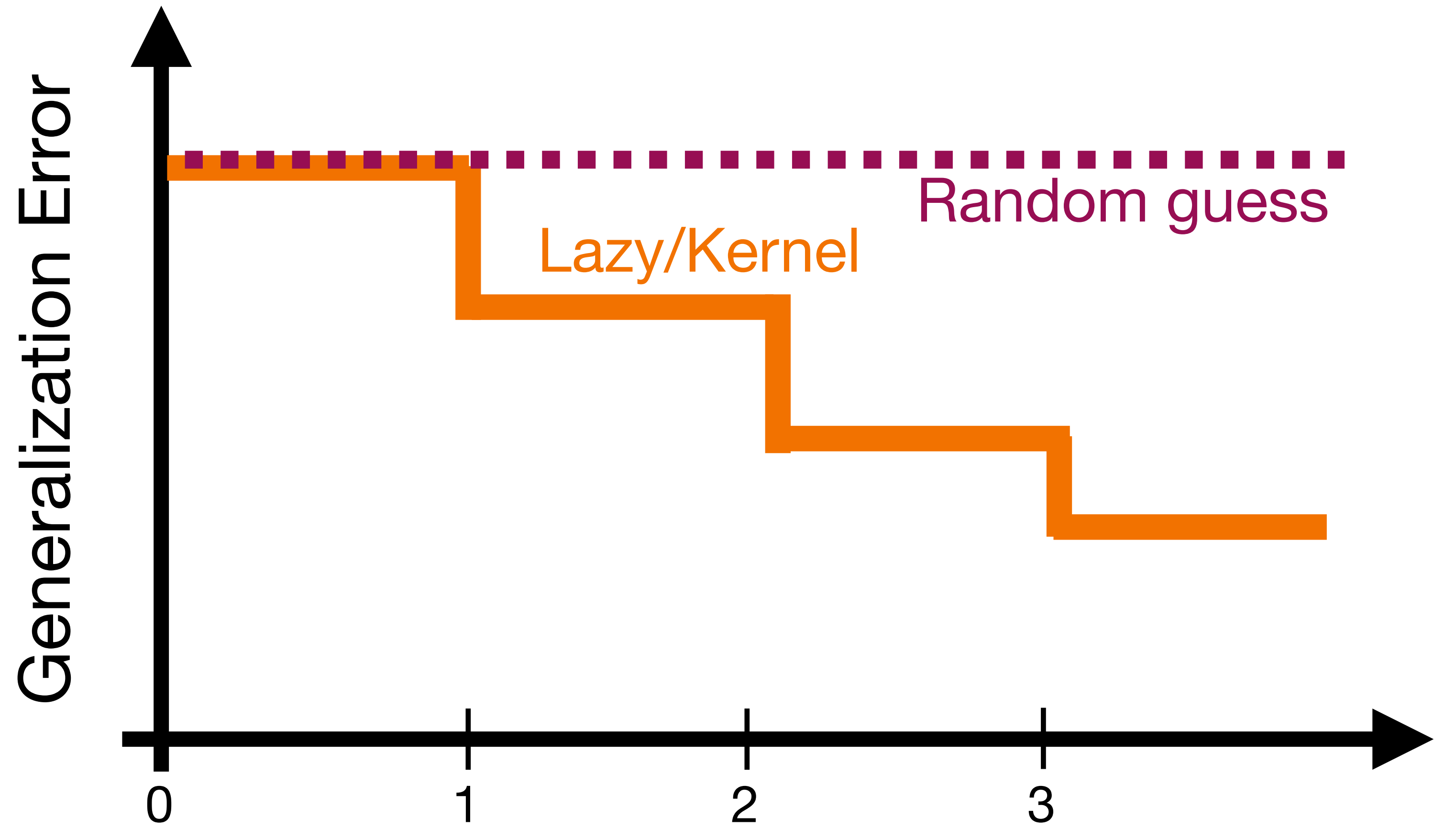
Recap of Sample Complexity

Theorem (Informal) [Mei, Misiakiewicz, Montanari '22]

In *absence* of feature learning (i.e. *at initialization)* one can only learn a ***polynomial*** **approximation of $f^\star$ of degree κ with** $\min(n,p) = O(d^\kappa)$

$$f^\star(\mathbf{x}) = \mathrm{cst} + \sum_i \mu_i^{(1)} h_i^\star + \sum_{ij} \mu_{ij}^{(2)} h_i^\star h_j^\star + \sum_{ijk} \mu_{ijk}^{(3)} h_i^\star h_j^\star h_k^\star + \dots$$

$$\mathscr{L} = \frac{1}{n}\sum_{i=1}^{n} (f^\star(\mathbf{x}) - \hat{\mathbf{f}}(\mathbf{x}))^2$$



See also [El Karaoui '10; Mei-Montanari '19; Gerace '20; Jacot, Simsek, Spadaro, Hongler, Gabriel '20; Hu, Lu, '20; Dhifallah, Lu '20; Loureiro, Gerbelot, Cui, Goldt '21; Montanari & Saeed '22; Xiao, Hu, Misiakiewicz, Lu, Pennington '22; Dandi '23; Aguirre-López, Franz, Pastore '24]

Recap of Sample Complexity

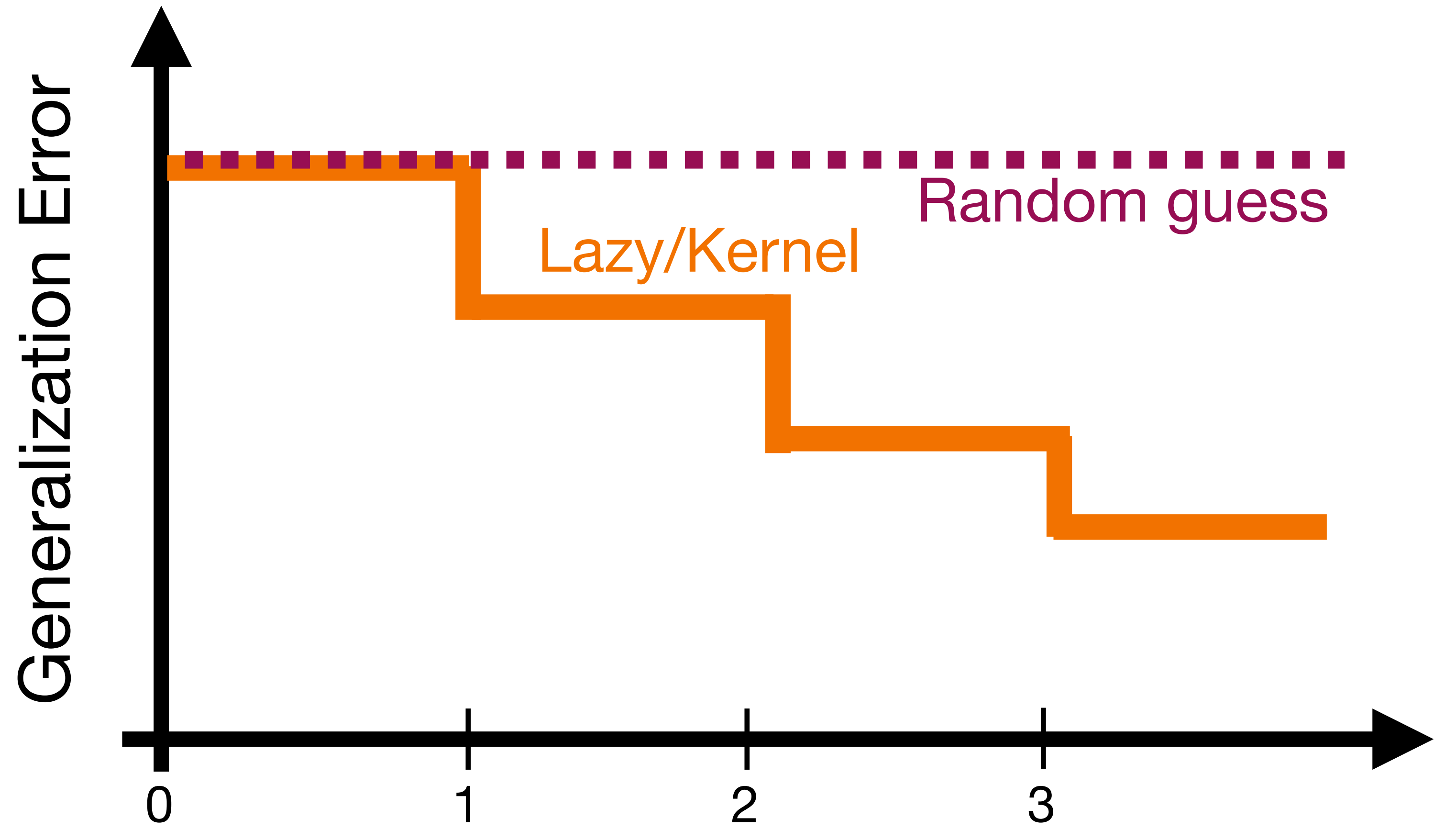
Theorem (Informal) [Mei, Misiakiewicz, Montanari '22]

In *absence* of feature learning (i.e. *at initialization)* one can only learn a ***polynomial*** **approximation of** $f^\star$ **of degree** κ **with** $\min(n,p) = O(d^\kappa)$

$$f^\star(\mathbf{x}) = \mathrm{cst} + \sum_i \mu_i^{(1)} h_i^\star + \sum_{ij} \mu_{ij}^{(2)} h_i^\star h_j^\star + \sum_{ijk} \mu_{ijk}^{(3)} h_i^\star h_j^\star h_k^\star + \dots$$

$$\mathscr{L} = \frac{1}{n} \sum_{i=1}^{n} (f^\star(\mathbf{x}) - \hat{\mathbf{f}}(\mathbf{x}))^2$$

$(n,p) = O(d)$



See also [El Karaoui '10; Mei-Montanari '19; Gerace '20; Jacot, Simsek, Spadaro, Hongler, Gabriel '20; Hu, Lu, '20; Dhifallah, Lu '20; Loureiro, Gerbelot, Cui, Goldt '21; Montanari & Saeed '22; Xiao, Hu, Misiakiewicz, Lu, Pennington '22; Dandi '23; Aguirre-López, Franz, Pastore '24]

Recap of Sample Complexity

Theorem (Informal) [Mei, Misiakiewicz, Montanari '22]

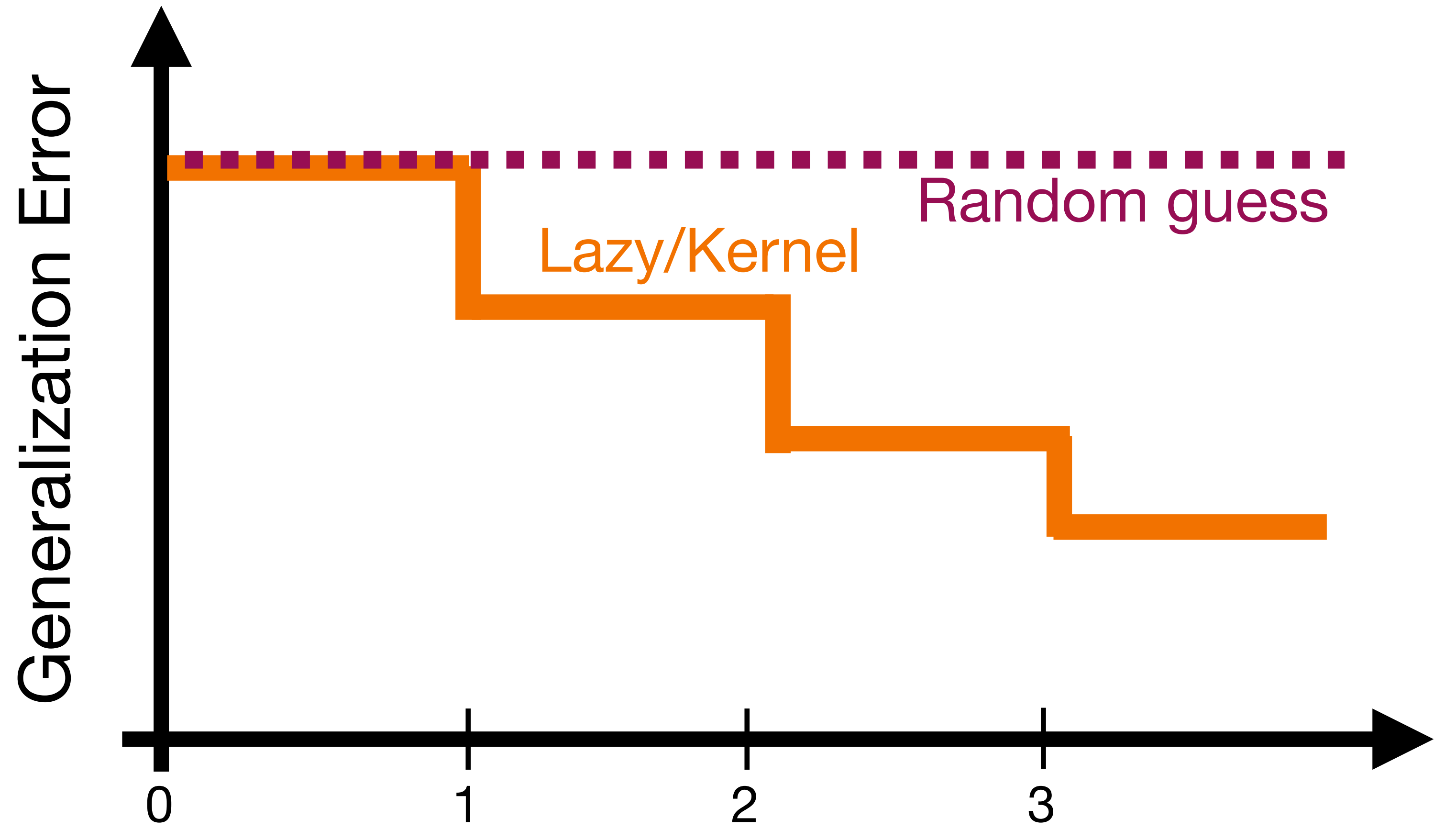
In *absence* of feature learning (i.e. *at initialization)* one can only learn a ***polynomial*** **approximation of $f^\star$ of degree κ with** $\min(n,p) = O(d^\kappa)$

$$f^\star(\mathbf{x}) = \text{cst} + \sum_i \mu_i^{(1)} h_i^\star + \sum_{ij} \mu_{ij}^{(2)} h_i^\star h_j^\star + \sum_{ijk} \mu_{ijk}^{(3)} h_i^\star h_j^\star h_k^\star + \dots$$

$$\mathscr{L} = \frac{1}{n}\sum_{i=1}^{n} (f^\star(\mathbf{x}) - \hat{\mathbf{f}}(\mathbf{x}))^2$$

$(n,p) = O(d)$

$(n,p) = O(d^2)$



See also [El Karaoui '10; Mei-Montanari '19; Gerace '20; Jacot, Simsek, Spadaro, Hongler, Gabriel '20; Hu, Lu, '20; Dhifallah, Lu '20; Loureiro, Gerbelot, Cui, Goldt '21; Montanari & Saeed '22; Xiao, Hu, Misiakiewicz, Lu, Pennington '22; Dandi '23; Aguirre-López, Franz, Pastore '24]

Recap of Sample Complexity

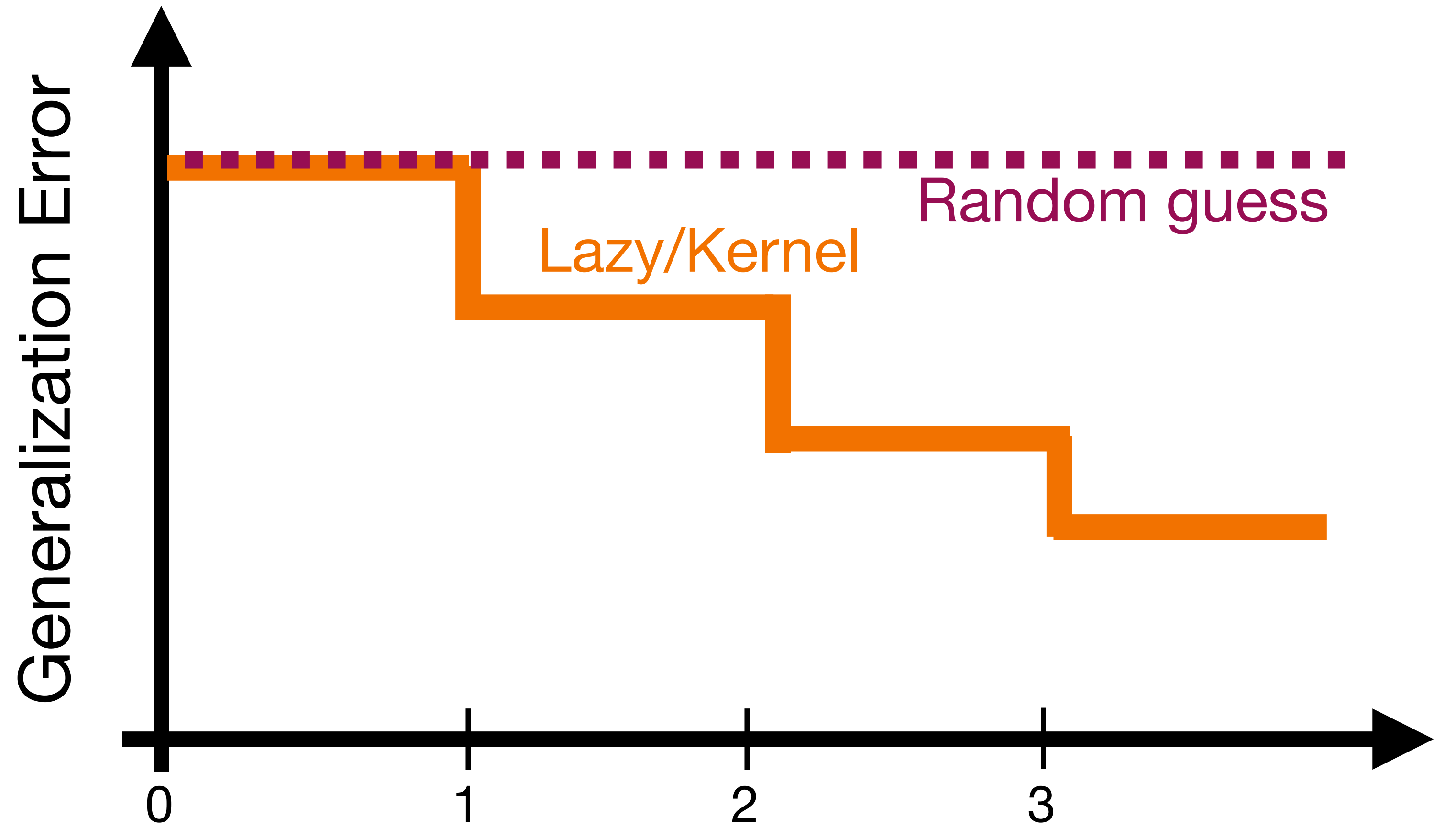
Theorem (Informal) [Mei, Misiakiewicz, Montanari '22]

In *absence* of feature learning (i.e. *at initialization)* one can only learn a ***polynomial*** **approximation of** $f^\star$ **of degree** κ **with** $\min(n,p) = O(d^\kappa)$

$$f^\star(\mathbf{x}) = \mathrm{cst} + \sum_i \mu_i^{(1)} h_i^\star + \sum_{ij} \mu_{ij}^{(2)} h_i^\star h_j^\star + \sum_{ijk} \mu_{ijk}^{(3)} h_i^\star h_j^\star h_k^\star + \dots$$

$$\mathscr{L} = \frac{1}{n} \sum_{i=1}^{n} (f^\star(\mathbf{x}) - \hat{\mathbf{f}}(\mathbf{x}))^2$$

$(n,p) = O(d)$ $(n,p) = O(d^2)$ $(n,p) = O(d^3)$



See also [El Karaoui '10; Mei-Montanari '19; Gerace '20; Jacot, Simsek, Spadaro, Hongler, Gabriel '20; Hu, Lu, '20; Dhifallah, Lu '20; Loureiro, Gerbelot, Cui, Goldt '21; Montanari & Saeed '22; Xiao, Hu, Misiakiewicz, Lu, Pennington '22; Dandi '23; Aguirre-López, Franz, Pastore '24]

Recap of Sample Complexity

Theorem (Informal) [Mei, Misiakiewicz, Montanari '22]

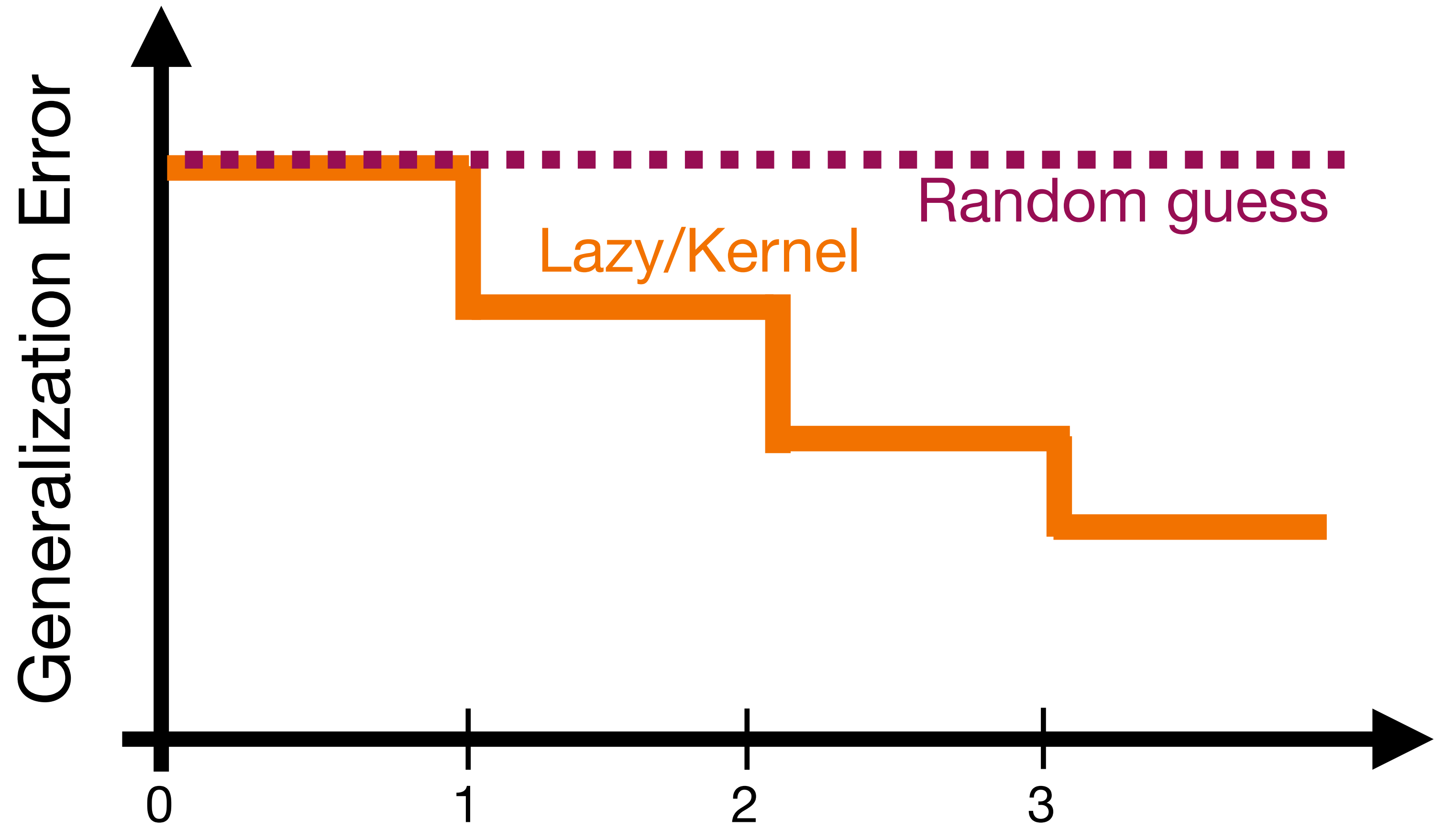
In *absence* of feature learning (i.e. *at initialization)* one can only learn a ***polynomial*** **approximation of** $f^\star$ **of degree** κ **with** $\min(n,p) = O(d^\kappa)$

$$f^\star(\mathbf{x}) = \text{cst} + \sum_i \mu_i^{(1)} h_i^\star + \sum_{ij} \mu_{ij}^{(2)} h_i^\star h_j^\star + \sum_{ijk} \mu_{ijk}^{(3)} h_i^\star h_j^\star h_k^\star + \dots$$

$$\mathscr{L} = \frac{1}{n} \sum_{i=1}^{n} (f^\star(\mathbf{x}) - \hat{\mathbf{f}}(\mathbf{x}))^2$$

$(n,p) = O(d)$ $(n,p) = O(d^2)$ $(n,p) = O(d^3)$

No adaptivity $\Longrightarrow$ searching in a $\mathcal{O}(d^k)$ dimensional subspace of polynomials*



See also [El Karaoui '10; Mei-Montanari '19; Gerace '20; Jacot, Simsek, Spadaro, Hongler, Gabriel '20; Hu, Lu, '20; Dhifallah, Lu '20; Loureiro, Gerbelot, Cui, Goldt '21; Montanari & Saeed '22; Xiao, Hu, Misiakiewicz, Lu, Pennington '22; Dandi '23; Aguirre-López, Franz, Pastore '24]

Recap of Sample Complexity

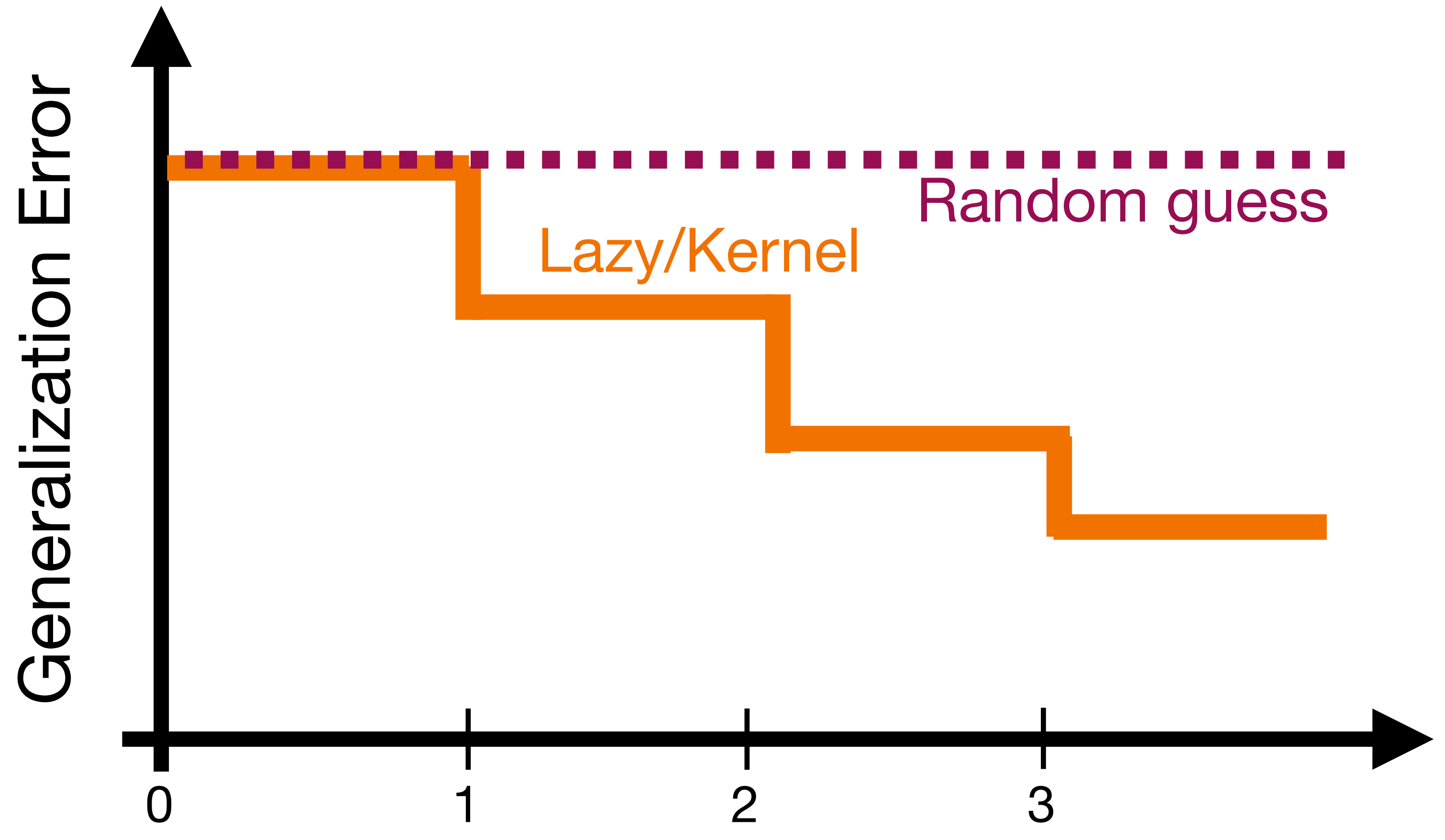
Theorem (Informal) [Mei, Misiakiewicz, Montanari '22]

In *absence* of feature learning (i.e. *at initialization)* one can only learn a ***polynomial*** **approximation of** $f^\star$ **of degree** κ **with** $\min(n,p) = O(d^\kappa)$

$$f^\star(\mathbf{x}) = \text{cst} + \sum_i \mu_i^{(1)} h_i^\star + \sum_{ij} \mu_{ij}^{(2)} h_i^\star h_j^\star + \sum_{ijk} \mu_{ijk}^{(3)} h_i^\star h_j^\star h_k^\star + \dots$$

$$\mathcal{L} = \frac{1}{n} \sum_{i=1}^{n} (f^\star(\mathbf{x}) - \hat{\mathbf{f}}(\mathbf{x}))^2$$

$(n,p) = O(d)$ $\quad$ $(n,p) = O(d^2)$ $\quad$ $(n,p) = O(d^3)$



No adaptivity $\Longrightarrow$ searching in a $\mathcal{O}(d^k)$ dimensional subspace of polynomials*

***possibly excluding a few "special" polynomials.**

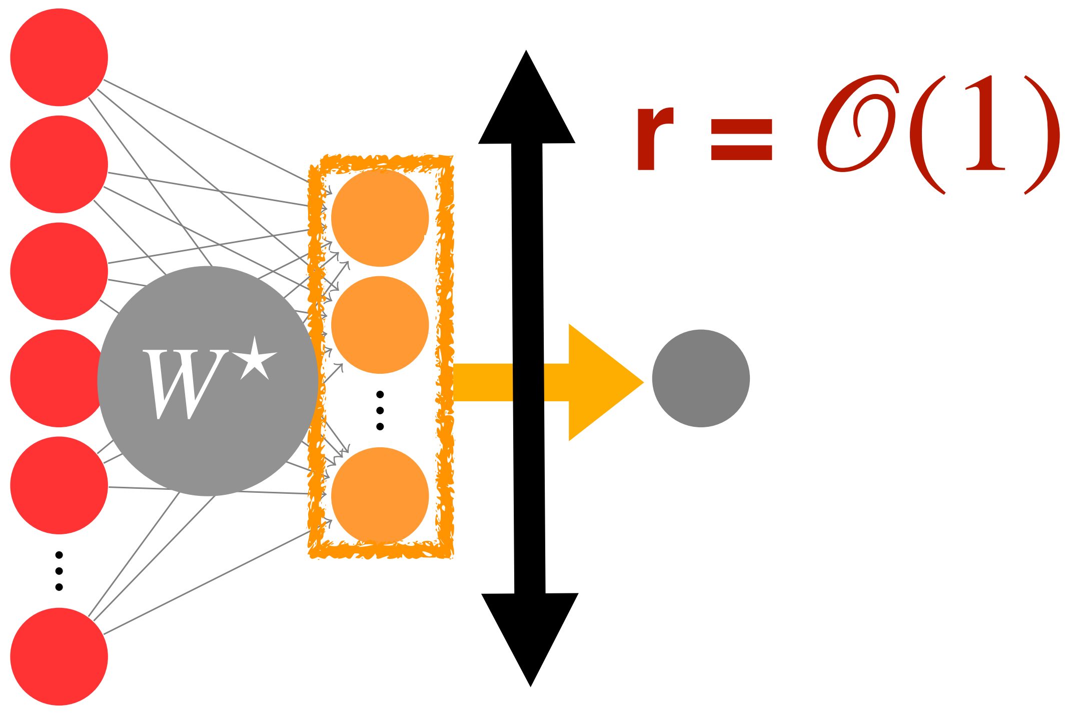
See also [El Karaoui '10; Mei-Montanari '19; Gerace '20; Jacot, Simsek, Spadaro, Hongler, Gabriel '20; Hu, Lu, '20; Dhifallah, Lu '20; Loureiro, Gerbelot, Cui, Goldt '21; Montanari & Saeed '22; Xiao, Hu, Misiakiewicz, Lu, Pennington '22; Dandi '23; Aguirre-López, Franz, Pastore '24]

Advantage of Feature Learning

Ben Arous et al. 2021, Abbe et al. 2022,2023, Damian et al. 2022, Dandi et al. 2023, 2024, etc.

Advantage of Feature Learning

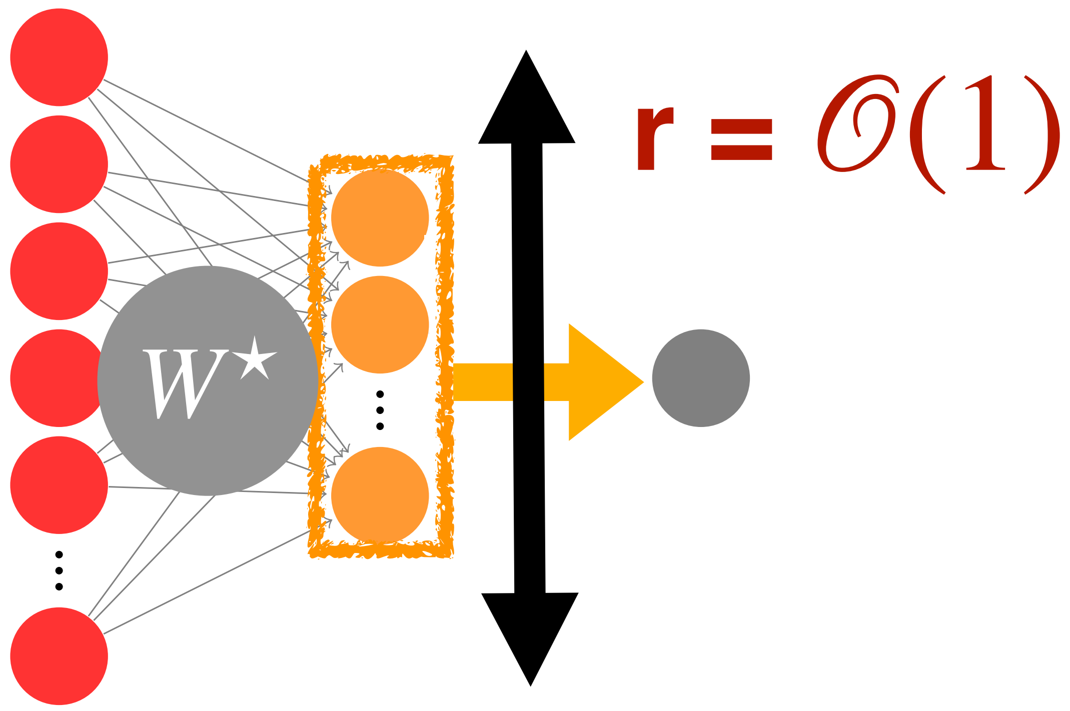
$$y = f^\star(\mathbf{x}) = \mathbf{g}^\star(\mathbf{x}_\star = \mathbf{W}^\star \mathbf{x})$$



Ben Arous et al. 2021, Abbe et al. 2022,2023, Damian et al. 2022, Dandi et al. 2023, 2024, etc.

Advantage of Feature Learning

$$y = f^\star(\mathbf{x}) = \mathbf{g}^\star(\mathbf{x}_\star = \mathbf{W}^\star\mathbf{x})$$

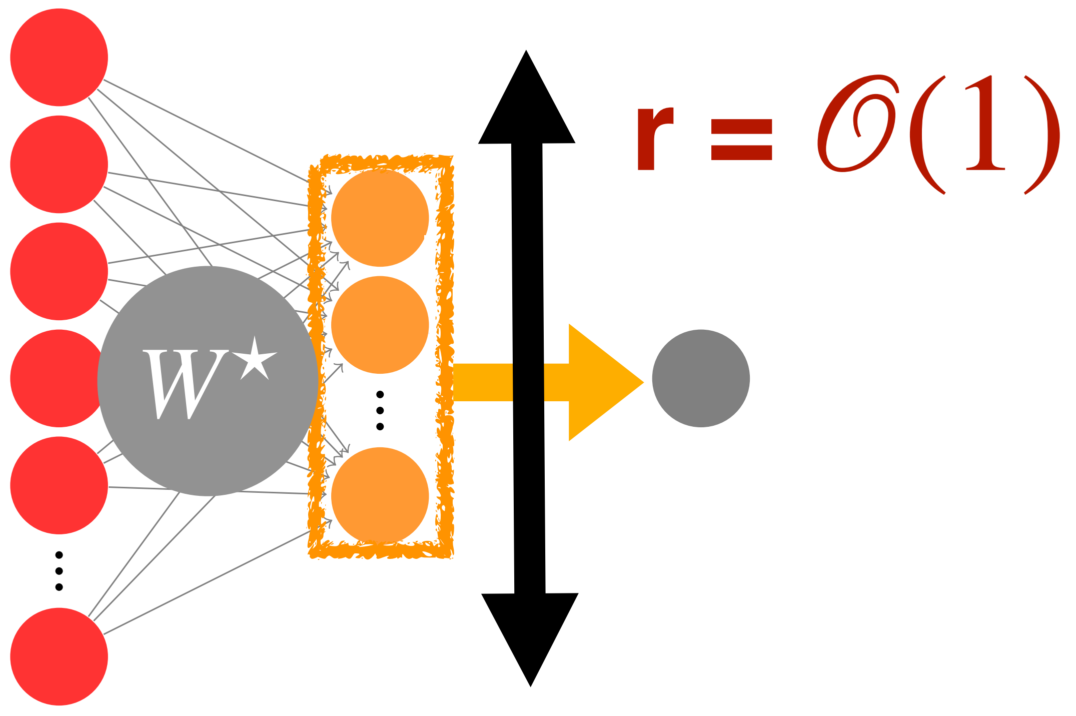


Information/ generative exponent ≤ 2

Ben Arous et al. 2021, Abbe et al. 2022,2023, Damian et al. 2022, Dandi et al. 2023, 2024, etc.

Advantage of Feature Learning

$y = f^\star(\mathbf{x}) = \mathbf{g}^\star(\mathbf{x}_\star = \mathbf{W}^\star \mathbf{x})$

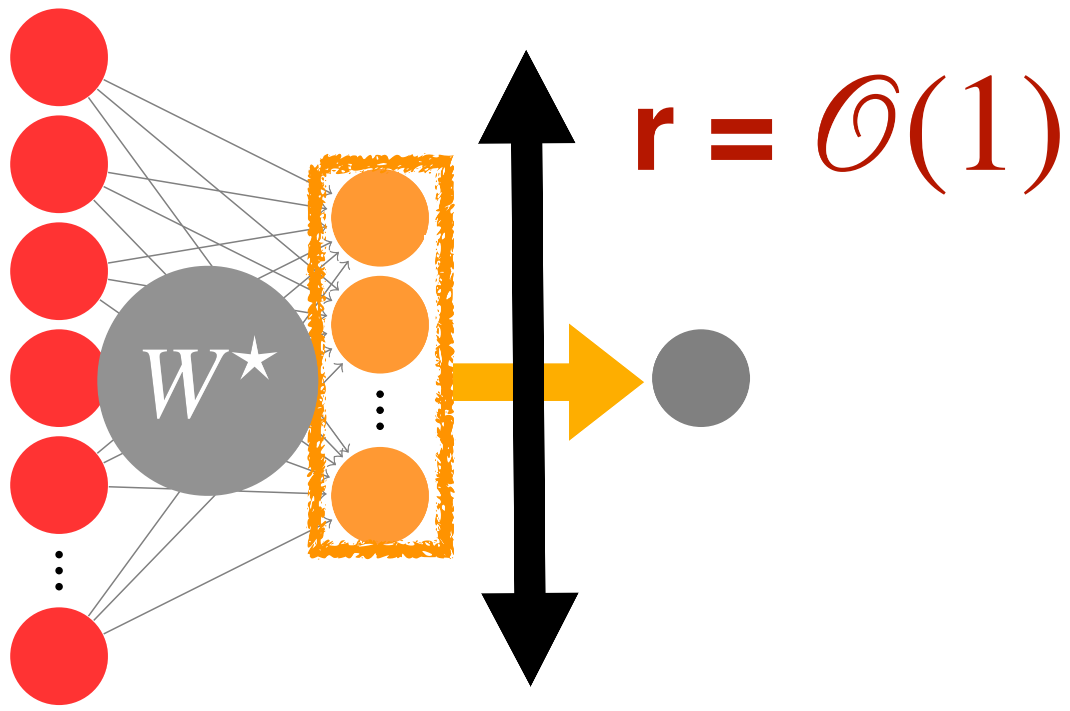


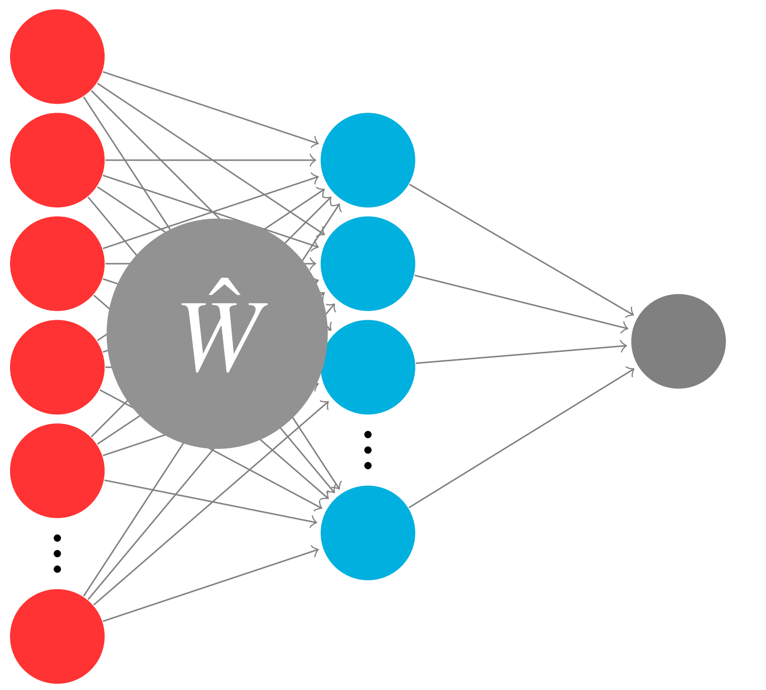
Information/ generative exponent ≤ 2 →

Ben Arous et al. 2021, Abbe et al. 2022,2023, Damian et al. 2022, Dandi et al. 2023, 2024, etc.

Advantage of Feature Learning

$$y = f^{\star}(\mathbf{x}) = \mathbf{g}^{\star}(\mathbf{x}_{\star} = \mathbf{W}^{\star}\mathbf{x})$$





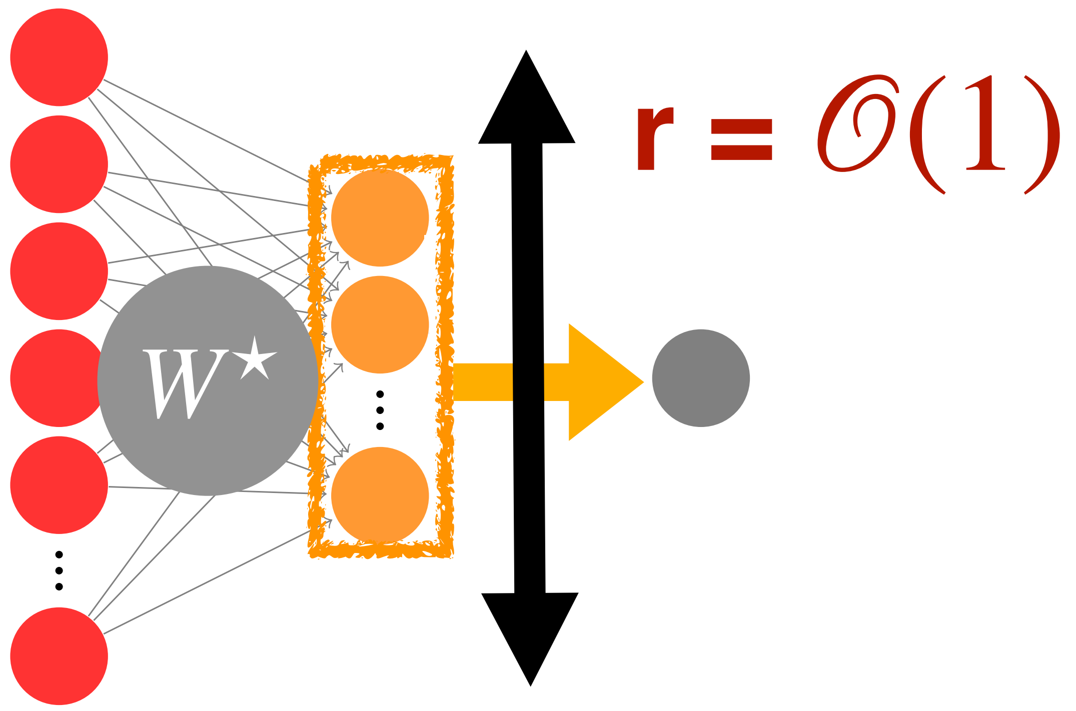
$$\hat{y} = \hat{\mathbf{a}} \cdot \sigma(\hat{W}\mathbf{x})$$

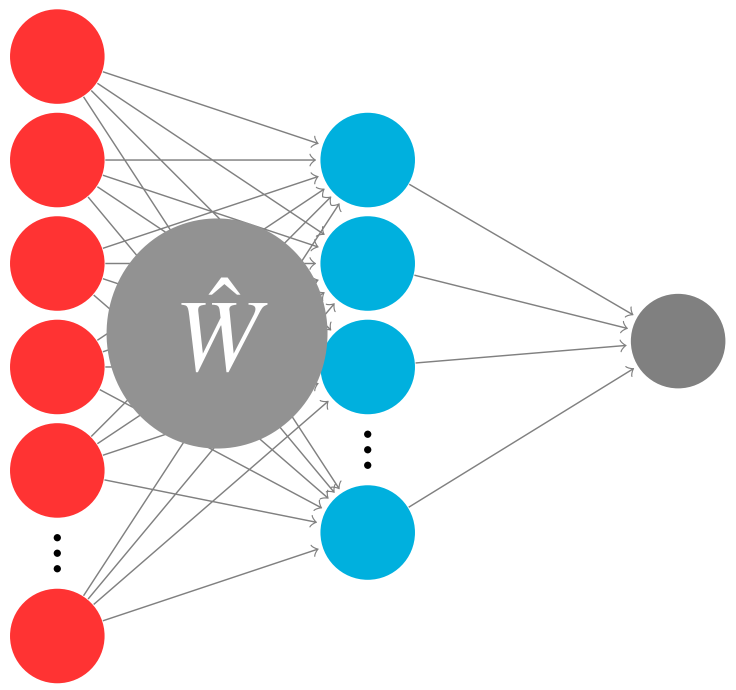
Information/ generative exponent ≤ 2 →

Ben Arous et al. 2021, Abbe et al. 2022,2023, Damian et al. 2022, Dandi et al. 2023, 2024, etc.

Advantage of Feature Learning

$$y = f^\star(\mathbf{x}) = \mathbf{g}^\star(\mathbf{x}_\star = \mathbf{W}^\star \mathbf{x})$$





$$\hat{y} = \hat{\mathbf{a}} \cdot \sigma(\hat{W}\mathbf{x})$$

Information/ generative exponent ≤ 2

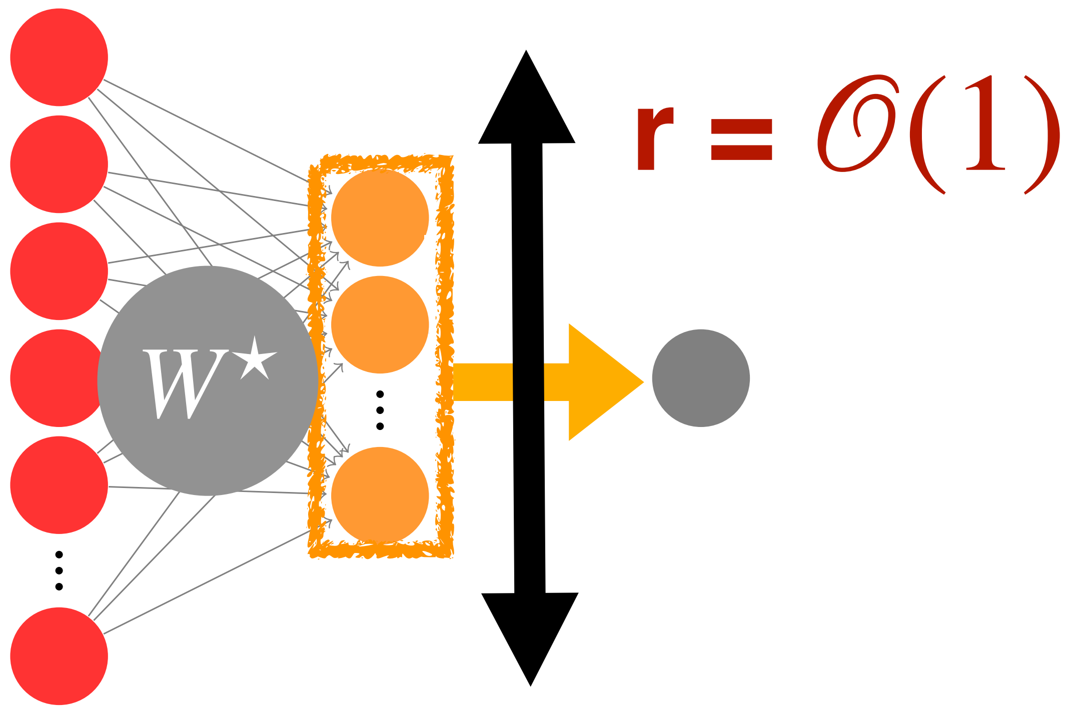
With $\tilde{O}(d)$ samples, GD/SGD on $\hat{W}$ recovers

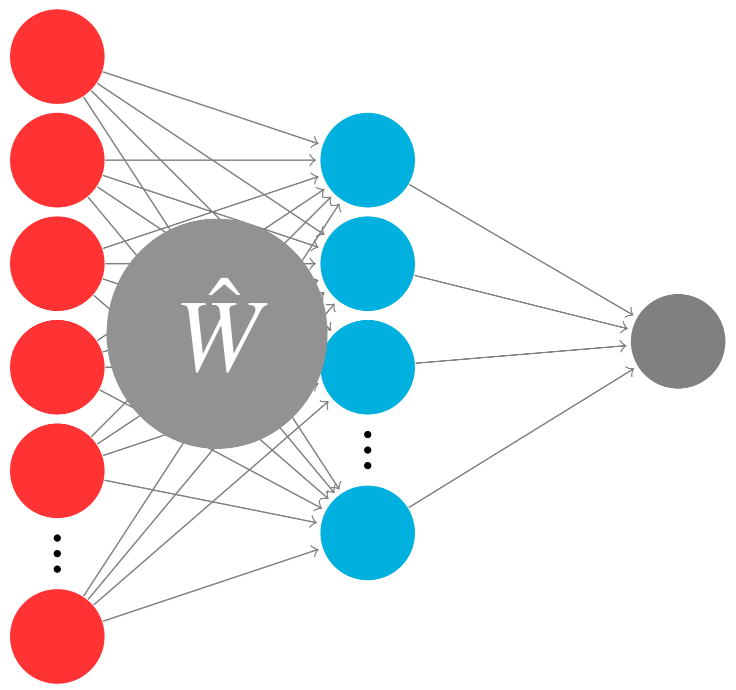
$$\hat{W} \approx Z_1 W^\star + Z_2$$

Ben Arous et al. 2021, Abbe et al. 2022,2023, Damian et al. 2022, Dandi et al. 2023, 2024, etc.

Advantage of Feature Learning

$y = f^\star(\mathbf{x}) = \mathbf{g}^\star(\mathbf{x}_\star = \mathbf{W}^\star \mathbf{x})$





$\hat{y} = \hat{\mathbf{a}} \cdot \sigma(\hat{W}\mathbf{x})$

Information/ generative exponent ≤ 2

With $\tilde{O}(d)$ samples, GD/SGD on $\hat{W}$ recovers

$$\hat{W} \approx Z_1 W^\star + Z_2$$

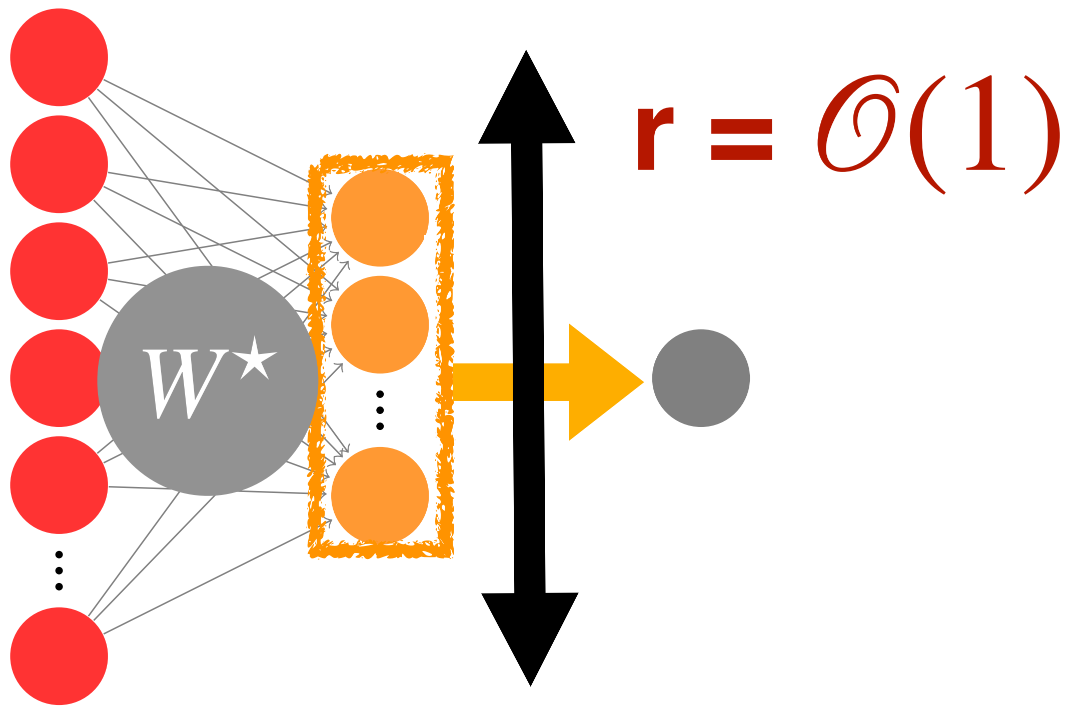
Then the neural net is equivalent to

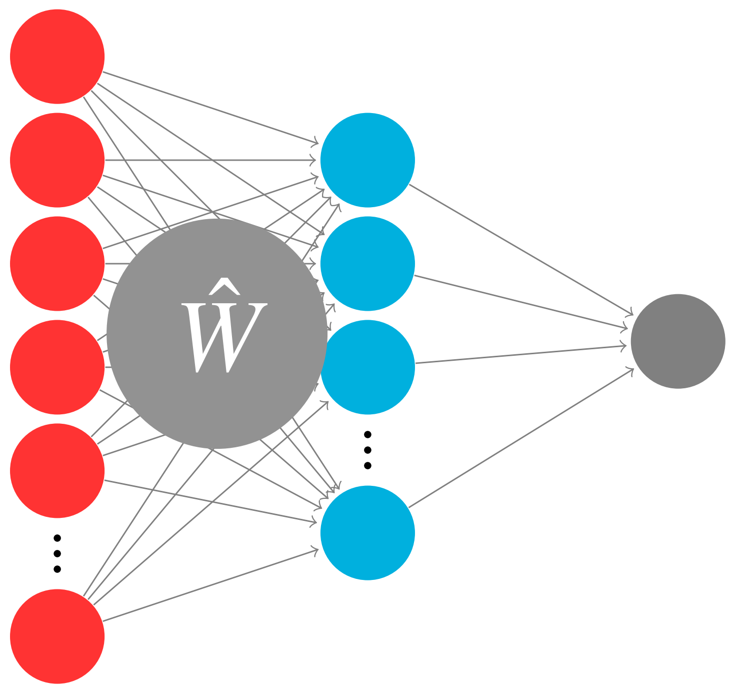
$$\hat{y} \approx \hat{\mathbf{a}} \cdot \sigma((Z_1 W^\star + Z_2)\mathbf{x}) = \hat{\mathbf{a}} \cdot \sigma(\mathbf{Z_1}\mathbf{x}^\star + \mathbf{Z_3})$$

Ben Arous et al. 2021, Abbe et al. 2022,2023, Damian et al. 2022, Dandi et al. 2023, 2024, etc.

Advantage of Feature Learning

$y = f^\star(\mathbf{x}) = \mathbf{g}^\star(\mathbf{x}_\star = \mathbf{W}^\star \mathbf{x})$





$\hat{y} = \hat{\mathbf{a}} \cdot \sigma(\hat{W}\mathbf{x})$

Information/ generative exponent ≤ 2

With $\tilde{O}(d)$ samples, GD/SGD on $\hat{W}$ recovers

$$\hat{W} \approx Z_1 W^\star + Z_2$$

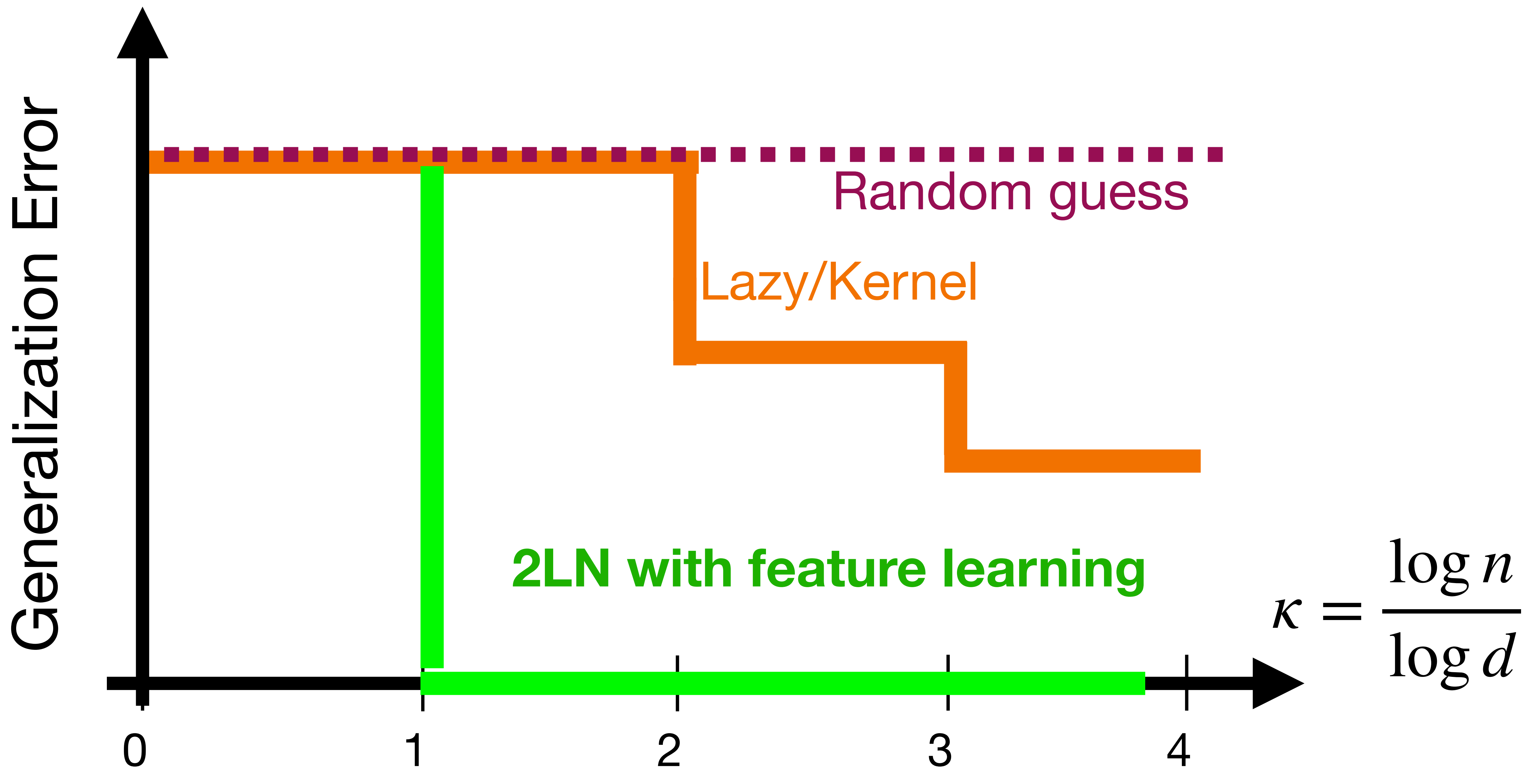
Then the neural net is equivalent to

$$\hat{y} \approx \hat{\mathbf{a}} \cdot \sigma((Z_1 W^\star + Z_2)\mathbf{x}) = \hat{\mathbf{a}} \cdot \sigma(\mathbf{Z_1 x^\star} + \mathbf{Z_3})$$

Random feature in (finite) **reduced** space
$d \to d^{\text{eff}} = r = O(1)$

Ben Arous et al. 2021, Abbe et al. 2022,2023, Damian et al. 2022, Dandi et al. 2023, 2024, etc.

Dimensionality reduction

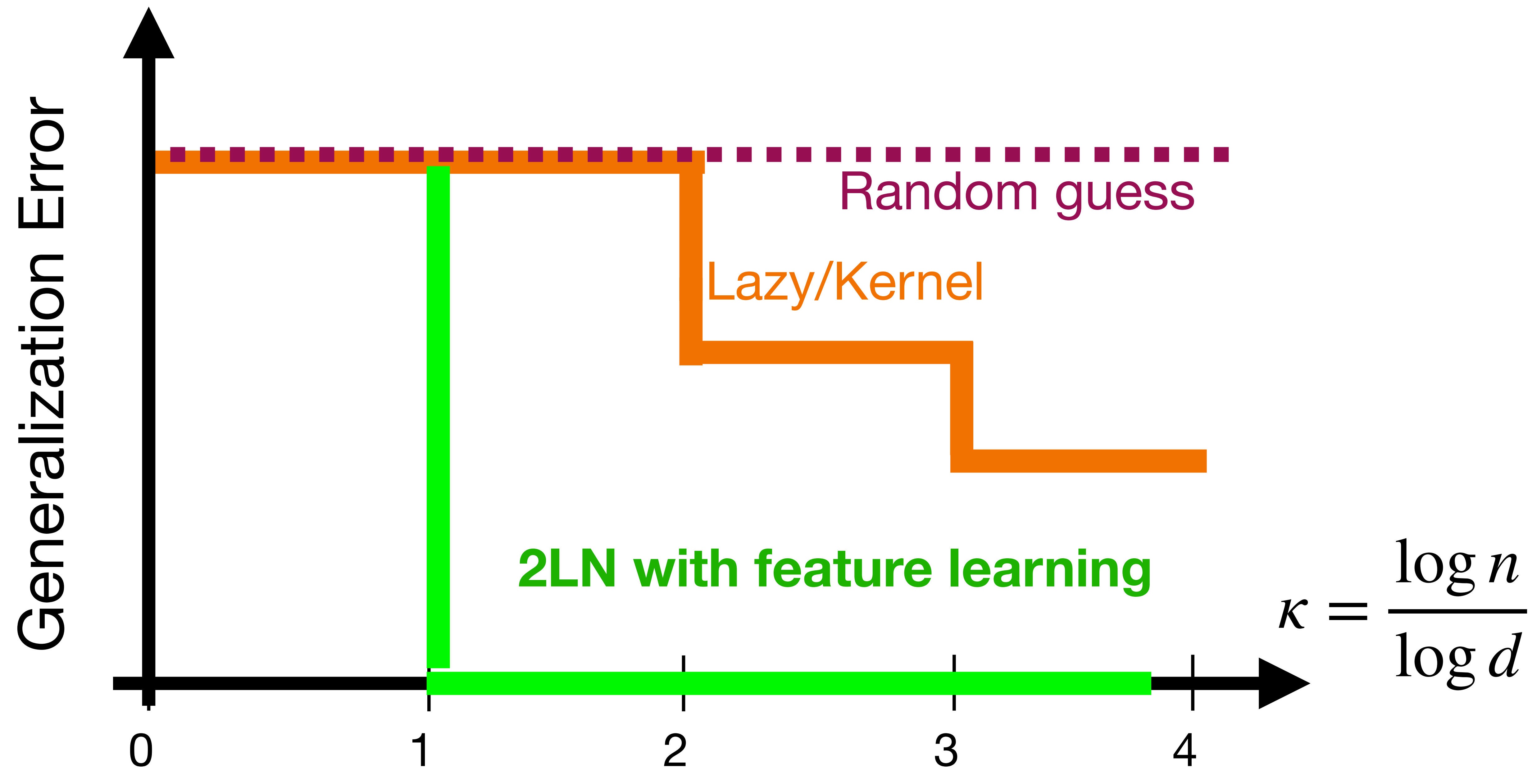


Sample Complexity Reduction

$\mathcal{O}(d^k) \to \tilde{O}(d)$

Dimensionality reduction

searching in a $\mathcal{O}(d^k)$ dimensional subspace of polynomials

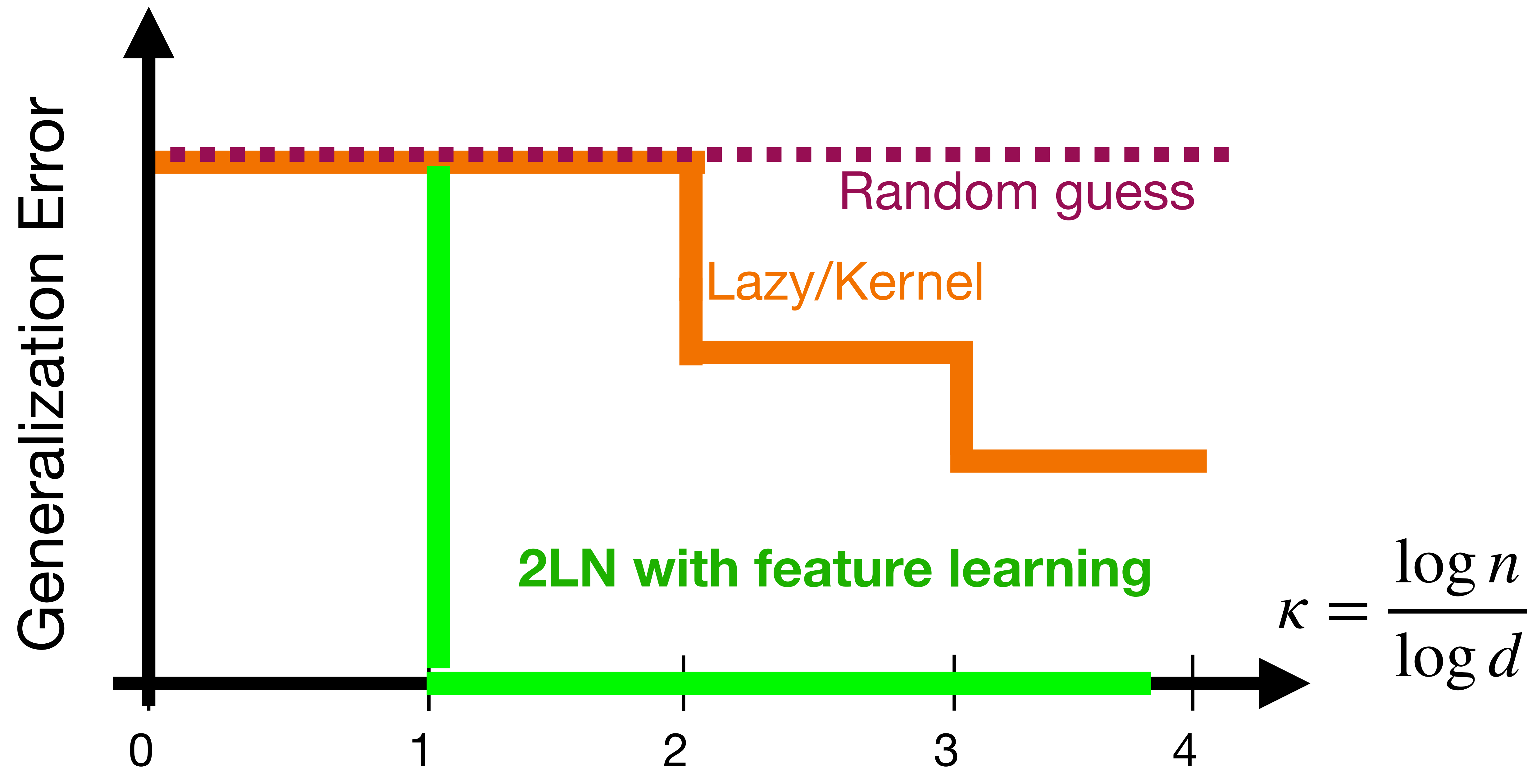


Sample Complexity Reduction

$\mathcal{O}(d^k) \rightarrow \tilde{O}(d)$

Dimensionality reduction

searching in a $\mathcal{O}(d^k)$ dimensional subspace of polynomials



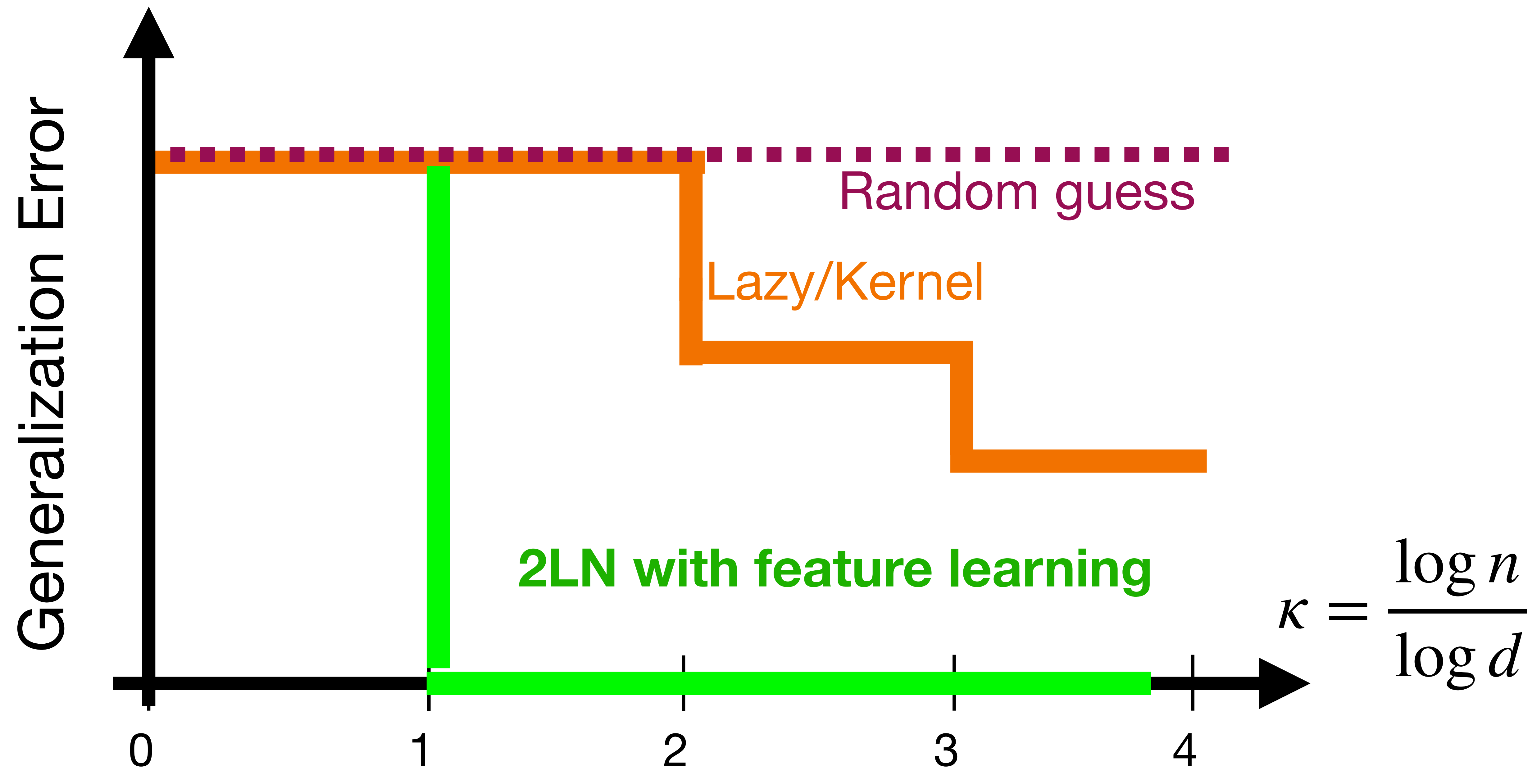
Sample Complexity Reduction
$\mathcal{O}(d^k) \rightarrow \tilde{O}(d)$

Dimensionality reduction

searching in a $\mathcal{O}(d^k)$ dimensional subspace of polynomials

→

searching in a $\mathcal{O}(r^k)$ dimensional subspace of polynomials



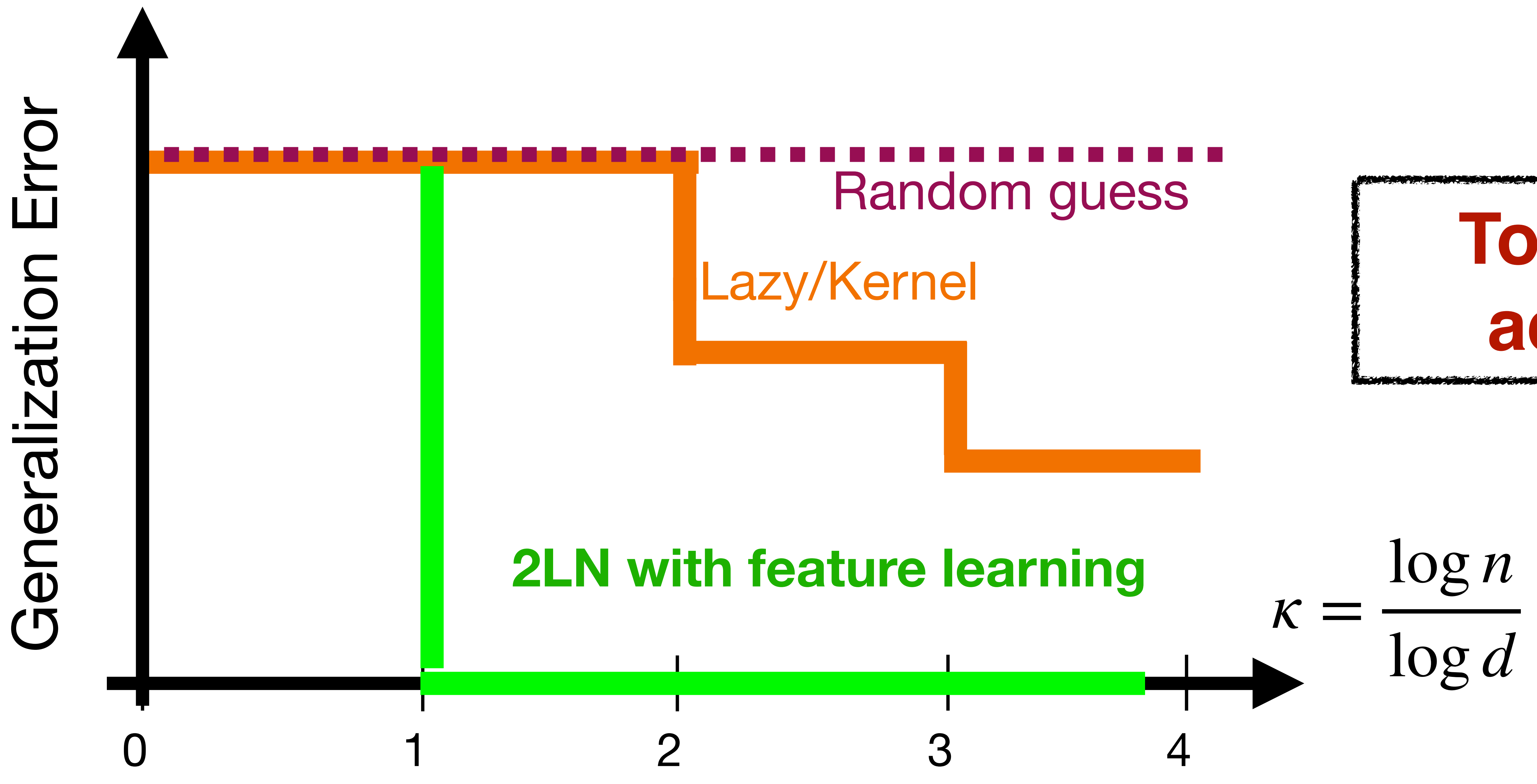
Sample Complexity Reduction
$\mathcal{O}(d^k) \rightarrow \tilde{O}(d)$

Dimensionality reduction

searching in a $\mathcal{O}(d^k)$ dimensional subspace of polynomials

→

searching in a $\mathcal{O}(r^k)$ dimensional subspace of polynomials



Too easy…no further advantage of depth

Sample Complexity Reduction

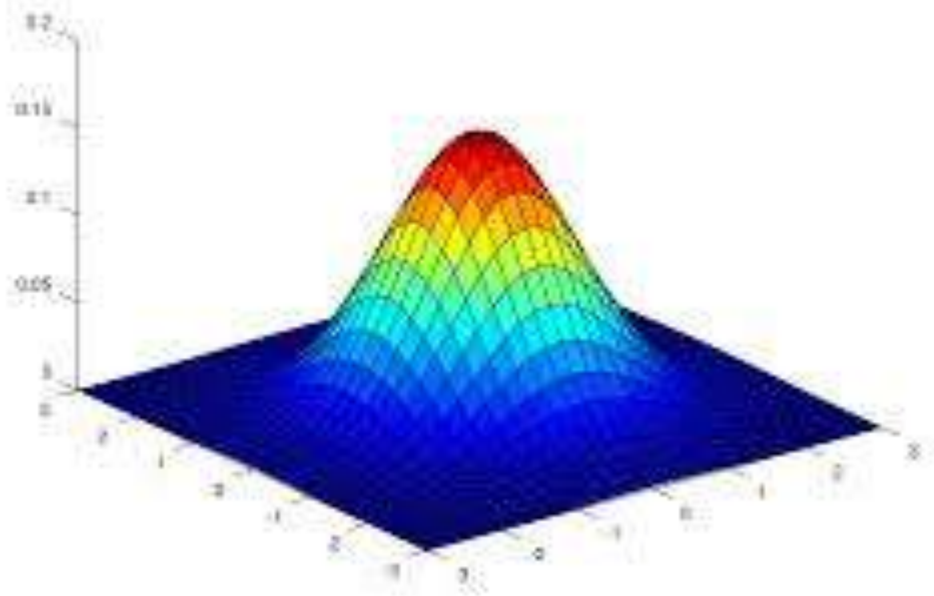
$\mathcal{O}(d^k) \to \tilde{O}(d)$

Can we generalize this picture to arbitrary depth?

A SIGHT beyond Two Layers (Single-Index Gaussian Hierarchical Targets)

$$f^\star(\mathbf{x}) = g^\star\left(\frac{\mathbf{a}^{\star\top} P_k\left(W^\star \mathbf{x}\right)}{\sqrt{d^\varepsilon}}\right), \mathbf{x} \in \mathbb{R}^d$$

P_k: polynomial of degree k
$g^\star$: non-linearity



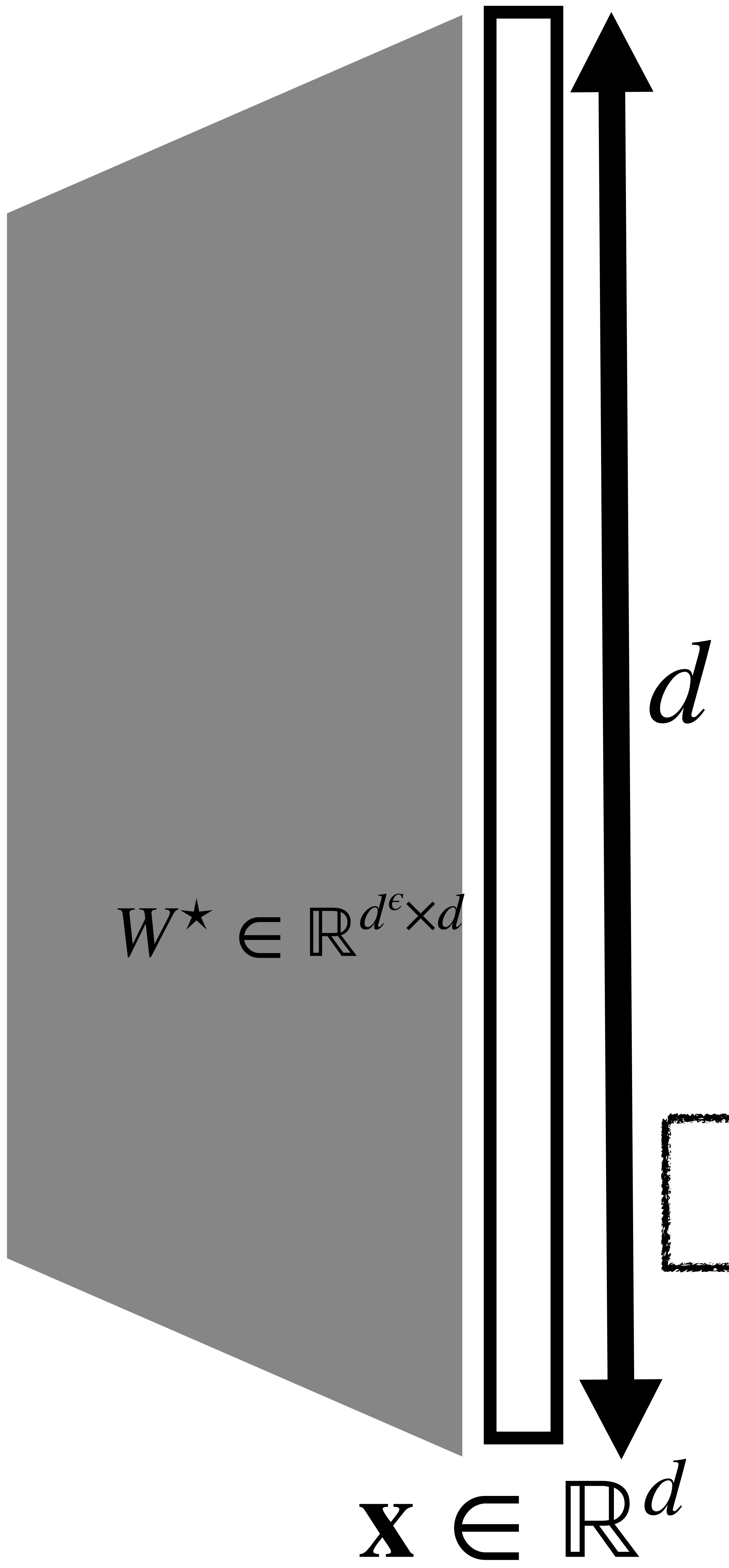
high-dimensional Gaussian
$\mathbf{x} \sim \mathcal{N}(\mathbf{0}, \mathbf{I_d})$

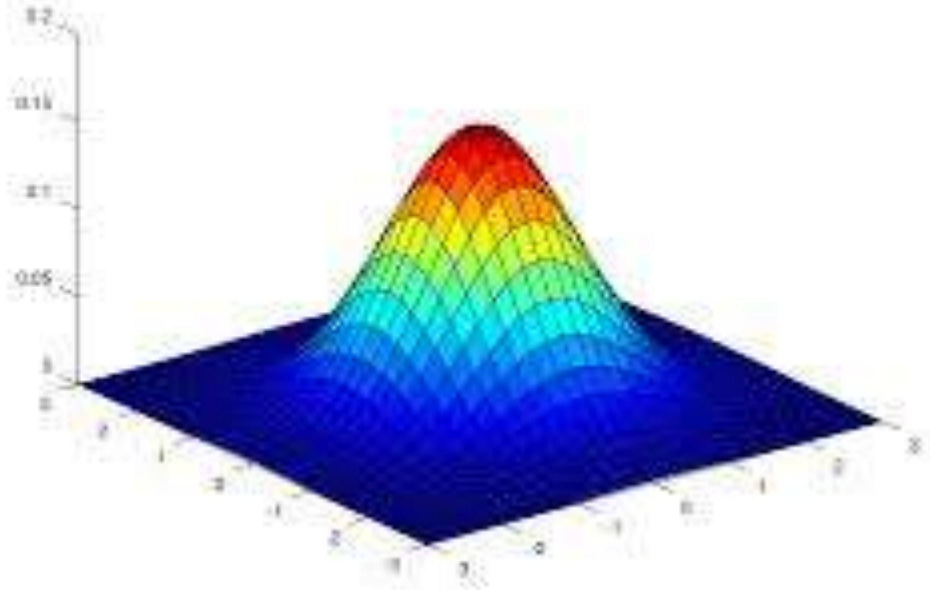
A SIGHT beyond Two Layers (Single-Index Gaussian Hierarchical Targets)

$$f^\star(\mathbf{x}) = g^\star\left(\frac{{\mathbf{a}^\star}^\top P_k\left(W^\star \mathbf{x}\right)}{\sqrt{d^\varepsilon}}\right), \ \mathbf{x} \in \mathbb{R}^d$$

P_k: **polynomial of degree** k

$g^\star$: **non-linearity**





high-dimensional Gaussian

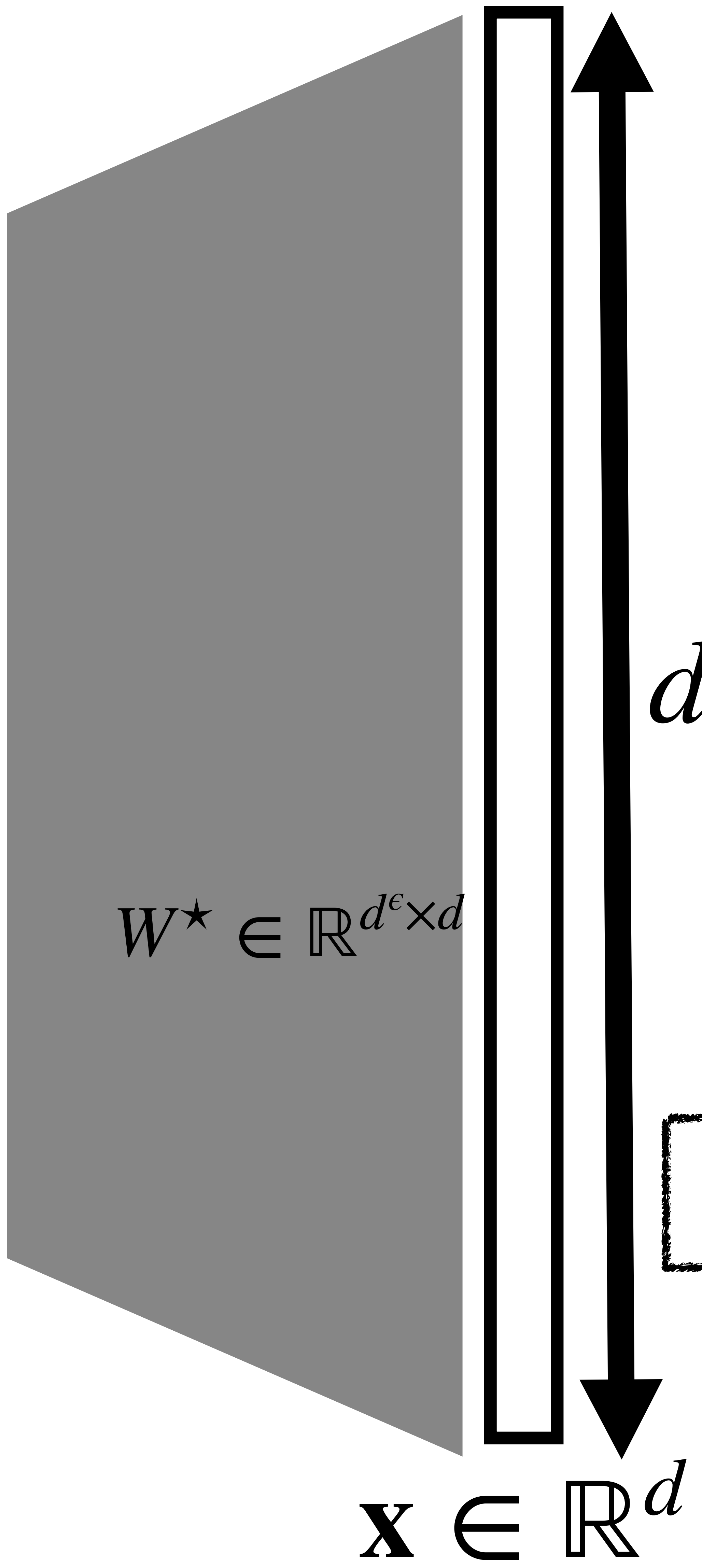
$\mathbf{x} \sim \mathcal{N}(\mathbf{0}, \mathbf{I_d})$

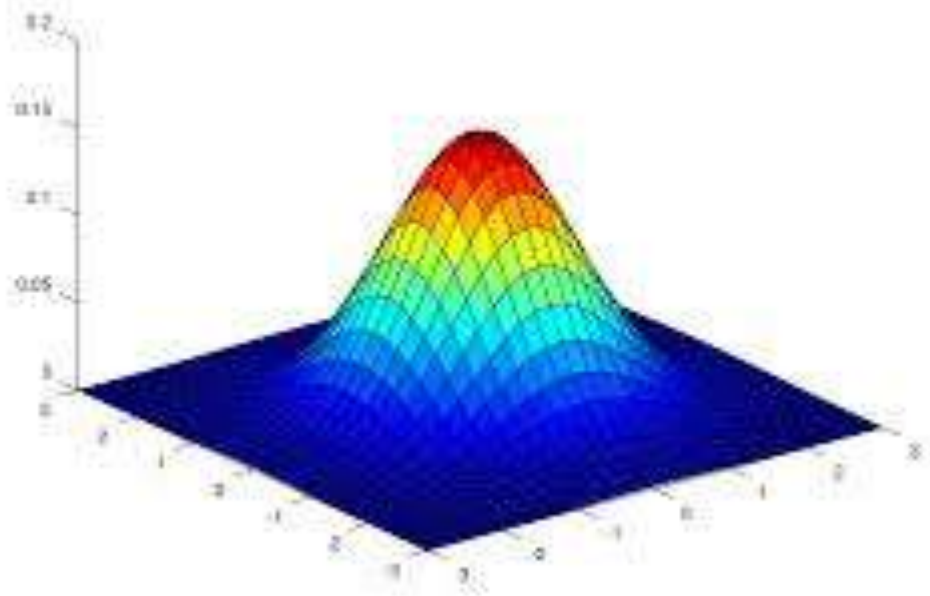
A SIGHT beyond Two Layers (Single-Index Gaussian Hierarchical Targets)

$$f^\star(\mathbf{x}) = g^\star\left(\frac{{\mathbf{a}^\star}^\top P_k\left(W^\star \mathbf{x}\right)}{\sqrt{d^\varepsilon}}\right), \mathbf{x} \in \mathbb{R}^d$$

P_k: polynomial of degree k

$g^\star$: non-linearity





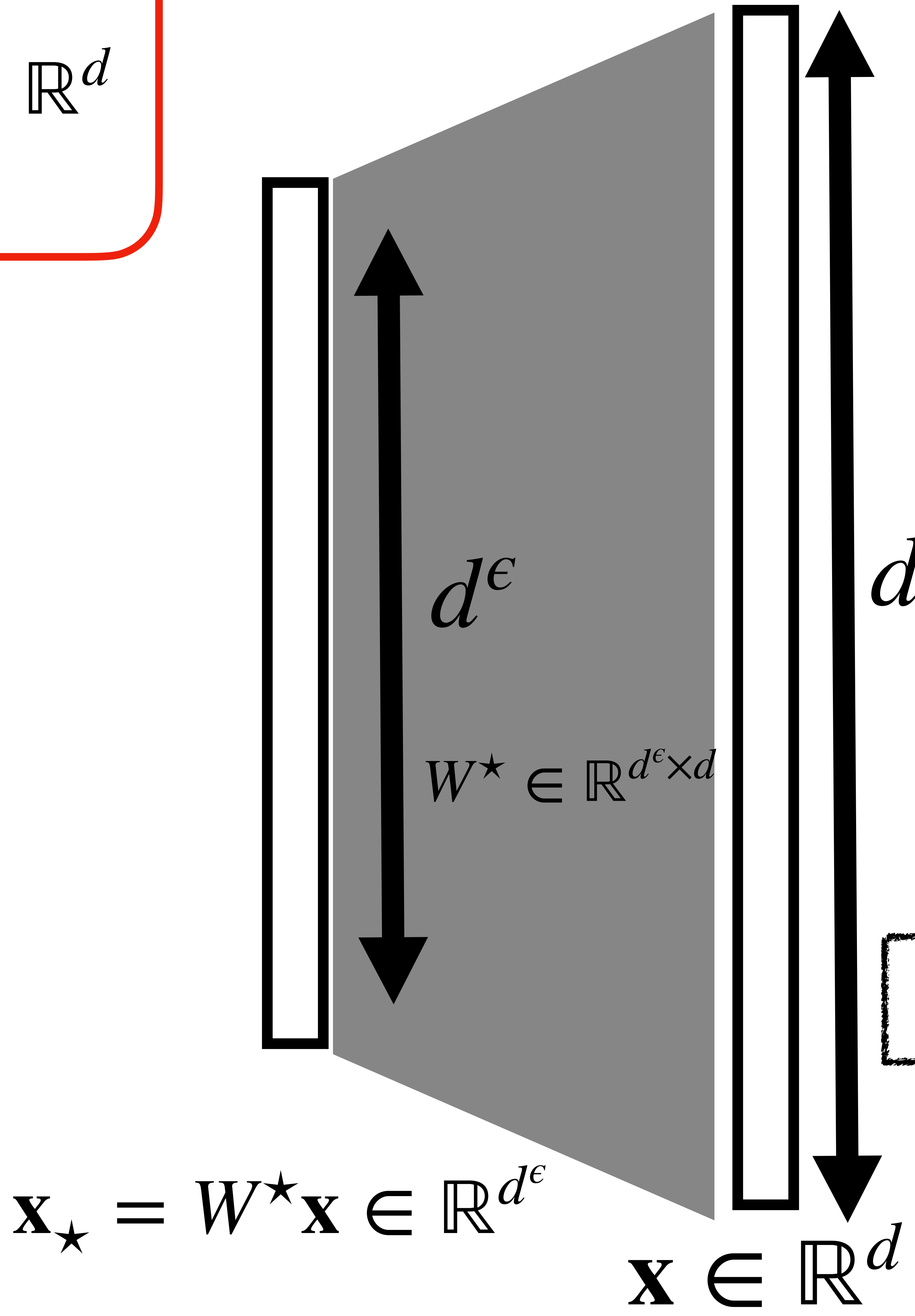
high-dimensional Gaussian
$\mathbf{x} \sim \mathcal{N}(\mathbf{0}, \mathbf{I_d})$

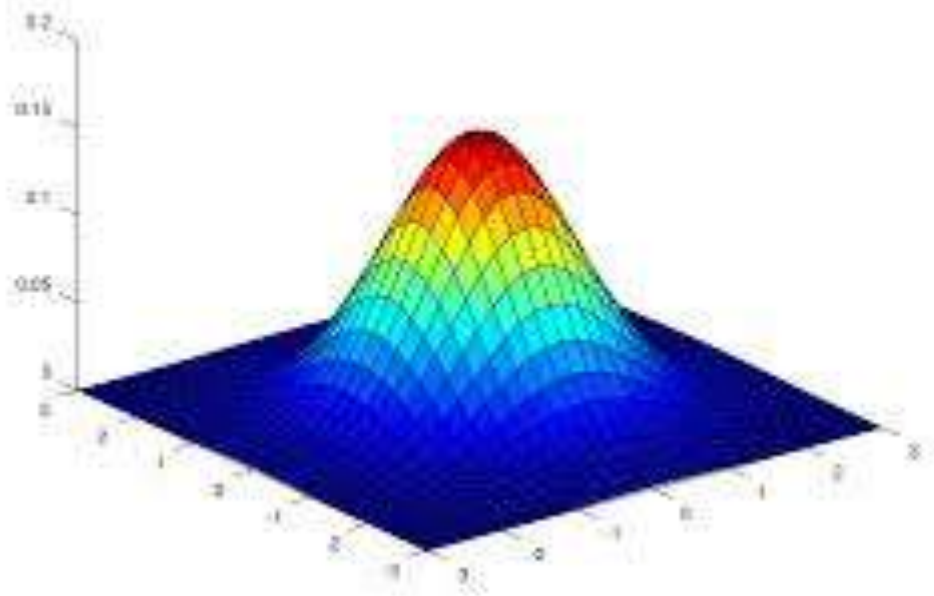
Subspace dimension d^ϵ
for $0 < \epsilon < 1$

A SIGHT beyond Two Layers (Single-Index Gaussian Hierarchical Targets)

$$f^\star(\mathbf{x}) = g^\star\left(\frac{\mathbf{a}^{\star\top} P_k\left(W^\star \mathbf{x}\right)}{\sqrt{d^\varepsilon}}\right), \ \mathbf{x} \in \mathbb{R}^d$$

P_k: polynomial of degree k
$g^\star$: non-linearity





high-dimensional Gaussian
$\mathbf{x} \sim \mathcal{N}(\mathbf{0}, \mathbf{I_d})$

Subspace dimension d^ϵ
for $0 < \epsilon < 1$

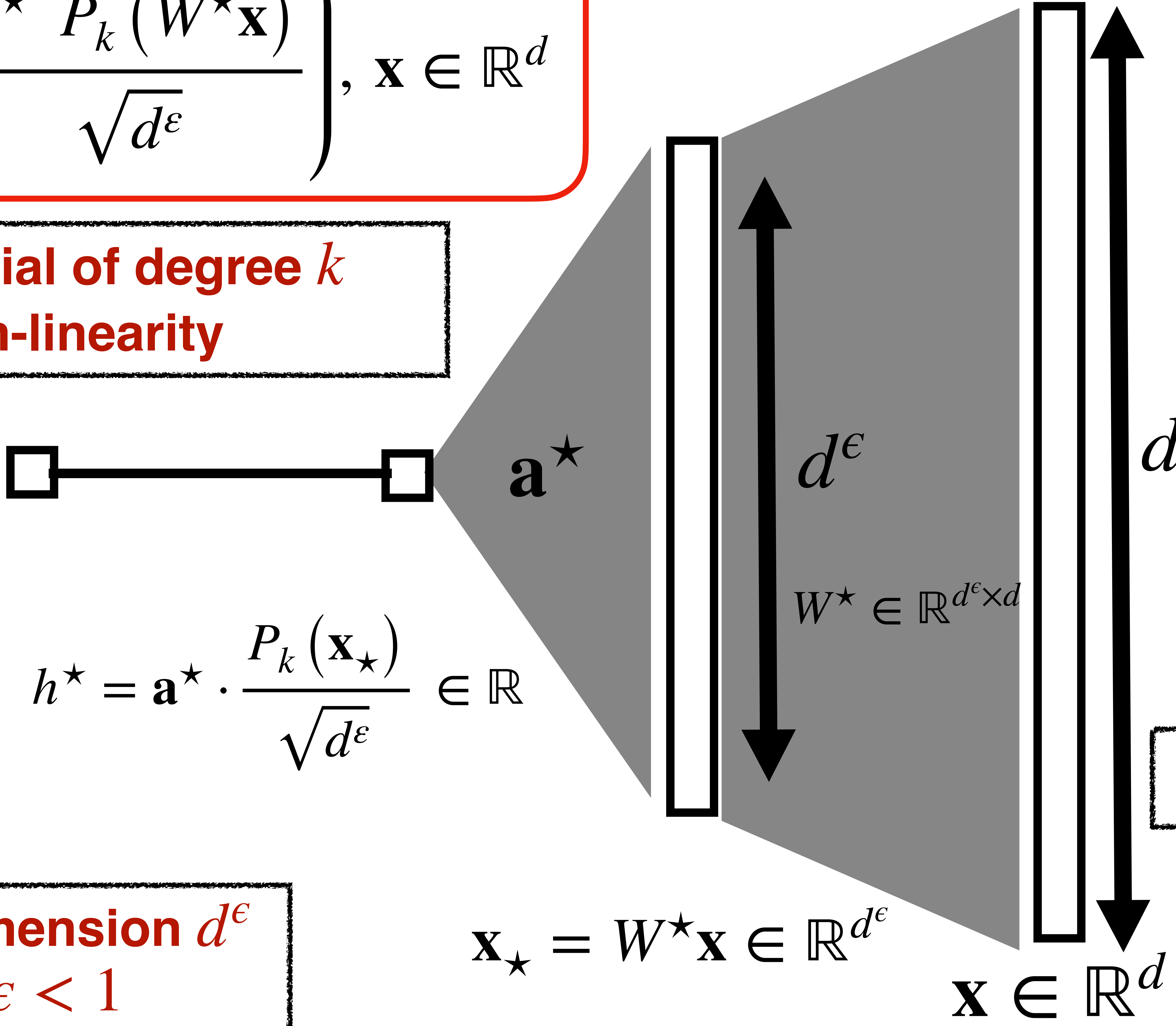
A SIGHT beyond Two Layers (Single-Index Gaussian Hierarchical Targets)

$$f^\star(\mathbf{x}) = g^\star\left(\frac{\mathbf{a}^{\star\top} P_k\left(W^\star \mathbf{x}\right)}{\sqrt{d^\varepsilon}}\right), \mathbf{x} \in \mathbb{R}^d$$

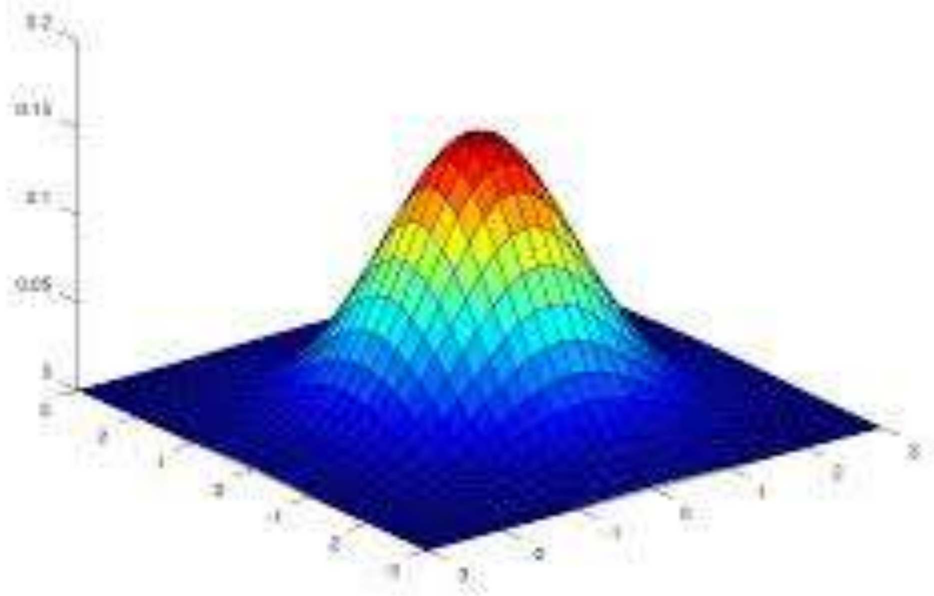
P_k: polynomial of degree k

$g^\star$: non-linearity

$f^\star(\mathbf{x}) = g^\star(h^\star)$



$$h^\star = \mathbf{a}^\star \cdot \frac{P_k\left(\mathbf{x}_\star\right)}{\sqrt{d^\varepsilon}} \in \mathbb{R}$$



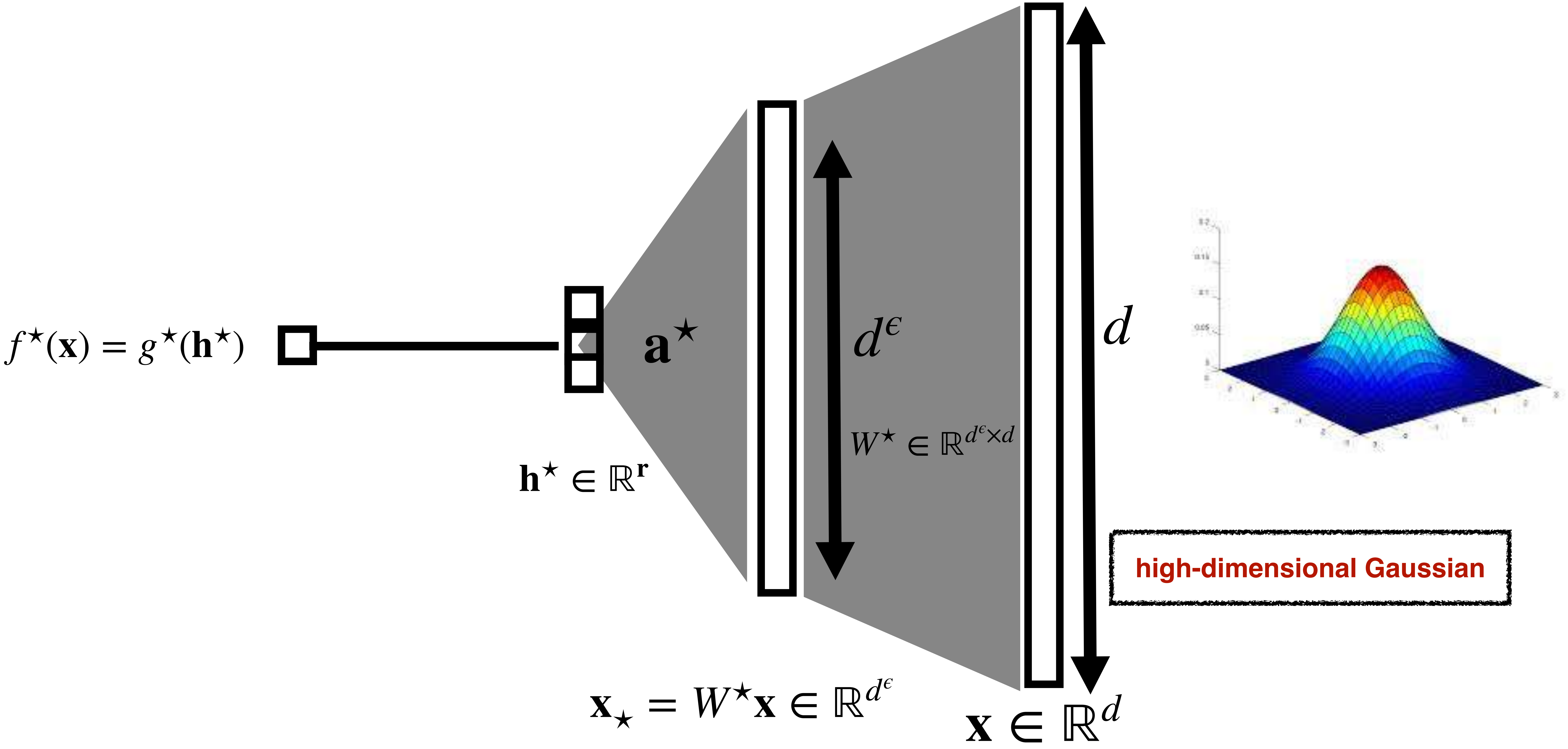
high-dimensional Gaussian
$\mathbf{x} \sim \mathcal{N}(\mathbf{0}, \mathbf{I_d})$

Subspace dimension d^ϵ
for $0 < \epsilon < 1$

MIGHT (Multi-Index Gaussian Hierarchical Targets)

$$f^\star(\mathbf{x}) = g^\star\left(\mathbf{h}_1^\star(\mathbf{x}), \ldots, \mathbf{h}_r^\star(\mathbf{x})\right), \; \mathbf{x} \in \mathbb{R}^d$$

$$h_m^\star(\mathbf{x}) = \frac{\mathbf{1}}{\sqrt{\mathbf{d}^\varepsilon}} \mathbf{a}_\mathbf{m}^{\star\top} \mathbf{P}_{\mathbf{k,m}}\left(\mathbf{W}_\mathbf{m}^\star \mathbf{x}\right), \mathbf{m} = \mathbf{1}, \ldots \mathbf{r}$$



Key features of SIGHTS/MIGHTS

Key features of SIGHTS/MIGHTS

- SIGHT/MIGHT are simply three-layer neural networks.

Key features of SIGHTS/MIGHTS

- SIGHT/MIGHT are simply three-layer neural networks.
- Dimension of features reduces from d to d^{ϵ} to $\mathcal{O}(1)$.

Key features of SIGHTS/MIGHTS

- SIGHT/MIGHT are simply three-layer neural networks.
- Dimension of features reduces from d to d^{ϵ} to $\mathcal{O}(1)$.
- By CLT, $h*$ asymptotically Gaussian and independent.

Key features of SIGHTS/MIGHTS

- SIGHT/MIGHT are simply three-layer neural networks.
- Dimension of features reduces from d to d^{ϵ} to $\mathcal{O}(1)$.
- By CLT, $h*$ asymptotically Gaussian and independent.
- Motivated by Nichani et al. 2023:

Key features of SIGHTS/MIGHTS

- SIGHT/MIGHT are simply three-layer neural networks.
- Dimension of features reduces from d to d^{ϵ} to $\mathcal{O}(1)$.
- By CLT, $h*$ asymptotically Gaussian and independent.
- Motivated by Nichani et al. 2023:

$$f^{\star}(\mathbf{x}) = \mathbf{g}^{\star}(\mathbf{x}^{\top}\mathbf{A}\mathbf{x})$$

Key features of SIGHTS/MIGHTS

- SIGHT/MIGHT are simply three-layer neural networks.
- Dimension of features reduces from d to d^{ϵ} to $\mathcal{O}(1)$.
- By CLT, h^* asymptotically Gaussian and independent.
- Motivated by Nichani et al. 2023:

$$f^\star(\mathbf{x}) = \mathbf{g}^\star(\mathbf{x}^\top \mathbf{A}\mathbf{x}) \longrightarrow$$

Key features of SIGHTS/MIGHTS

- SIGHT/MIGHT are simply three-layer neural networks.
- Dimension of features reduces from d to d^{ϵ} to $\mathcal{O}(1)$.
- By CLT, h^* asymptotically Gaussian and independent.
- Motivated by Nichani et al. 2023:

$$f^{\star}(\mathbf{x}) = \mathbf{g}^{\star}(\mathbf{x}^{\top}\mathbf{A}\mathbf{x}) \longrightarrow$$

2-layer NN can't learn non-linear features

1 level of dimension-reduction

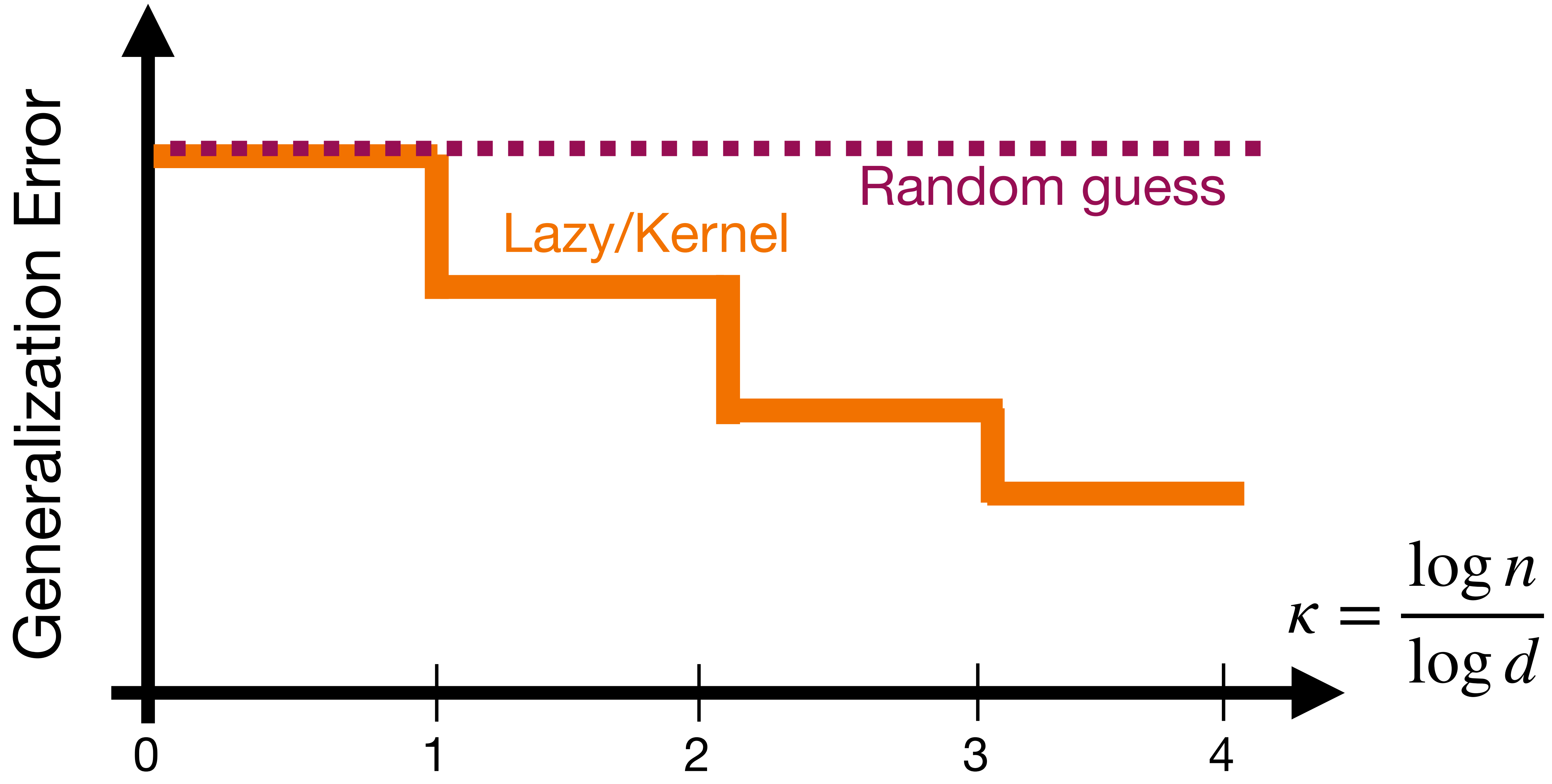
No adaptivity of first layer required

III

Learning SIGHT with Two-layer NNs

Lazy learning (Random Features/NTK)

$$f^\star(\mathbf{x}) = g^\star\left(\frac{\mathbf{a}^{\star\top} P_k\left(W^\star \mathbf{x}\right)}{\sqrt{d^\varepsilon}}\right), \mathbf{x} \in \mathbb{R}^d$$



Learning with a two-layer net

$$f^\star(\mathbf{x}) = g^\star\left(\frac{{\mathbf{a}^\star}^\top P_k\left(W^\star \mathbf{x}\right)}{\sqrt{d^\varepsilon}}\right), \mathbf{x} \in \mathbb{R}^d$$

$$f^\star(\mathbf{x}) = g^\star\left(\frac{{\mathbf{a}^\star}^\top P_k\left(x_\star\right)}{\sqrt{d^\varepsilon}}\right), \mathbf{x}_\star = W^\star \mathbf{x} \in \mathbb{R}^{d^\epsilon}$$

$$\hat{y} = \hat{W}_2 \sigma(\hat{W}_1 \mathbf{x}))$$

Learning with a two-layer net

$$f^\star(\mathbf{x}) = g^\star\left(\frac{\mathbf{a}^{\star\top} P_k\left(W^\star \mathbf{x}\right)}{\sqrt{d^\varepsilon}}\right), \mathbf{x} \in \mathbb{R}^d$$

$$f^\star(\mathbf{x}) = g^\star\left(\frac{\mathbf{a}^{\star\top} P_k\left(x_\star\right)}{\sqrt{d^\varepsilon}}\right), \mathbf{x}_\star = W^\star \mathbf{x} \in \mathbb{R}^{d^\epsilon}$$

parameter count of $W^\star$

$$\hat{y} = \hat{W}_2 \sigma(\hat{W}_1 \mathbf{x})$$

With $n = \mathcal{O}(d^{1+\epsilon})$ **data,** $\hat{W}_1$ **can learn the features with GD :** $\hat{W}_1 \approx Z_1 W^\star + Z_2$

Learning with a two-layer net

$$f^\star(\mathbf{x}) = g^\star\left(\frac{{\mathbf{a}^\star}^\top P_k\left(W^\star \mathbf{x}\right)}{\sqrt{d^\varepsilon}}\right), \mathbf{x} \in \mathbb{R}^d$$

$$f^\star(\mathbf{x}) = g^\star\left(\frac{{\mathbf{a}^\star}^\top P_k\left(x_\star\right)}{\sqrt{d^\varepsilon}}\right), \mathbf{x}_\star = W^\star \mathbf{x} \in \mathbb{R}^{d^\epsilon}$$

parameter count of $W^\star$

$$\hat{y} = \hat{W}_2 \sigma(\hat{W}_1 \mathbf{x}))$$

With $n = \mathcal{O}(d^{1+\epsilon})$ data, $\hat{W}_1$ can learn the features with GD : $\hat{W}_1 \approx Z_1 W^\star + Z_2$

$$\hat{y} \approx \hat{W}_2 \sigma((Z_1 W^\star + Z_2)\mathbf{x}) \approx \hat{W}_2 \sigma(Z_1 x_\star + Z_3)$$

Learning with a two-layer net

$$f^\star(\mathbf{x}) = g^\star\left(\frac{\mathbf{a}^{\star\top} P_k\left(W^\star \mathbf{x}\right)}{\sqrt{d^\varepsilon}}\right), \mathbf{x} \in \mathbb{R}^d$$

$$f^\star(\mathbf{x}) = g^\star\left(\frac{\mathbf{a}^{\star\top} P_k\left(x_\star\right)}{\sqrt{d^\varepsilon}}\right), \mathbf{x}_\star = W^\star \mathbf{x} \in \mathbb{R}^{d^\epsilon}$$

parameter count of $W^\star$

$$\hat{y} = \hat{W}_2 \sigma(\hat{W}_1 \mathbf{x})$$

With $n = \mathcal{O}(d^{1+\epsilon})$ data, $\hat{W}_1$ can learn the features with GD : $\hat{W}_1 \approx Z_1 W^\star + Z_2$

$$\hat{y} \approx \hat{W}_2 \sigma((Z_1 W^\star + Z_2)\mathbf{x}) \approx \hat{W}_2 \sigma(Z_1 x_\star + Z_3)$$

random feature/lazy learning but in reduced dimension $\mathbb{R}^{d^\epsilon}$

Learning with a two-layer net

$$f^\star(\mathbf{x}) = g^\star\left(\frac{\mathbf{a}^{\star\top} P_k\left(W^\star \mathbf{x}\right)}{\sqrt{d^\varepsilon}}\right), \mathbf{x} \in \mathbb{R}^d$$

$$f^\star(\mathbf{x}) = g^\star\left(\frac{\mathbf{a}^{\star\top} P_k\left(x_\star\right)}{\sqrt{d^\varepsilon}}\right), \mathbf{x}_\star = W^\star \mathbf{x} \in \mathbb{R}^{d^\epsilon}$$

parameter count of $W^\star$

$$\hat{y} = \hat{W}_2 \sigma(\hat{W}_1 \mathbf{x}))$$

With $n = \mathcal{O}(d^{1+\epsilon})$ data, $\hat{W}_1$ can learn the features with GD : $\hat{W}_1 \approx Z_1 W^\star + Z_2$

$$\hat{y} \approx \hat{W}_2 \sigma((Z_1 W^\star + Z_2)\mathbf{x}) \approx \hat{W}_2 \sigma(Z_1 x_\star + Z_3)$$

random feature/lazy learning but in reduced dimension $\mathbb{R}^{d^\epsilon}$

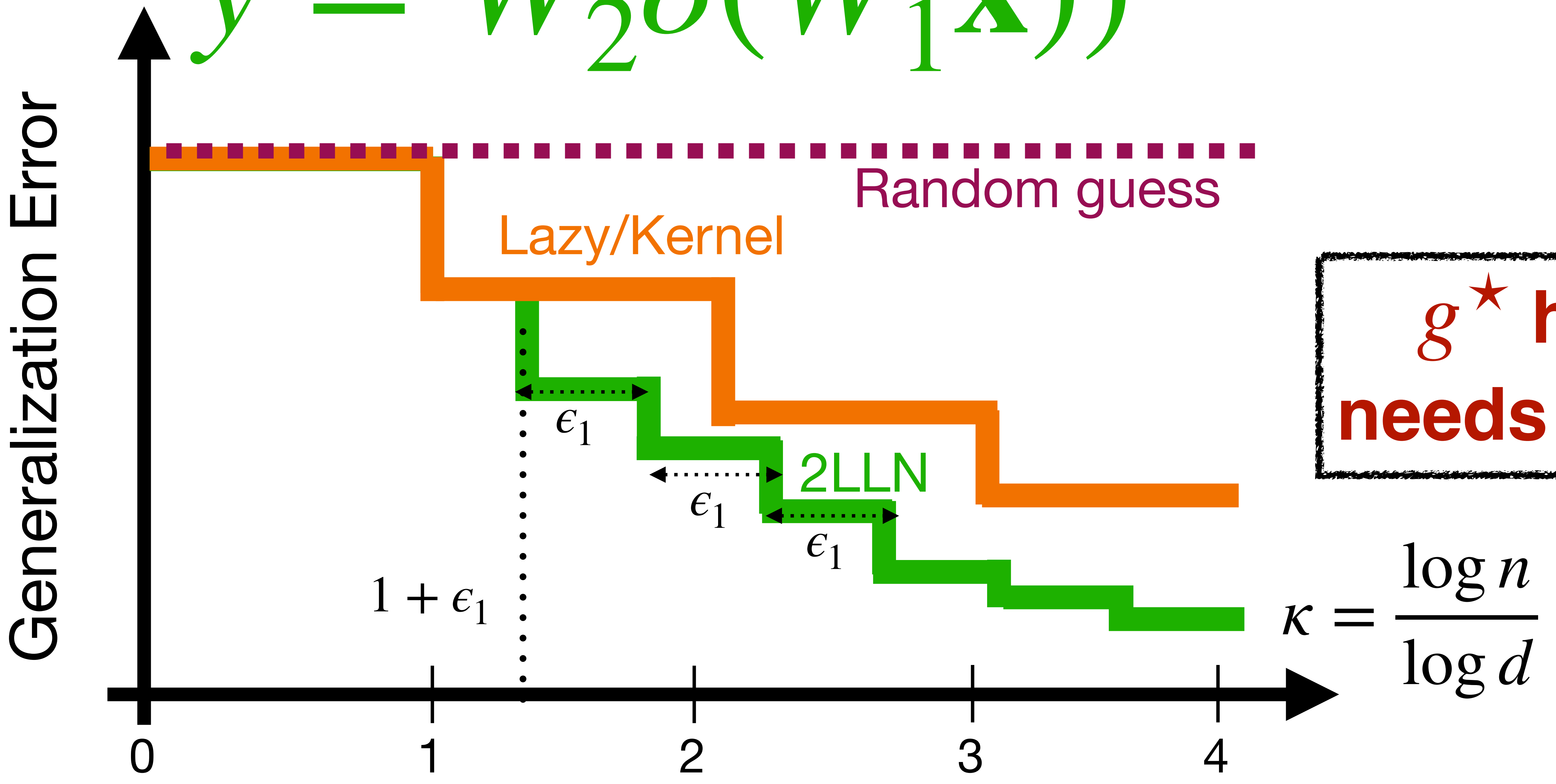
Adaptive Learning with a two-layer net

$$f^\star(\mathbf{x}) = g^\star\left(\frac{\mathbf{a}^{\star\top} P_k\left(W^\star \mathbf{x}\right)}{\sqrt{d^\varepsilon}}\right), \mathbf{x} \in \mathbb{R}^d$$

$$f^\star(\mathbf{x}) = g^\star\left(\frac{\mathbf{a}^{\star\top} P_k\left(x_\star\right)}{\sqrt{d^{\varepsilon_1}}}\right), \mathbf{x}_\star \in \mathbb{R}^{d^{\epsilon_1}}$$

$$\hat{y} = \hat{W}_2 \sigma(\hat{W}_1 \mathbf{x}))$$

Only 1 level of dimension reduction



$g^\star$ has degree m $\Longrightarrow$ needs sample, width $d^{km\epsilon_1}$

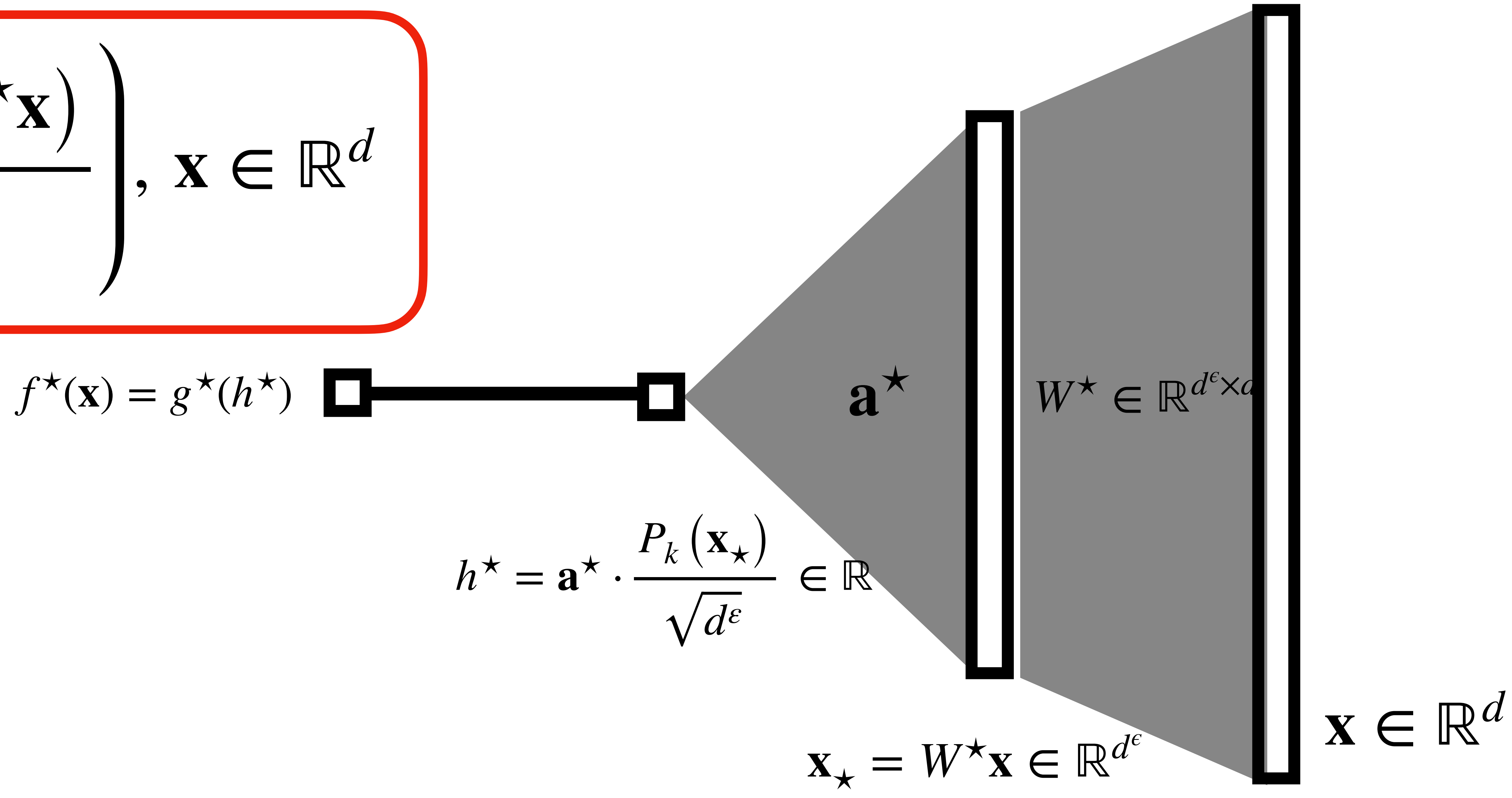
random feature/lazy learning but in reduced dimension $\mathbb{R}^{d^\epsilon}$

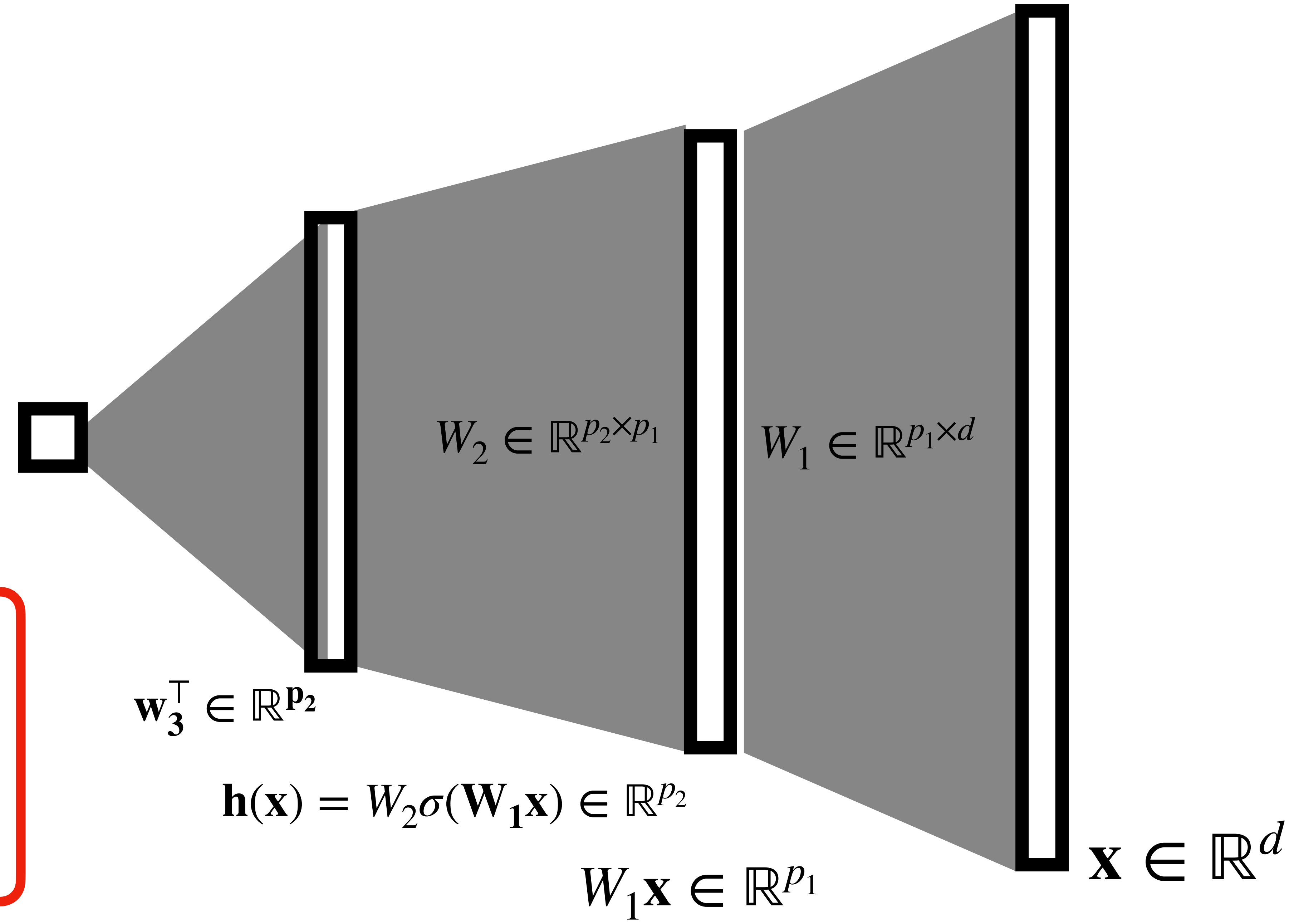
IV

Learning SIGHT with Three-layer NNs

Idealized scenario for layer-wise training in 3-layer NNS

$$f^\star(\mathbf{x}) = g^\star\left(\frac{\mathbf{a}^{\star\top} P_k\left(W^\star \mathbf{x}\right)}{\sqrt{d^{\varepsilon_1}}}\right), \mathbf{x} \in \mathbb{R}^d$$

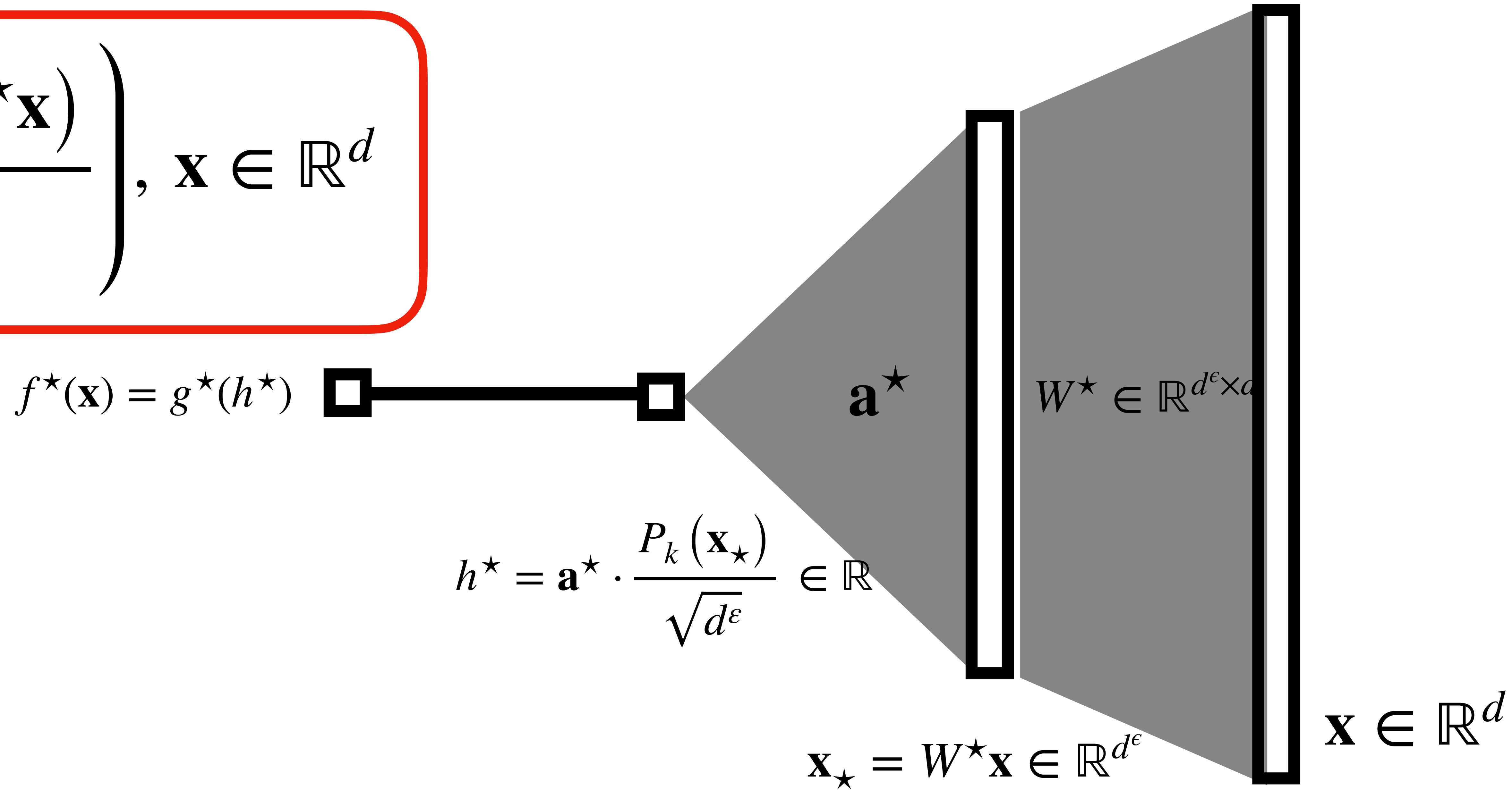


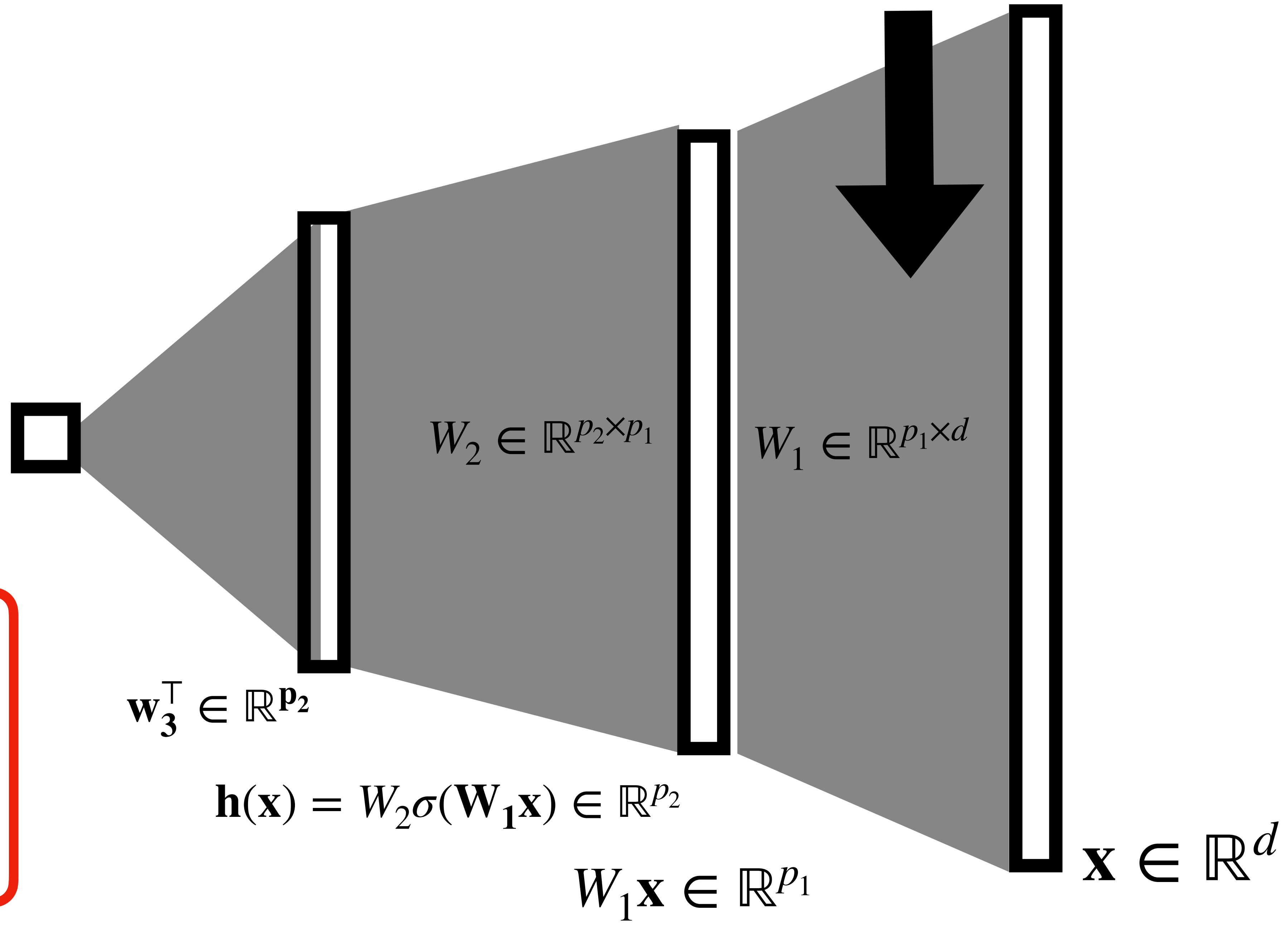


$$\hat{f}(\mathbf{x}) = \mathbf{w_3}^\top \sigma\left(W_2\sigma(W_1\mathbf{x} + b_1) + b_2\right)$$

Idealized scenario for layer-wise training in 3-layer NNS

$$f^\star(\mathbf{x}) = g^\star\left(\frac{\mathbf{a}^{\star\top} P_k\left(W^\star\mathbf{x}\right)}{\sqrt{d^{\varepsilon_1}}}\right),\ \mathbf{x}\in\mathbb{R}^d$$

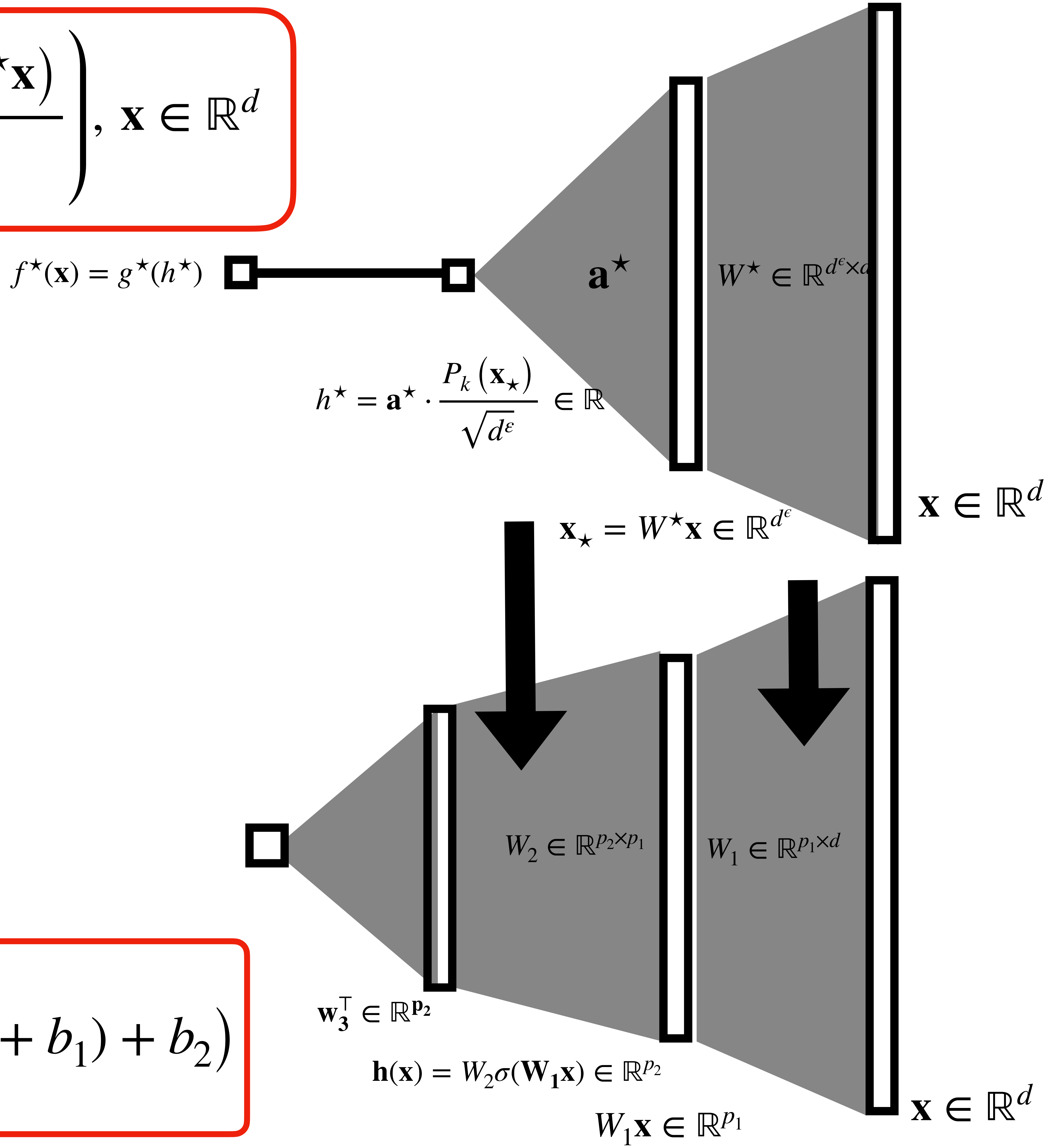




$$\hat{f}(\mathbf{x}) = \mathbf{w_3}^\top\sigma\left(W_2\sigma(W_1\mathbf{x}+b_1)+b_2\right)$$

Idealized scenario for layer-wise training in 3-layer NNS

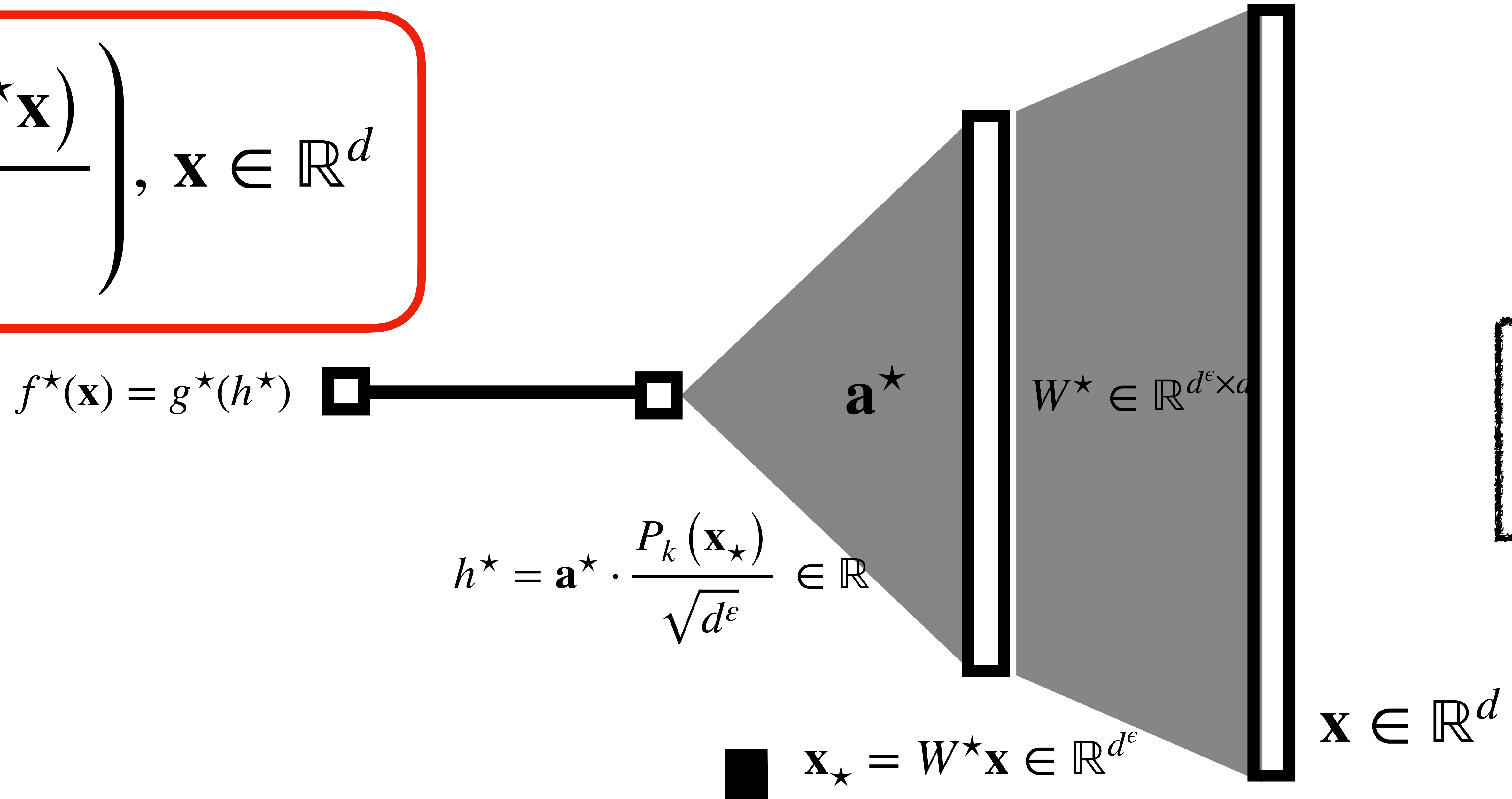
$$f^\star(\mathbf{x}) = g^\star\left(\frac{\mathbf{a}^{\star\top} P_k\left(W^\star \mathbf{x}\right)}{\sqrt{d^{\varepsilon_1}}}\right), \quad \mathbf{x} \in \mathbb{R}^d$$



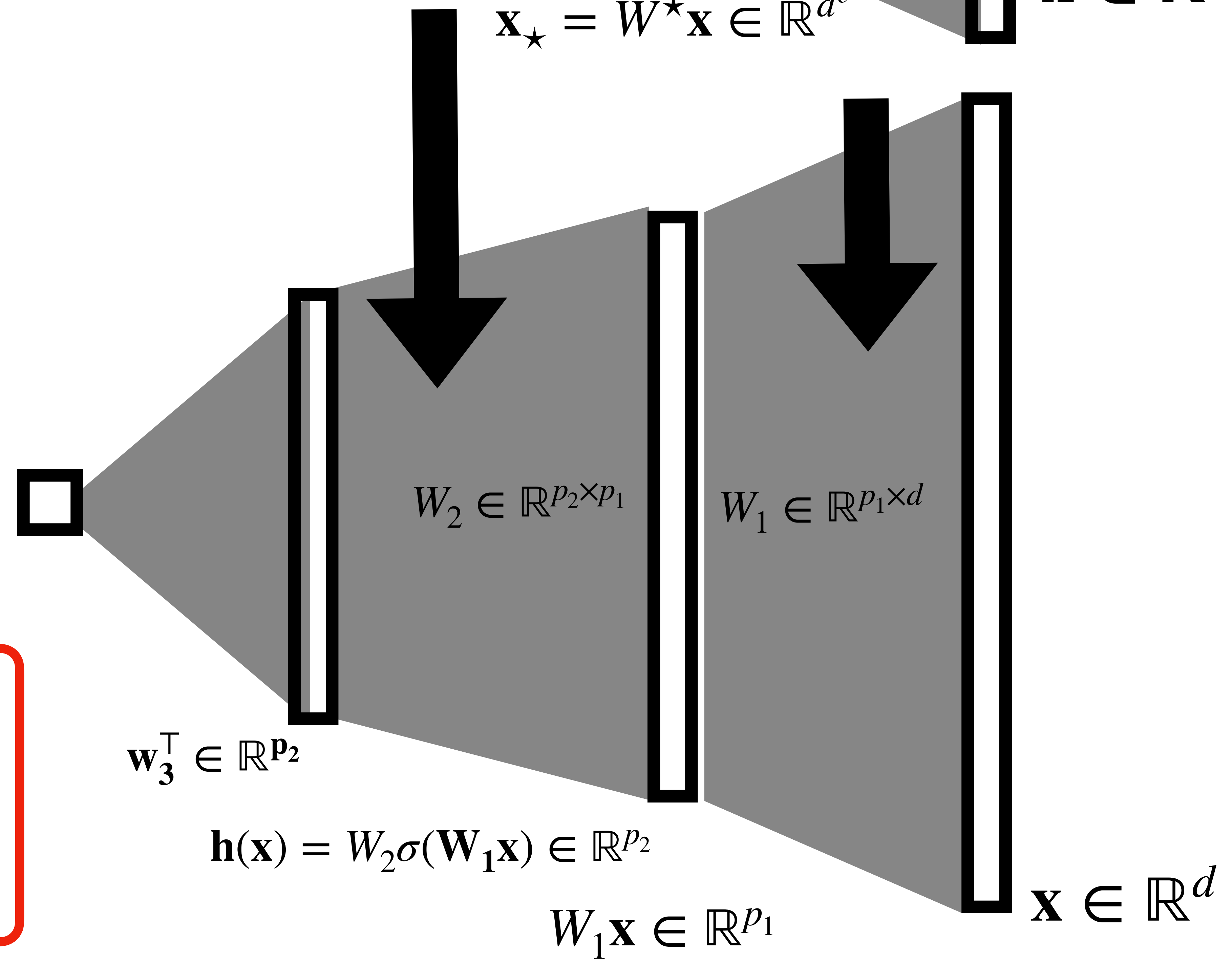
$$\hat{f}(\mathbf{x}) = \mathbf{w}_3{}^\top \sigma\left(W_2\sigma(W_1\mathbf{x} + b_1) + b_2\right)$$

Idealized scenario for layer-wise training in 3-layer NNS

$$f^\star(\mathbf{x}) = g^\star\left(\frac{\mathbf{a}^{\star\top} P_k\left(W^\star \mathbf{x}\right)}{\sqrt{d^{\varepsilon_1}}}\right), \mathbf{x} \in \mathbb{R}^d$$



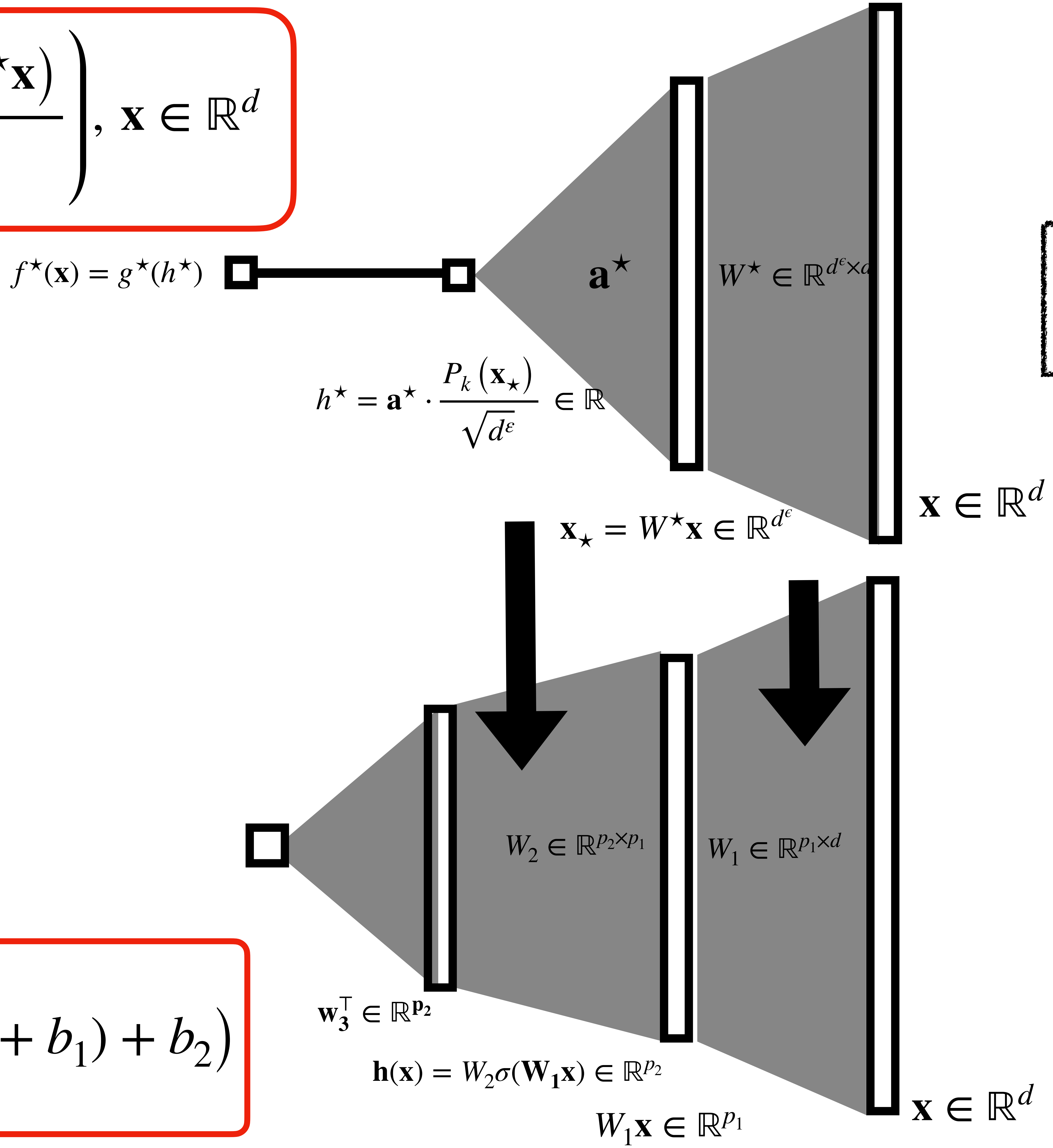
W_2 learns features not weights $a^\star$



$$\hat{f}(\mathbf{x}) = \mathbf{w_3}^\top \sigma\left(W_2\sigma(W_1\mathbf{x} + b_1) + b_2\right)$$

Idealized scenario for layer-wise training in 3-layer NNS

$$f^\star(\mathbf{x}) = g^\star\left(\frac{\mathbf{a}^{\star\top} P_k\left(W^\star \mathbf{x}\right)}{\sqrt{d^{\varepsilon_1}}}\right), \ \mathbf{x} \in \mathbb{R}^d$$

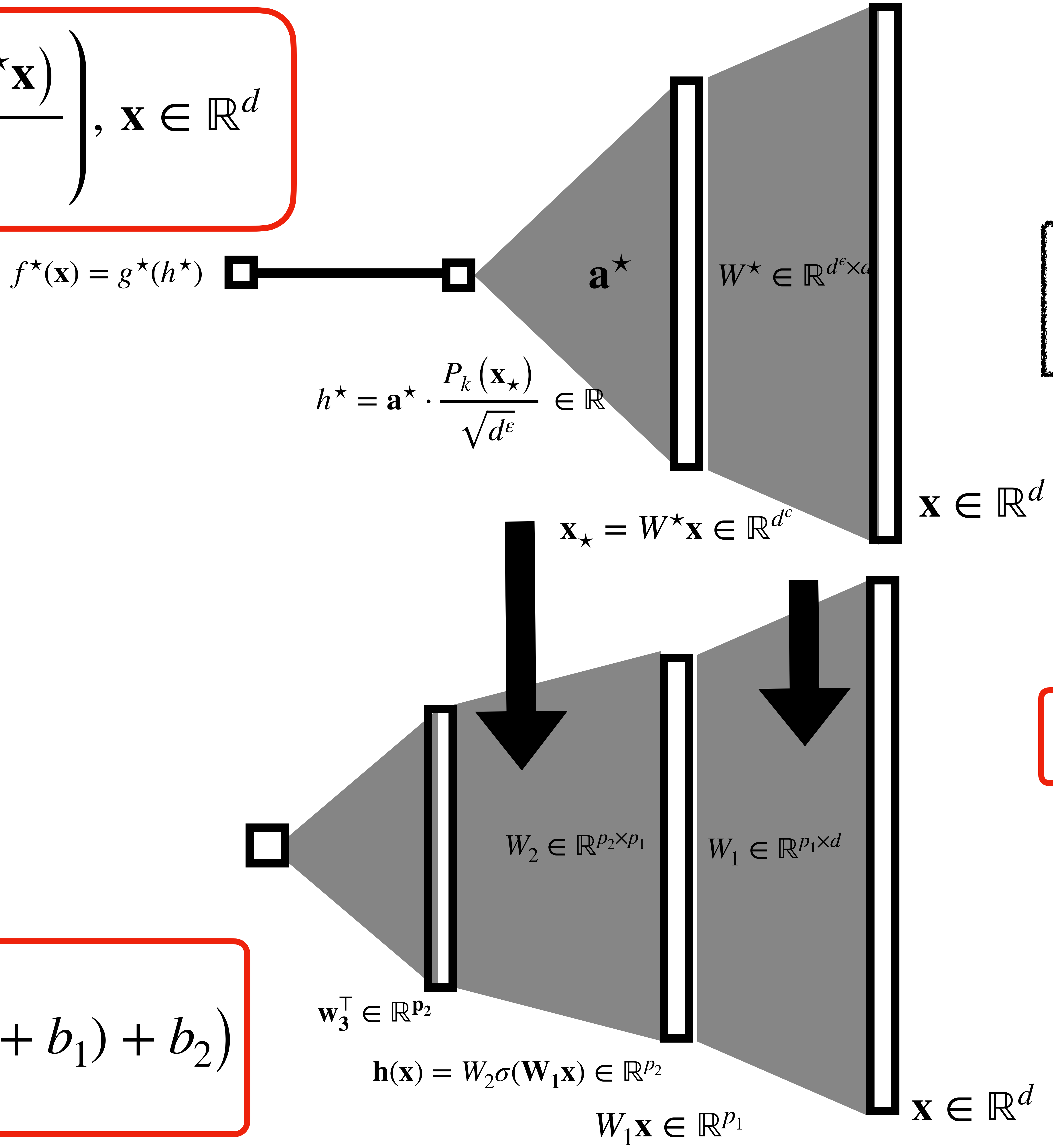


W_2 learns features not weights $a^\star$

$$\hat{f}(\mathbf{x}) = \mathbf{w_3}^\top \sigma\left(W_2\sigma(W_1\mathbf{x} + b_1) + b_2\right)$$

Idealized scenario for layer-wise training in 3-layer NNS

$$f^\star(\mathbf{x}) = g^\star\left(\frac{\mathbf{a}^{\star\top} P_k\left(W^\star \mathbf{x}\right)}{\sqrt{d^{\varepsilon_1}}}\right), \mathbf{x} \in \mathbb{R}^d$$



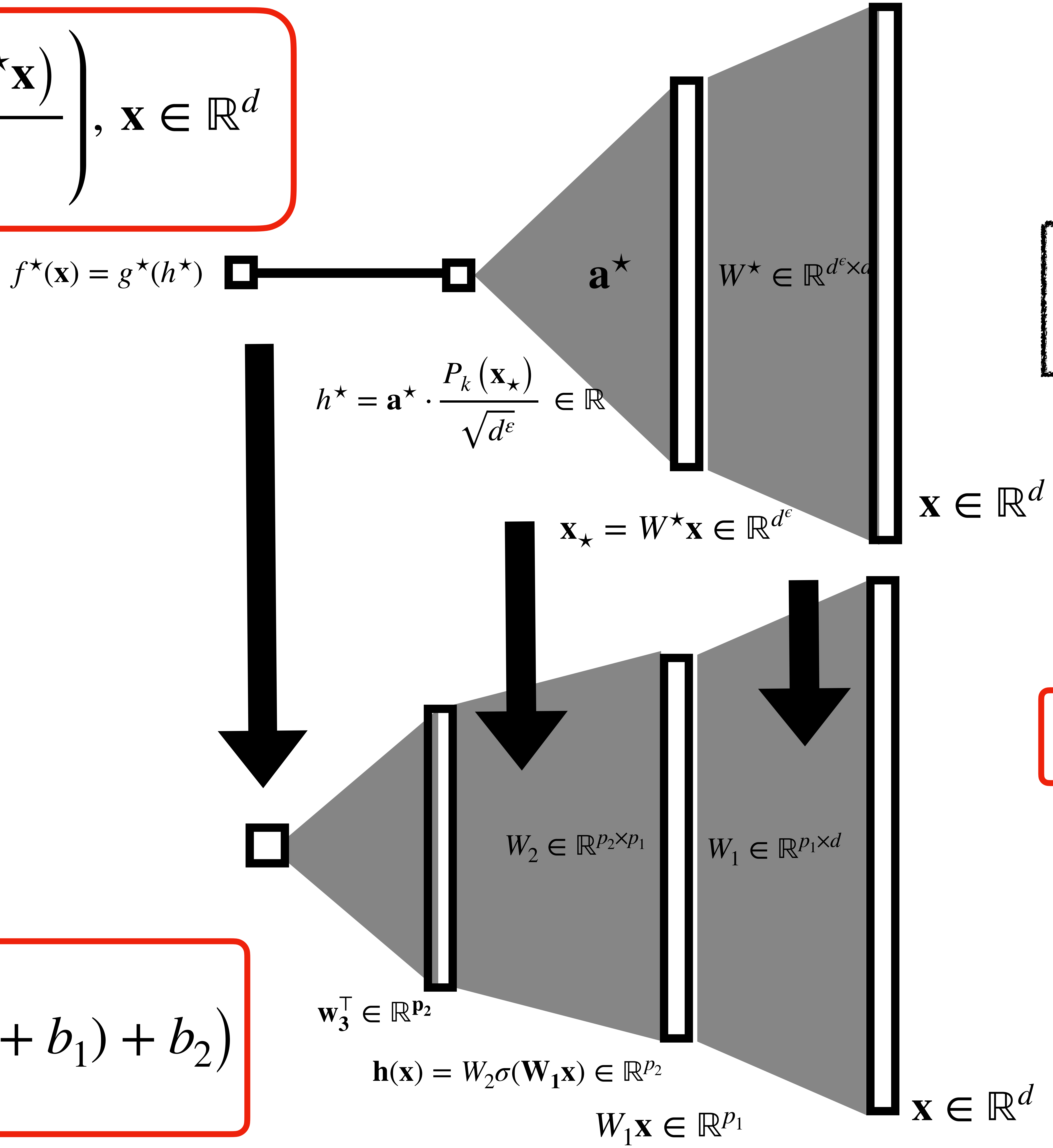
W_2 learns features not weights $a^\star$

$$\mathbf{h}(\mathbf{x}) = W_2\sigma(\mathbf{W}_1\mathbf{x}) \propto h^\star(\mathbf{x})$$

$$\hat{f}(\mathbf{x}) = \mathbf{w}_3{}^\top\sigma\left(W_2\sigma(W_1\mathbf{x} + b_1) + b_2\right)$$

Idealized scenario for layer-wise training in 3-layer NNS

$$f^\star(\mathbf{x}) = g^\star\left(\frac{\mathbf{a}^{\star\top} P_k\left(W^\star \mathbf{x}\right)}{\sqrt{d^{\varepsilon_1}}}\right), \mathbf{x} \in \mathbb{R}^d$$



W_2 learns features not weights $a^\star$

$$\mathbf{h}(\mathbf{x}) = W_2\sigma(\mathbf{W}_1\mathbf{x}) \propto h^\star(\mathbf{x})$$

$$\hat{f}(\mathbf{x}) = \mathbf{w}_3{}^\top\sigma\left(W_2\sigma(W_1\mathbf{x} + b_1) + b_2\right)$$

Expected behavior under idealized scenario

$$f^\star(\mathbf{x}) = g^\star\left(\frac{\mathbf{a}^{\star\top} P_k\left(W^\star \mathbf{x}\right)}{\sqrt{d^{\varepsilon_1}}}\right), \mathbf{x} \in \mathbb{R}^d$$

$$f^\star(\mathbf{x}) = g^\star\left(\frac{\mathbf{a}^{\star\top} P_k\left(x_\star\right)}{\sqrt{d^{\varepsilon}}}\right), \mathbf{x}_\star \in \mathbb{R}^{d^\epsilon}$$

$$\hat{y} = \hat{\mathbf{W}}^3 \sigma(\hat{W}_2 \sigma(\hat{W}_1 \mathbf{x}))$$

Expected behavior under idealized scenario

$$f^\star(\mathbf{x}) = g^\star\left(\frac{\mathbf{a}^{\star\top} P_k\left(W^\star \mathbf{x}\right)}{\sqrt{d^{\varepsilon_1}}}\right), \mathbf{x} \in \mathbb{R}^d$$

$$f^\star(\mathbf{x}) = g^\star\left(\frac{\mathbf{a}^{\star\top} P_k\left(x_\star\right)}{\sqrt{d^{\varepsilon}}}\right), \mathbf{x}_\star \in \mathbb{R}^{d^\epsilon}$$

$$\hat{y} = \hat{\mathbf{W}}^3 \sigma(\hat{W}_2 \sigma(\hat{W}_1 \mathbf{x}))$$

With $n = \mathcal{O}(d^{1+\epsilon})$ **data,** $\hat{W}_1$ **can learn the features with GD :** $\hat{W}_1 \approx Z_1 W^\star + Z_2$

$$\hat{y} \approx \hat{\mathbf{w}}_\mathbf{3}{}^\top \sigma(\hat{W}_2 \sigma(Z_1 x_\star + Z_2))$$

Expected behavior under idealized scenario

$$f^\star(\mathbf{x}) = g^\star\left(\frac{\mathbf{a}^{\star\top} P_k\left(W^\star\mathbf{x}\right)}{\sqrt{d^{\varepsilon_1}}}\right), \ \mathbf{x} \in \mathbb{R}^d$$

$$f^\star(\mathbf{x}) = g^\star\left(\frac{\mathbf{a}^{\star\top} P_k\left(x_\star\right)}{\sqrt{d^{\varepsilon}}}\right), \ \mathbf{x}_\star \in \mathbb{R}^{d^\epsilon}$$

$$\hat{y} = \hat{\mathbf{W}}^3\sigma(\hat{W}_2\sigma(\hat{W}_1\mathbf{x}))$$

With $n = \mathcal{O}(d^{1+\epsilon})$ **data,** $\hat{W}_1$ **can learn the features with GD :** $\hat{W}_1 \approx Z_1 W^\star + Z_2$

With each $n = \mathcal{O}(d^{\epsilon})$ **data,** $\hat{W}_2\sigma(\,.\,)$ **learns a higher-degree component of** P_k

$$\hat{y} \approx \hat{\mathbf{w}}_\mathbf{3}{}^\top\sigma(\hat{W}_2\sigma(Z_1 x_\star + Z_2))$$

Expected behavior under idealized scenario

$$f^\star(\mathbf{x}) = g^\star\left(\frac{\mathbf{a}^{\star\top} P_k\left(W^\star \mathbf{x}\right)}{\sqrt{d^{\varepsilon_1}}}\right), \mathbf{x} \in \mathbb{R}^d$$

$$f^\star(\mathbf{x}) = g^\star\left(\frac{\mathbf{a}^{\star\top} P_k\left(x_\star\right)}{\sqrt{d^{\varepsilon}}}\right), \mathbf{x}_\star \in \mathbb{R}^{d^\epsilon}$$

$$\hat{y} = \hat{\mathbf{W}}^3 \sigma(\hat{W}_2 \sigma(\hat{W}_1 \mathbf{x}))$$

With $n = \mathcal{O}(d^{1+\epsilon})$ **data,** $\hat{W}_1$ **can learn the features with GD :** $\hat{W}_1 \approx Z_1 W^\star + Z_2$

With each $n = \mathcal{O}(d^\epsilon)$ **data,** $\hat{W}_2 \sigma(\,.\,)$ **learns a higher-degree component of** P_k

$$\hat{y} \approx \hat{\mathbf{w}}_3{}^\top \sigma(\hat{W}_2 \sigma(Z_1 x_\star + Z_2))$$

$$\approx \hat{\mathbf{w}}_3{}^\top \sigma(Z_1 h^\star + Z_2)$$

Expected behavior under idealized scenario

$$f^\star(\mathbf{x}) = g^\star\left(\frac{\mathbf{a}^{\star\top} P_k\left(W^\star \mathbf{x}\right)}{\sqrt{d^{\varepsilon_1}}}\right), \ \mathbf{x} \in \mathbb{R}^d$$

$$f^\star(\mathbf{x}) = g^\star\left(h^\star\right), h^\star = \mathbf{a}^\star \cdot \frac{P_k\left(\mathbf{x}_\star\right)}{\sqrt{d^{\varepsilon}}} \in \mathbb{R}$$

$$\hat{y} = \hat{\mathbf{W}}^3 \sigma(\hat{W}_2 \sigma(\hat{W}_1 \mathbf{x}))$$

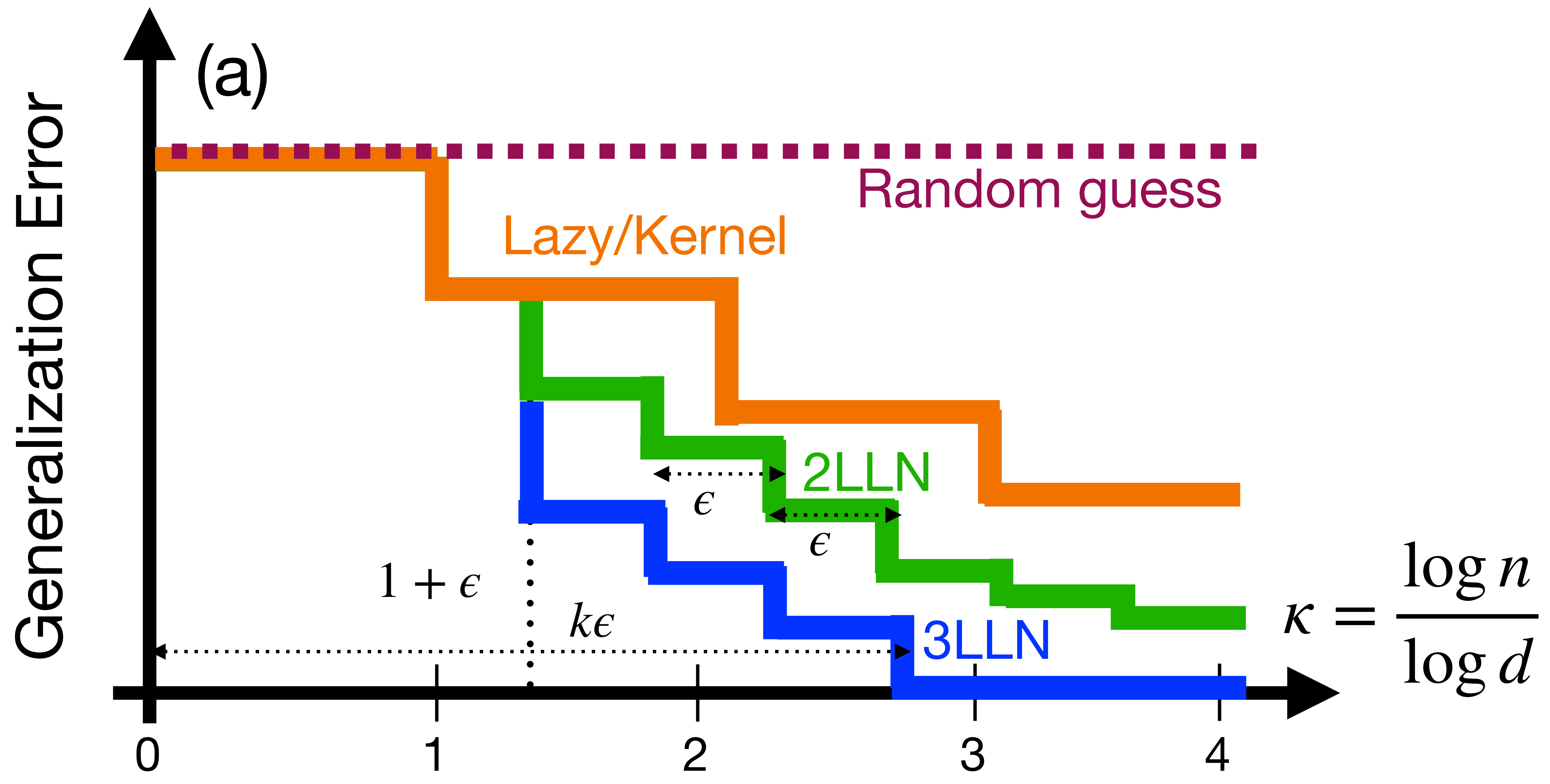
With $n = \mathcal{O}(d^{1+\epsilon})$ **data,** $\hat{W}_1$ **can learn the features with GD :** $\hat{W}_1 \approx Z_1 W^\star + Z_2$

With each $n = \mathcal{O}(d^{\epsilon})$ **data,** $\hat{W}_2 \sigma(\,.\,)$ **learns a higher-degree component of** P_k

$$\hat{y} \approx \hat{\mathbf{w}}_3{}^\top \sigma(\hat{W}_2 \sigma(Z_1 x_\star + Z_2))$$

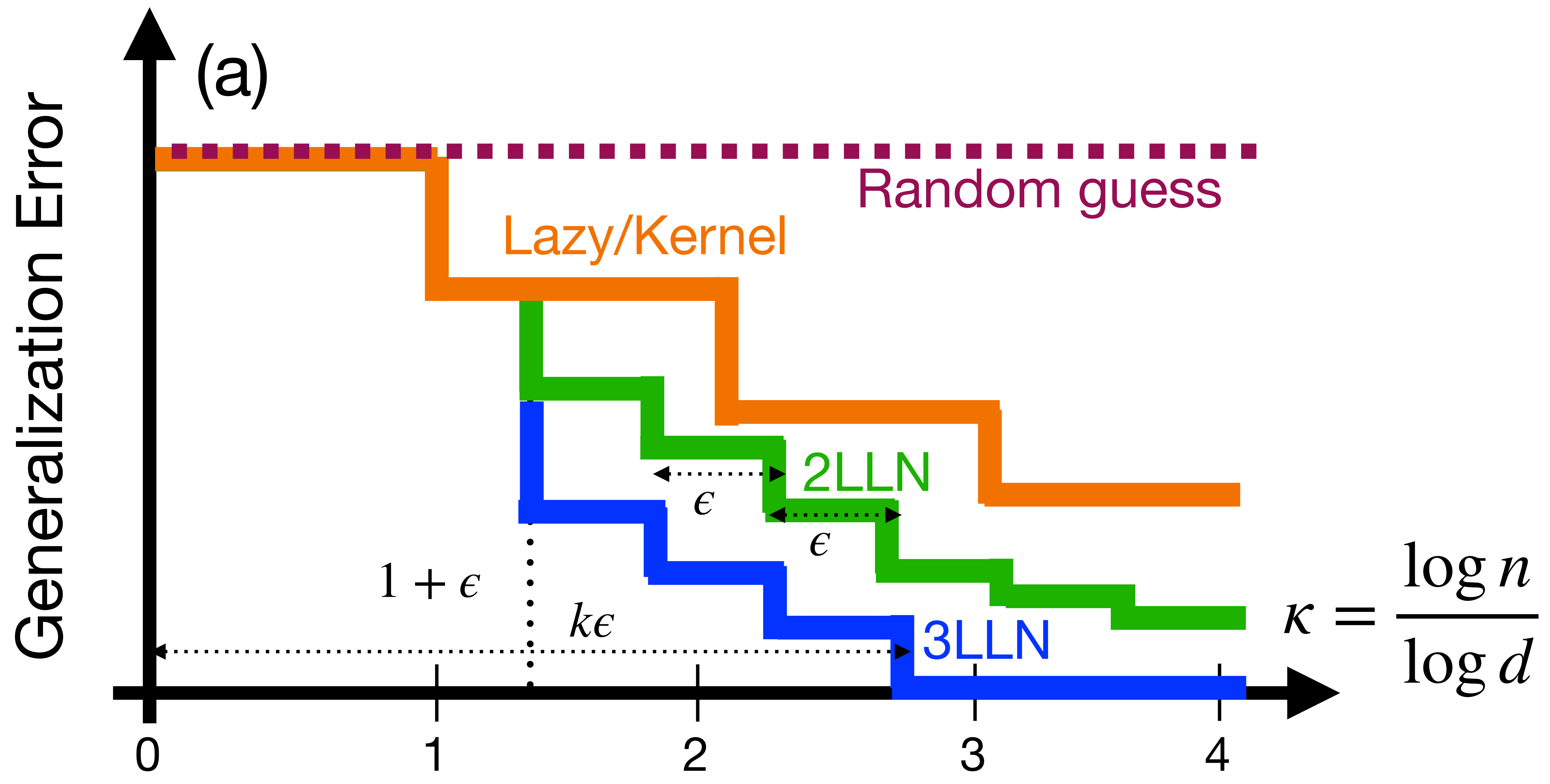
$$\approx \hat{\mathbf{w}}_3{}^\top \sigma(Z_1 h^\star + Z_2)$$

Two levels of dimension reduction



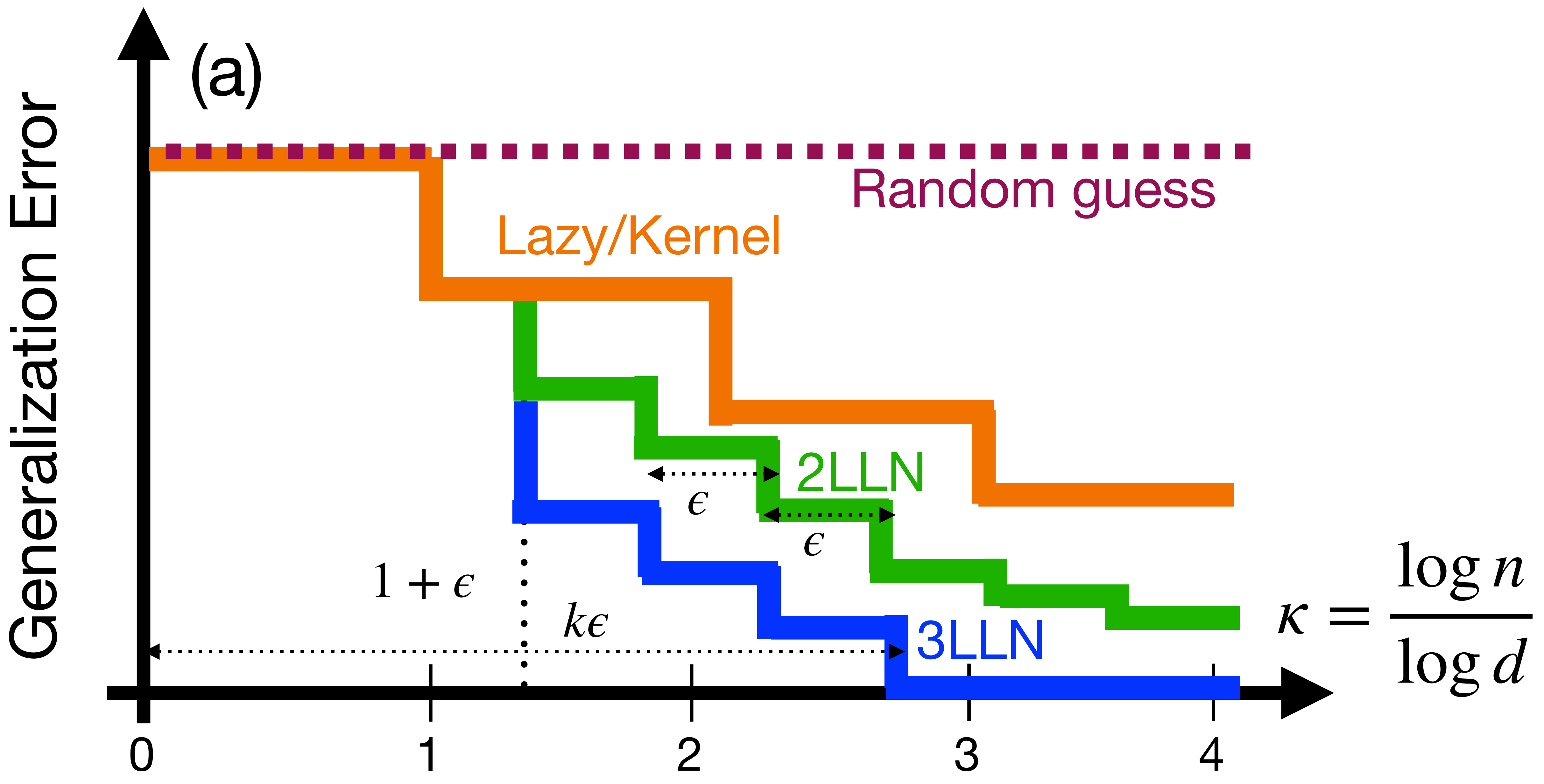
Two levels of dimension reduction

$$f^\star(\mathbf{x}) = \mathbf{g}^\star\left(\frac{\mathbf{a}^{\star\top}\mathbf{P}_\mathbf{k}\left(\mathbf{W}^\star\mathbf{x}\right)}{\sqrt{\mathbf{d}^\varepsilon}}\right), \mathbf{x} \in \mathbb{R}^\mathbf{d}$$



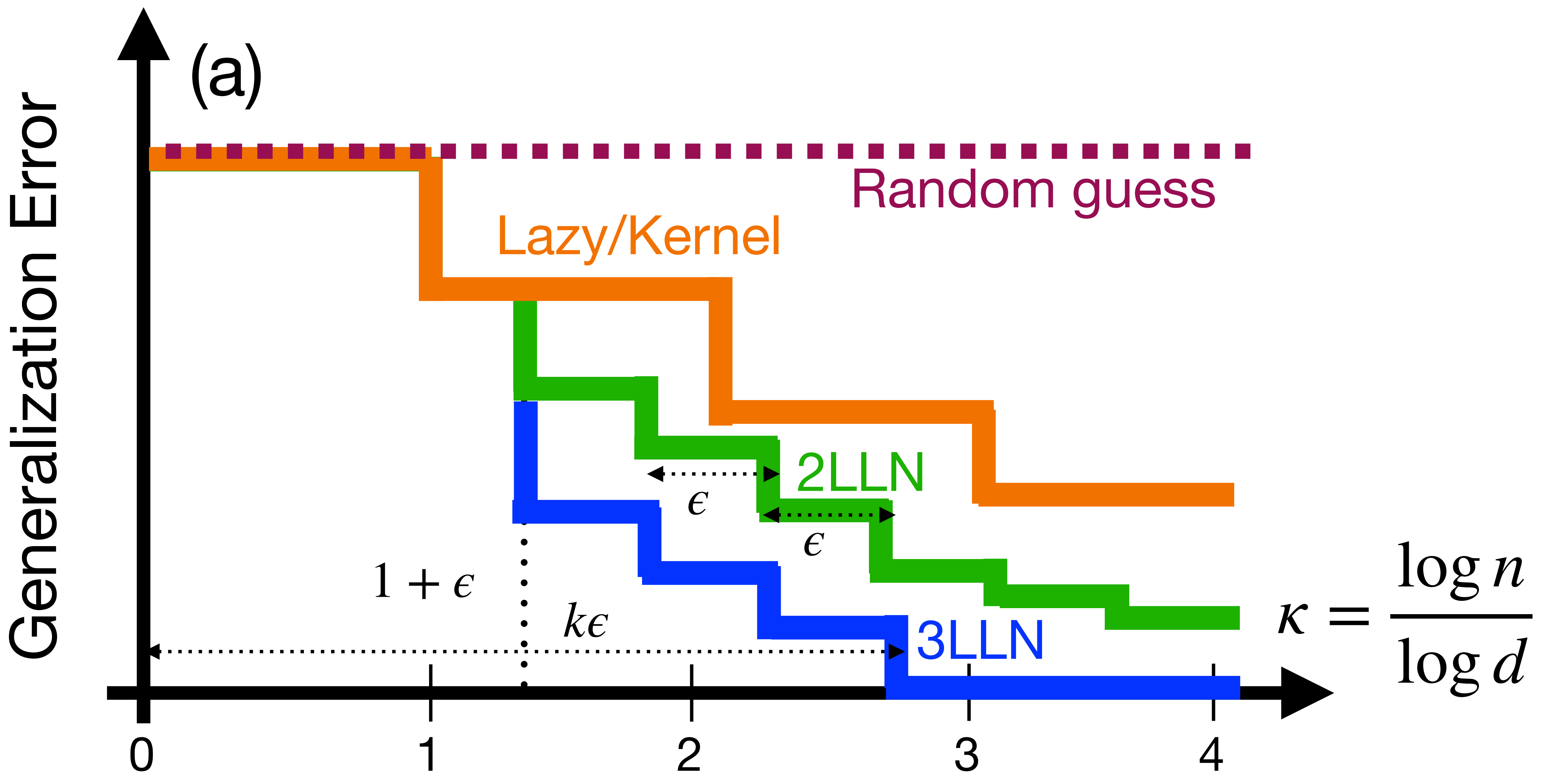
Two levels of dimension reduction

$$f^{\star}(\mathbf{x}) = \mathbf{g}^{\star}\left(\frac{\mathbf{a}^{\star\top}\mathbf{P_k}\left(\mathbf{W}^{\star}\mathbf{x}\right)}{\sqrt{\mathbf{d}^{\varepsilon}}}\right), \ \mathbf{x} \in \mathbb{R}^{\mathbf{d}}$$



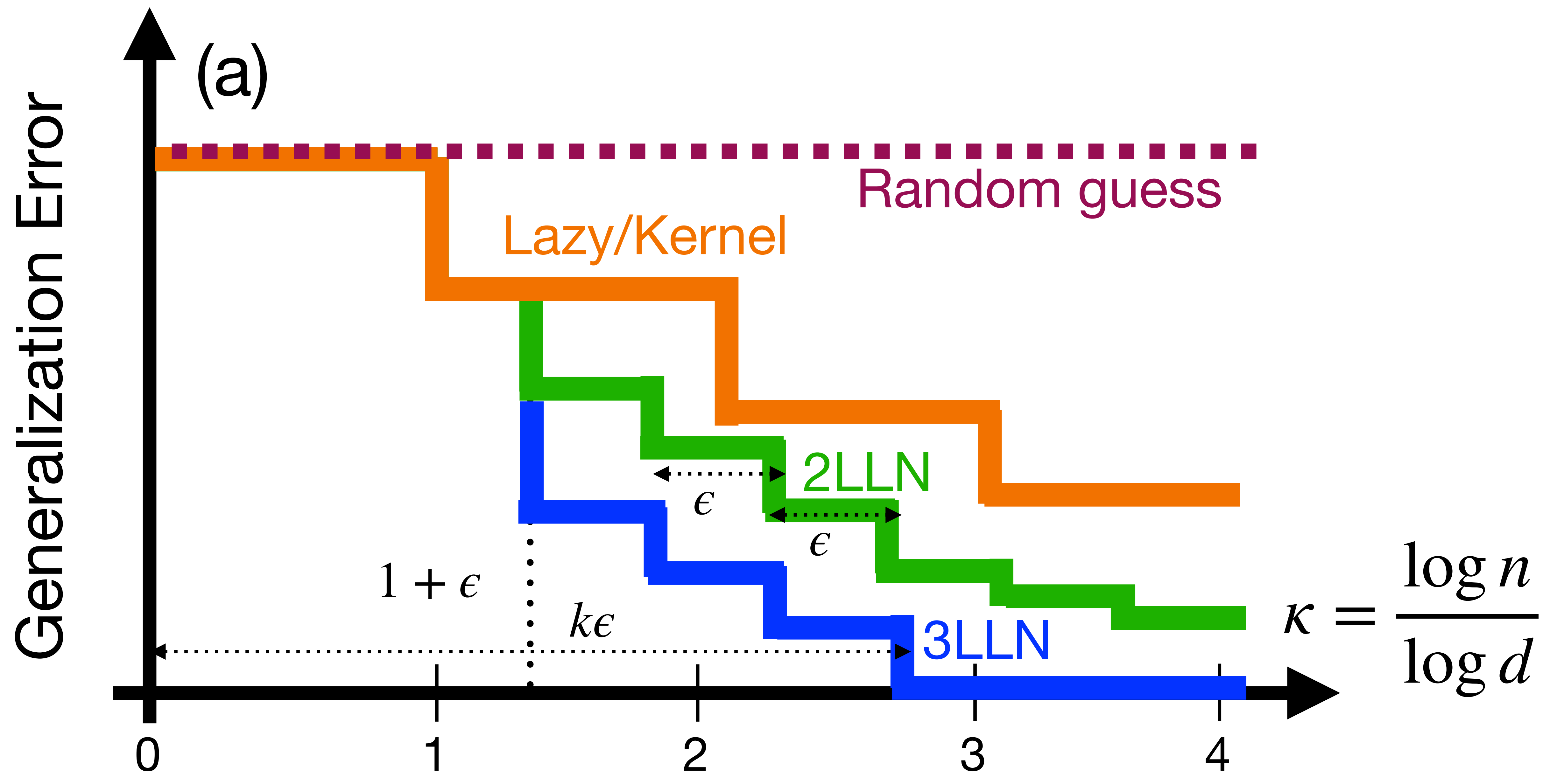
Two levels of dimension reduction

$$f^\star(\mathbf{x}) = \mathbf{g}^\star\left(\frac{\mathbf{a}^{\star\top}\mathbf{P_k}\left(\mathbf{W}^\star\mathbf{x}\right)}{\sqrt{\mathbf{d}^\varepsilon}}\right), \; \mathbf{x} \in \mathbb{R}^{\mathbf{d}} \quad \Longrightarrow \quad f^\star(\mathbf{x}) = \mathbf{g}^\star\left(\frac{\mathbf{a}^{\star\top}\mathbf{P_k}\left(\mathbf{x}_\star\right)}{\sqrt{\mathbf{d}^\varepsilon}}\right), \; \mathbf{x}_\star \in \mathbb{R}^{\mathbf{d}_\epsilon}$$



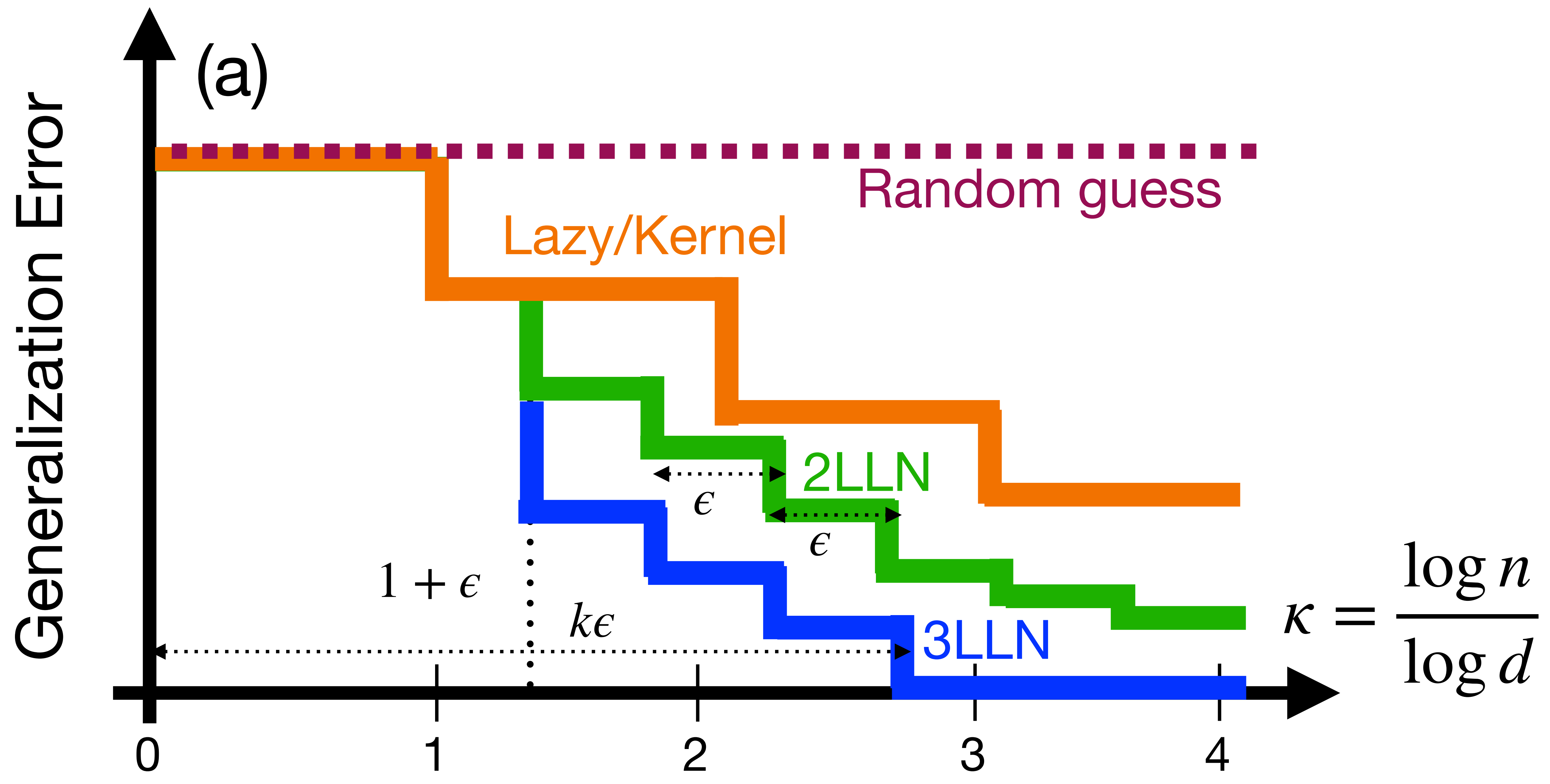
Two levels of dimension reduction

$$f^\star(\mathbf{x}) = \mathbf{g}^\star\left(\frac{\mathbf{a}^{\star\top}\mathbf{P_k}\left(\mathbf{W}^\star\mathbf{x}\right)}{\sqrt{\mathbf{d}^\varepsilon}}\right),\ \mathbf{x} \in \mathbb{R}^{\mathbf{d}} \Rightarrow f^\star(\mathbf{x}) = \mathbf{g}^\star\left(\frac{\mathbf{a}^{\star\top}\mathbf{P_k}\left(\mathbf{x}_\star\right)}{\sqrt{\mathbf{d}^\varepsilon}}\right),\ \mathbf{x}_\star \in \mathbb{R}^{\mathbf{d}_\epsilon} \Rightarrow$$



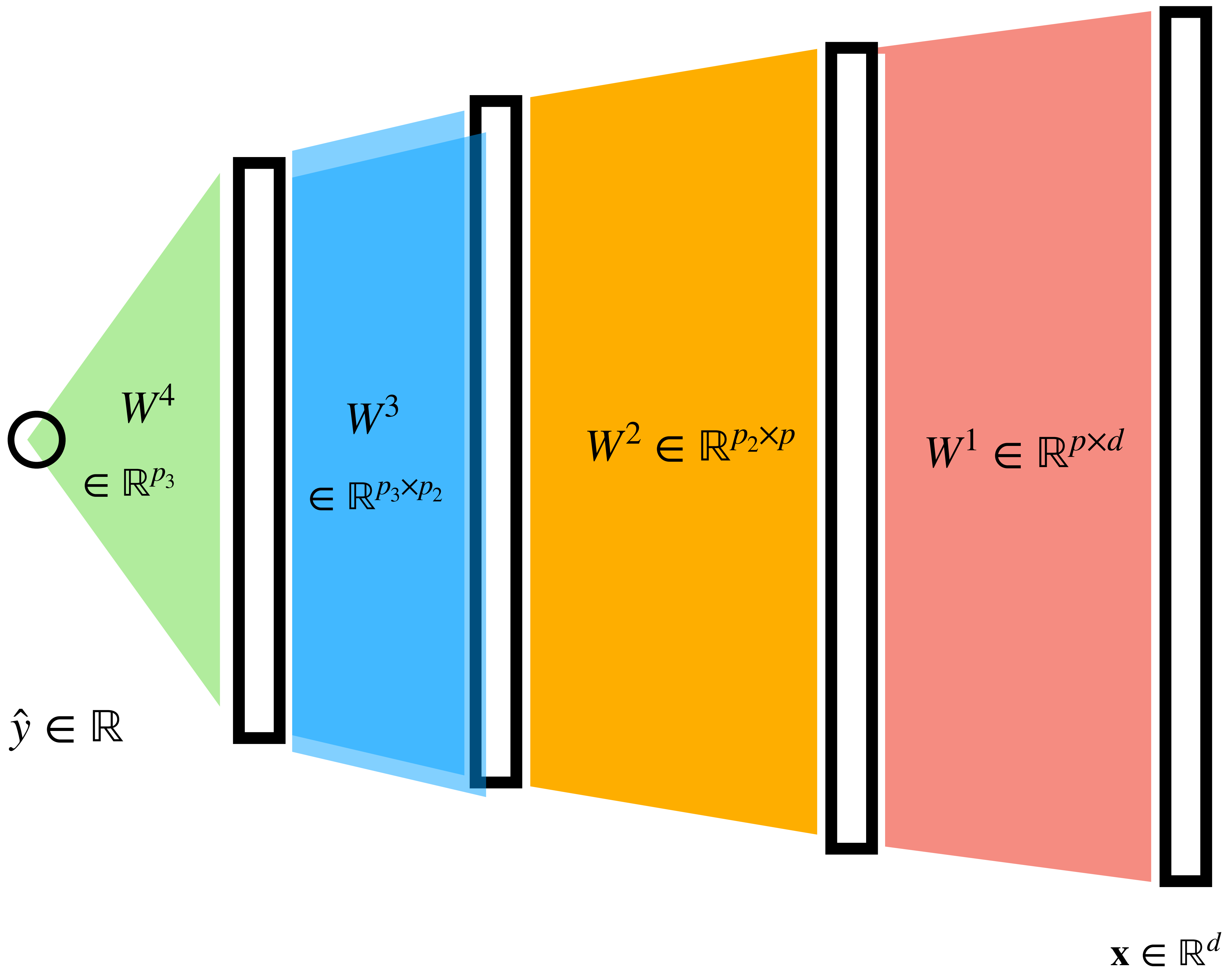
Two levels of dimension reduction

$$f^\star(\mathbf{x}) = \mathbf{g}^\star\left(\frac{\mathbf{a}^{\star\top}\mathbf{P}_\mathbf{k}\left(\mathbf{W}^\star\mathbf{x}\right)}{\sqrt{\mathbf{d}^\varepsilon}}\right), \ \mathbf{x} \in \mathbb{R}^\mathbf{d} \quad \Rightarrow \quad f^\star(\mathbf{x}) = \mathbf{g}^\star\left(\frac{\mathbf{a}^{\star\top}\mathbf{P}_\mathbf{k}\left(\mathbf{x}_\star\right)}{\sqrt{\mathbf{d}^\varepsilon}}\right), \ \mathbf{x}_\star \in \mathbb{R}^{\mathbf{d}_\epsilon} \quad \Rightarrow \quad f^\star(\mathbf{x}) = g^\star\left(h^\star\right)$$

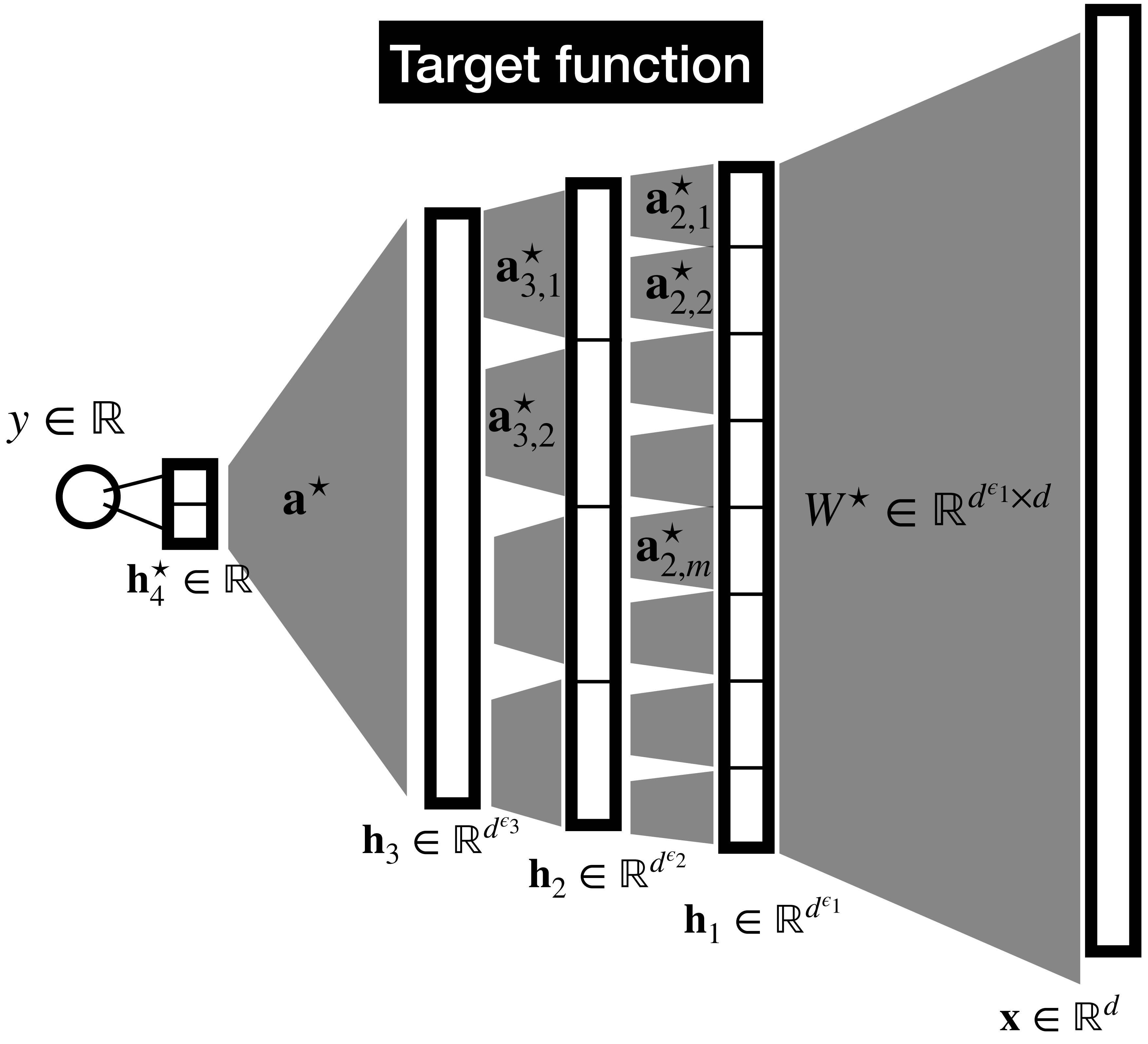


Deep version

Neural network

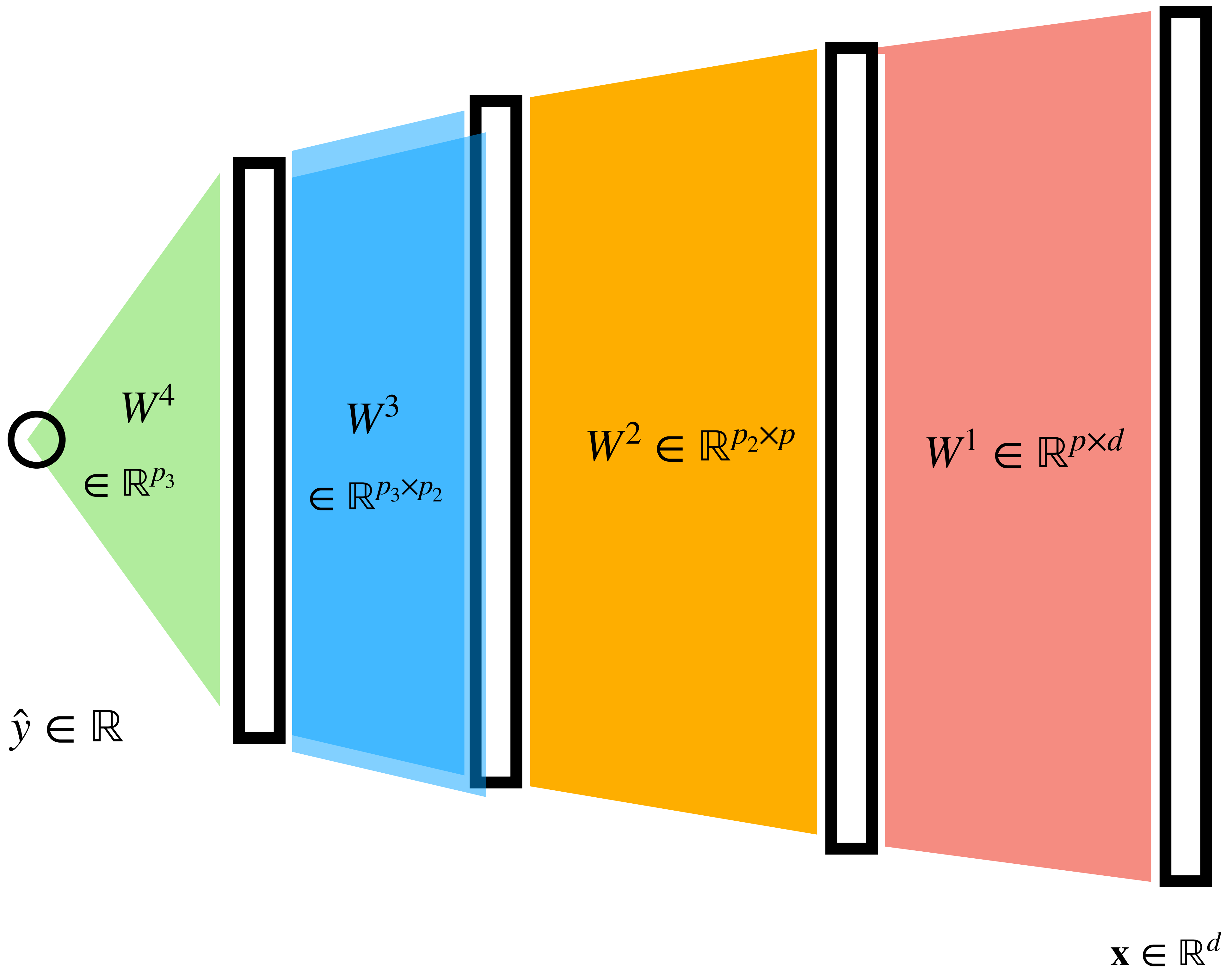


Target function

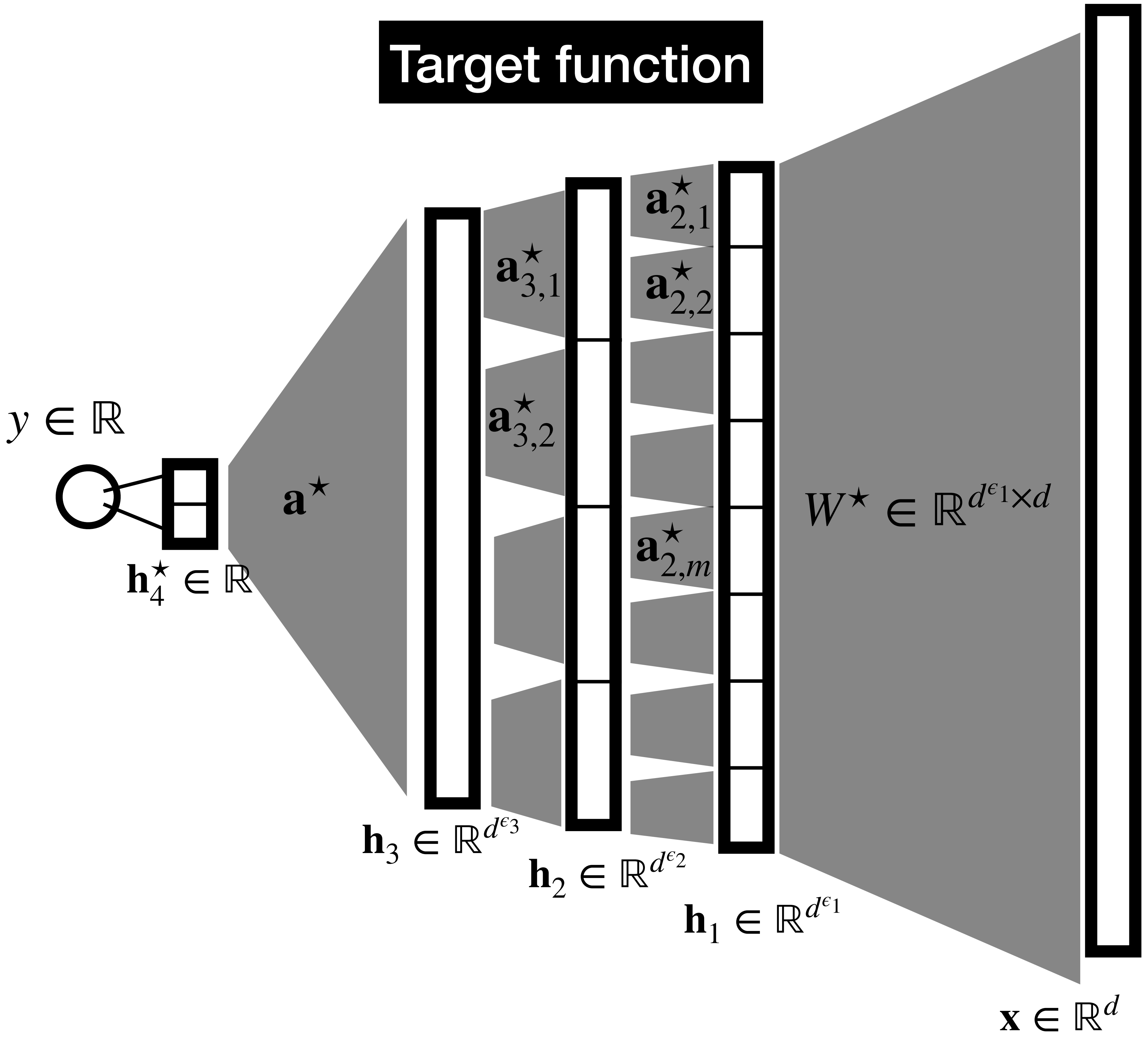


Deep version

Neural network



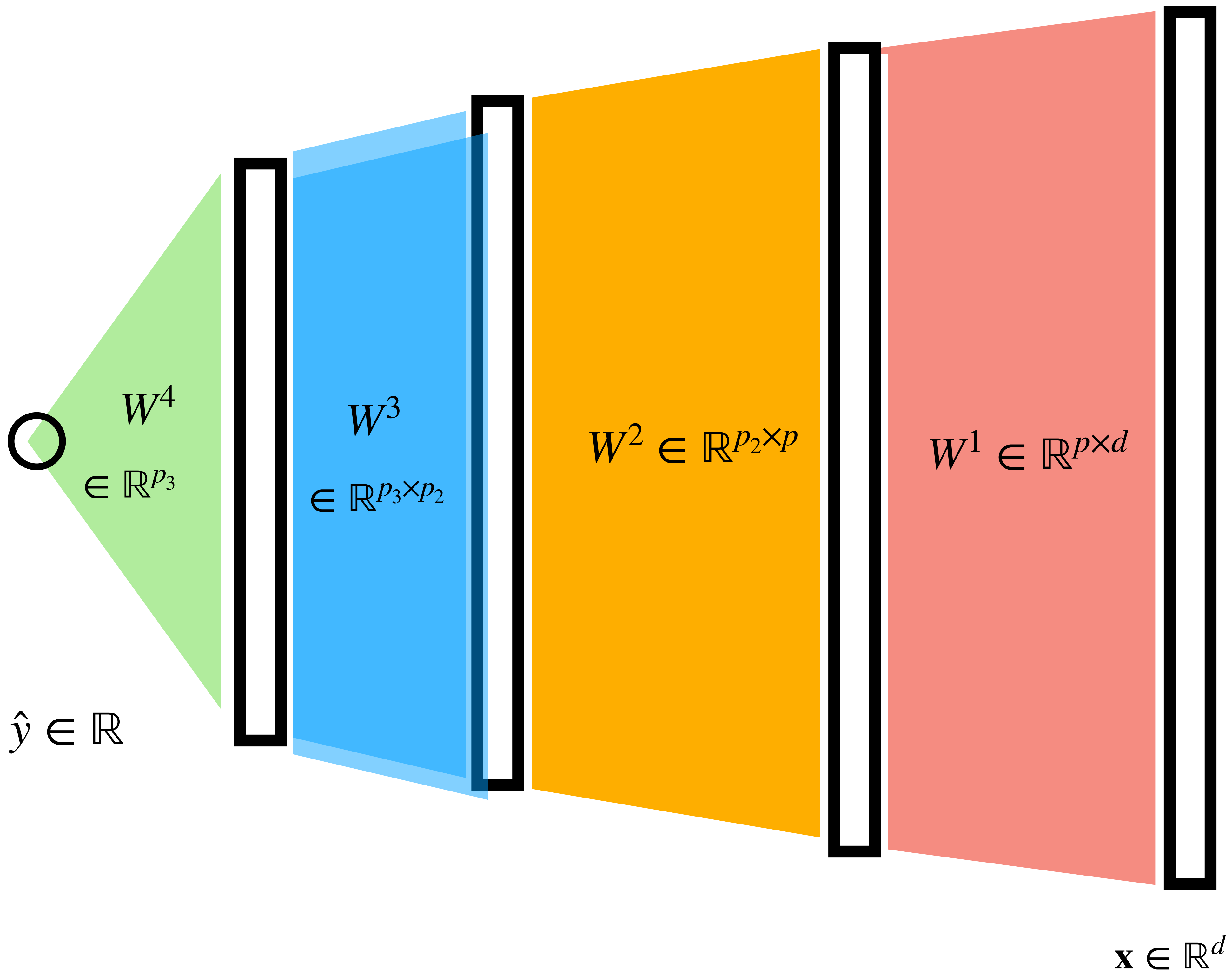
Target function



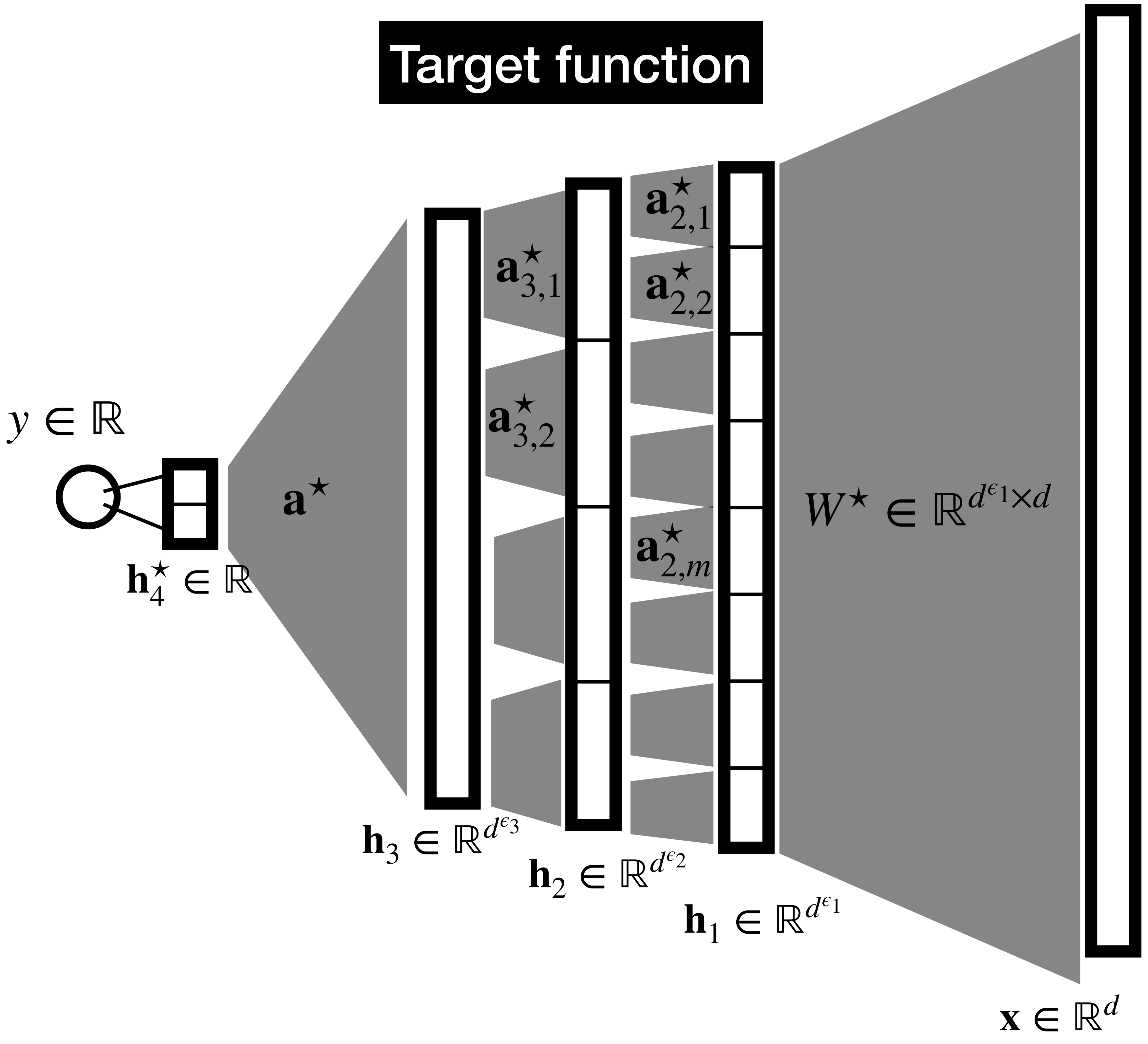
Tree-structure maintains independence of features

Deep version

Neural network



Target function

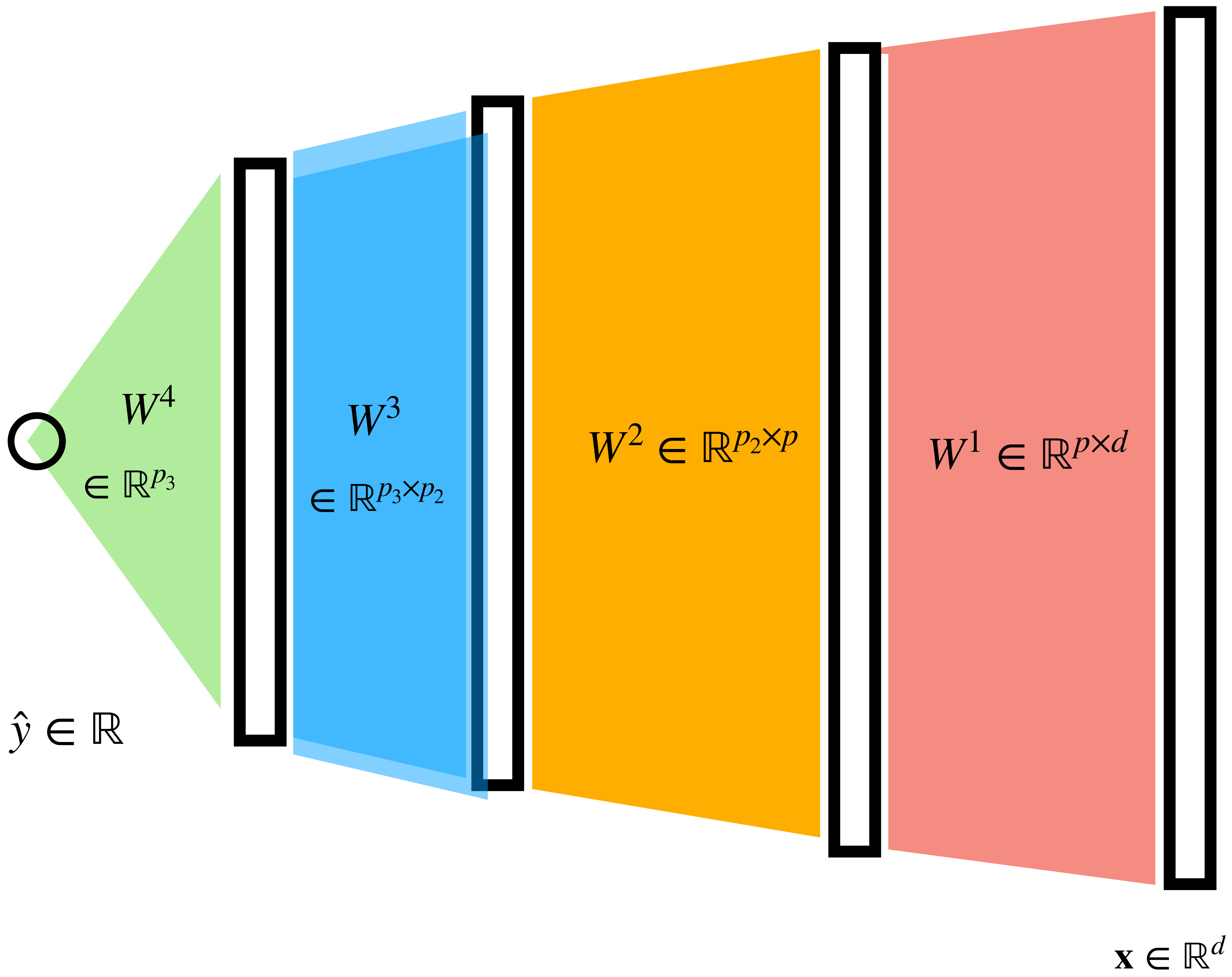


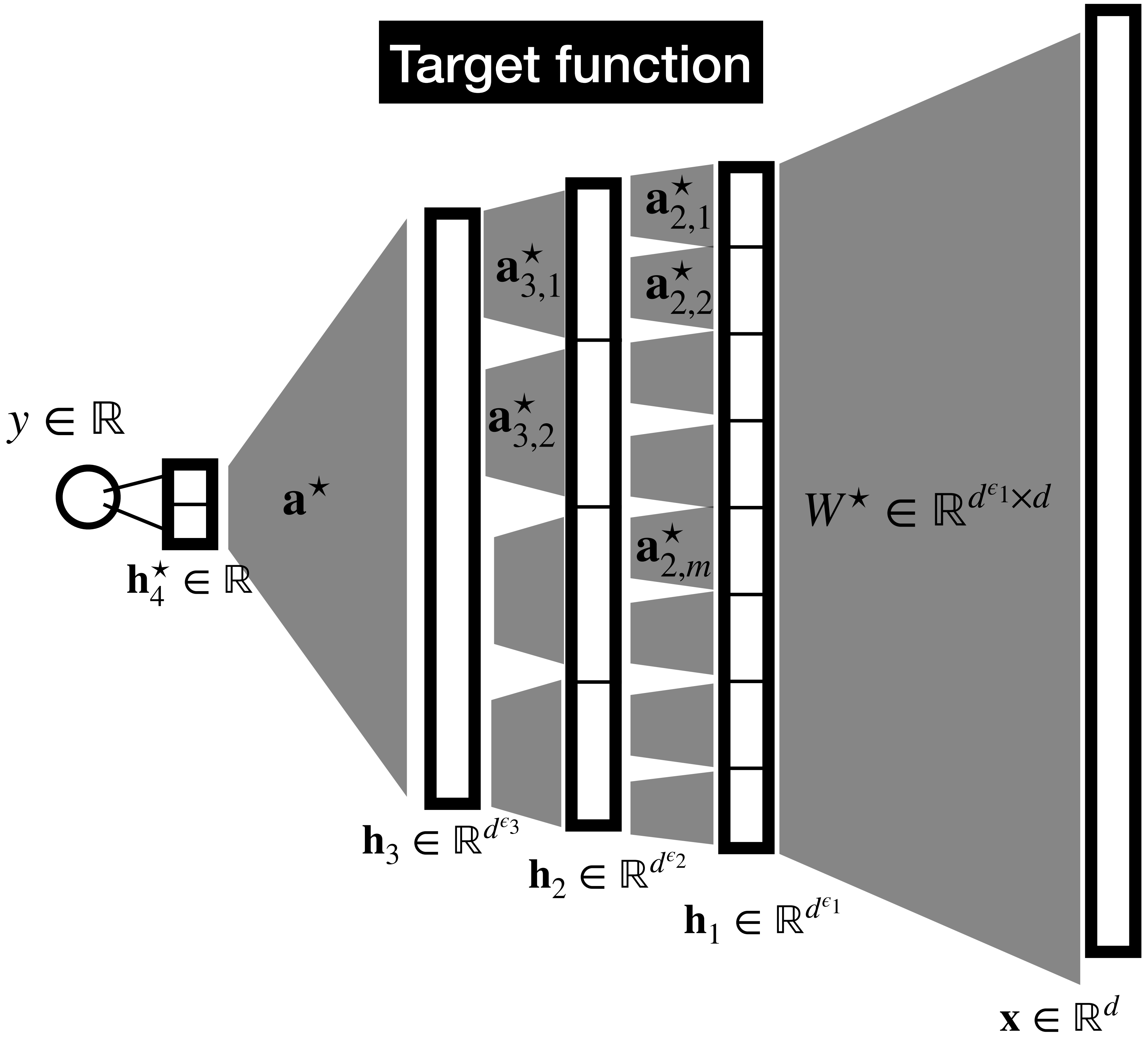
Iterative dimensionality reduction

$$d^{\epsilon_1} \to d^{\epsilon_2} \to d^{\epsilon_3}, \cdots, \to 1$$

Tree-structure maintains independence of features

Deep version





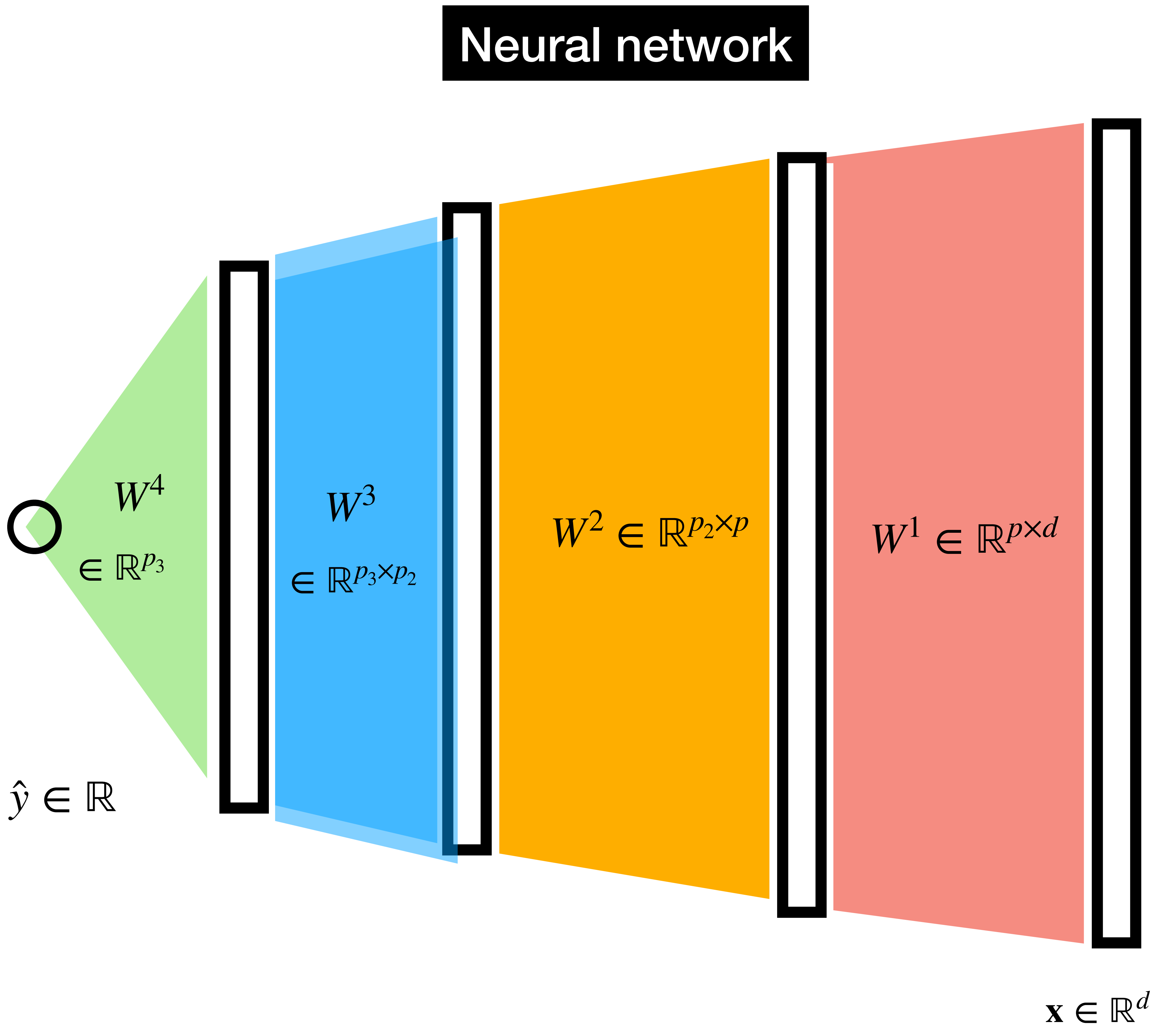
Iterative dimensionality reduction

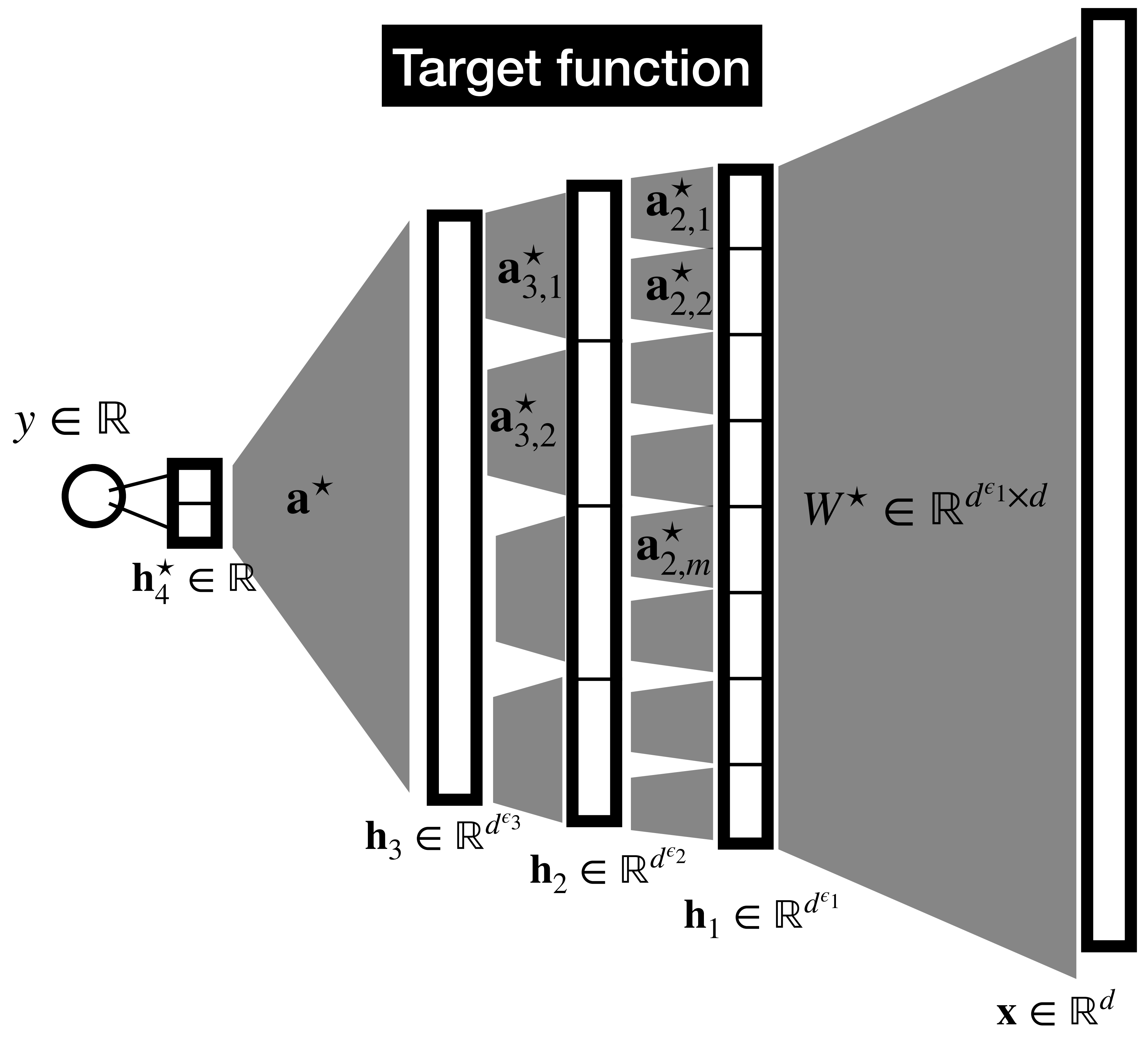
$$d^{\epsilon_1} \to d^{\epsilon_2} \to d^{\epsilon_3}, \cdots, \to 1$$

Tree-structure maintains independence of features

L-layer network

Deep version





Iterative dimensionality reduction

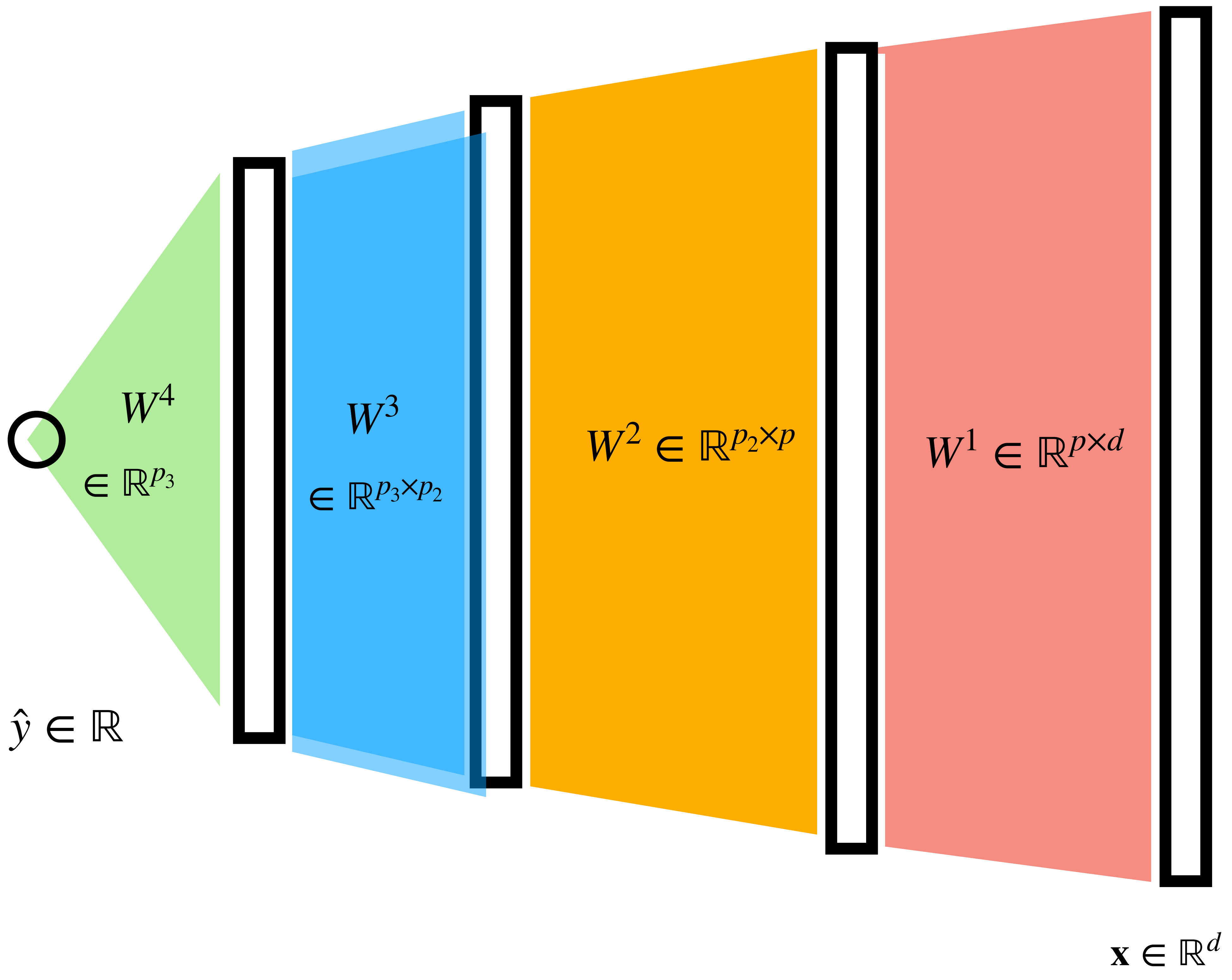
$$d^{\epsilon_1} \to d^{\epsilon_2} \to d^{\epsilon_3}, \cdots, \to 1$$

Tree-structure maintains independence of features

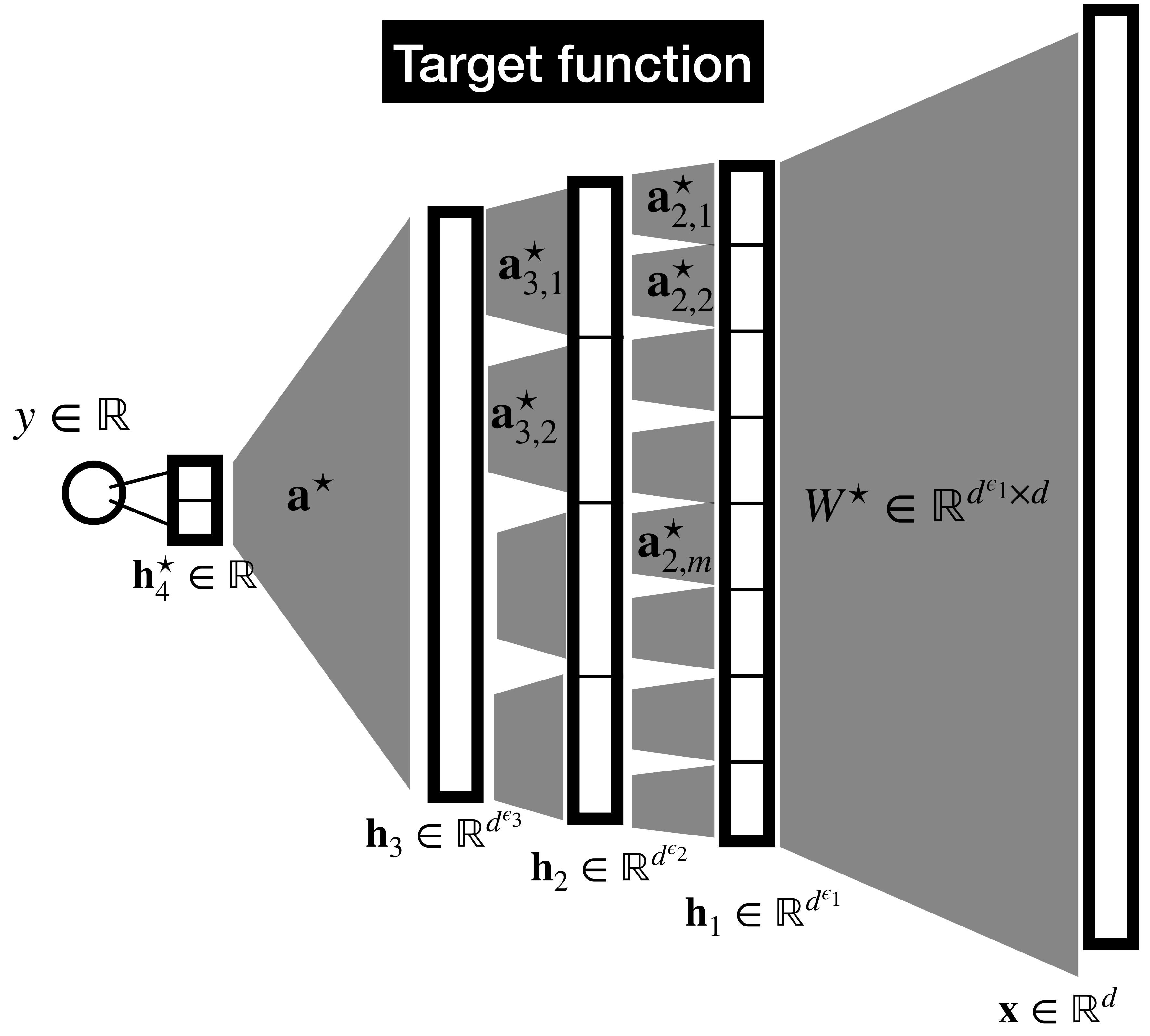
L-layer network →

Deep version

Neural network



Target function



Iterative dimensionality reduction

$$d^{\epsilon_1} \to d^{\epsilon_2} \to d^{\epsilon_3}, \cdots, \to 1$$

Tree-structure maintains independence of features

L-layer network → **L-1 levels of dimension reduction**

Conditions for learnability

Zeiler & Fergus, ***Visualizing and Understanding Convolutional Networks*****, ECCV 2014**

Conditions for learnability

Information exponent

$$\mathrm{IE}(\ell) = \inf\{k : \|\mathbb{E}[(\mathbf{x})^{\otimes k} f^{\star}(\mathbf{x})]\|_F \neq 0\}$$

Zeiler & Fergus, ***Visualizing and Understanding Convolutional Networks*****, ECCV 2014**

Conditions for learnability

Information exponent →

$$\mathrm{IE}(\ell) = \inf\{k : \|\mathbb{E}[(\mathbf{x})^{\otimes k} f^{\star}(\mathbf{x})]\|_F \neq 0\}$$

Zeiler & Fergus, ***Visualizing and Understanding Convolutional Networks*****, ECCV 2014**

Conditions for learnability

Information exponent → **Compositional Information exponent**

$$\mathrm{IE}(\ell) = \inf\{k : \|\mathbb{E}[(\mathbf{x})^{\otimes k} f^\star(\mathbf{x})]\|_F \neq 0\}$$

$$\mathrm{CIE}(\ell) = \inf\{k : \|\mathbb{E}[(h_\ell(\mathbf{x}))^{\otimes k} f^\star(\mathbf{x})]\|_F = \Theta(1)\}$$

Zeiler & Fergus, ***Visualizing and Understanding Convolutional Networks*****, ECCV 2014**

Conditions for learnability

Information exponent → **Compositional Information exponent**

$$\mathrm{IE}(\ell) = \inf\{k : \|\mathbb{E}[(\mathbf{x})^{\otimes k} f^\star(\mathbf{x})]\|_F \neq 0\}$$

$$\mathrm{CIE}(\ell) = \inf\{k : \|\mathbb{E}[(h_\ell(\mathbf{x}))^{\otimes k} f^\star(\mathbf{x})]\|_F = \Theta(1)\}$$

Requires non-trivial low-degree correlations between intermediate features and labels

Zeiler & Fergus, *Visualizing and Understanding Convolutional Networks*, ECCV 2014

Conditions for learnability

Information exponent → **Compositional Information exponent**

$$\mathrm{IE}(\ell) = \inf\{k : \|\mathbb{E}[(\mathbf{x})^{\otimes k} f^\star(\mathbf{x})]\|_F \neq 0\}$$

$$\mathrm{CIE}(\ell) = \inf\{k : \|\mathbb{E}[(h_\ell(\mathbf{x}))^{\otimes k} f^\star(\mathbf{x})]\|_F = \Theta(1)\}$$

Requires non-trivial low-degree correlations between intermediate features and labels

Matches behavior of real data

	Cal-101 (30/class)	Cal-256 (60/class)
SVM (1)	44.8 ± 0.7	24.6 ± 0.4
SVM (2)	66.2 ± 0.5	39.6 ± 0.3
SVM (3)	72.3 ± 0.4	46.0 ± 0.3
SVM (4)	76.6 ± 0.4	51.3 ± 0.1
SVM (5)	$\mathbf{86.2 \pm 0.8}$	65.6 ± 0.3
SVM (7)	$\mathbf{85.5 \pm 0.4}$	$\mathbf{71.7 \pm 0.2}$
Softmax (5)	82.9 ± 0.4	65.7 ± 0.5
Softmax (7)	$\mathbf{85.4 \pm 0.4}$	$\mathbf{72.6 \pm 0.1}$

Table 7. Analysis of the discriminative information contained in each layer of feature maps within our ImageNet-pretrained convnet. We train either a linear SVM or softmax on features from different layers (as indicated in brackets) from the convnet. Higher layers generally produce more discriminative features.

Zeiler & Fergus, *Visualizing and Understanding Convolutional Networks*, ECCV 2014

Full set of conditions

Essential assumptions	Technical assumptions

Full set of conditions

Essential assumptions

- $g^\star$: information exponent 1.
- P_k: information exponent/leap ≤ 2

$$\mathrm{CIE}(1) = IE(g^\star) \times IE(P_k)$$

- expressive σ (non-zero Hermites) and regular.

Technical assumptions

Full set of conditions

Essential assumptions

- $g^\star$: information exponent 1.
- P_k: information exponent/leap ≤ 2

$$\mathrm{CIE}(1) = IE(g^\star) \times IE(P_k)$$

- expressive σ (non-zero Hermites) and regular.

Technical assumptions

- $a_i^\star = 1 \ \forall i$ (symmetric targets)
- Correlation loss
- Re-initialization of layers
- expressive (non-zero Hermites) and regular.
- P_k: information exponent $\neq 1$(to avoid spikes)
- $\mathbb{E}[\sigma(\sigma(z))\mathrm{He}_2(z)]\mathbb{E}[P_k(z)\mathrm{He}_2(z)] > 0,$ $\mathbb{E}[\sigma(\sigma(z))z] = 0,$
- $\mathbb{E}_{z\sim\mathcal{N}(0,1)}[g^*(z)\mathrm{He}_j(z) = 0, \ 1 < j \leq k]$

Main theorem

Main theorem

$$\hat{f}(\mathbf{x}) = \mathbf{w_3}^\top \sigma\left(W_2\sigma(W_1\mathbf{x} + b_1) + b_2\right)$$

$$W_1 \in \mathbb{R}^{p_1 \times d}, W_2 \in \mathbb{R}^{p_2 \times p_1}, \mathbf{w_3} \in \mathbb{R}^{\mathbf{p_3}}$$

Main theorem

$$\hat{f}(\mathbf{x}) = \mathbf{w_3}^\top \sigma\left(W_2 \sigma(W_1 \mathbf{x} + b_1) + b_2\right)$$

$$W_1 \in \mathbb{R}^{p_1 \times d}, W_2 \in \mathbb{R}^{p_2 \times p_1}, \mathbf{w_3} \in \mathbb{R}^{\mathbf{p_3}}$$

Let $\delta > 0$ be arbitrarily small

Main theorem

$$\hat{f}(\mathbf{x}) = \mathbf{w_3}^\top \sigma\left(W_2 \sigma(W_1 \mathbf{x} + b_1) + b_2\right)$$

$$W_1 \in \mathbb{R}^{p_1 \times d}, W_2 \in \mathbb{R}^{p_2 \times p_1}, \mathbf{w_3} \in \mathbb{R}^{\mathbf{p_3}}$$

Let $\delta > 0$ be arbitrarily small

- First layer: $\mathcal{O}(\log d)$ steps of spherical SGD on W_1 with correlation loss, batch-size $\mathcal{O}(d^{1+\epsilon+\delta})$, vanishing step-size:

Main theorem

$$\hat{f}(\mathbf{x}) = \mathbf{w_3}^\top \sigma\left(W_2\sigma(W_1\mathbf{x} + b_1) + b_2\right)$$

$$W_1 \in \mathbb{R}^{p_1 \times d}, W_2 \in \mathbb{R}^{p_2 \times p_1}, \mathbf{w_3} \in \mathbb{R}^{\mathbf{p_3}}$$

Let $\delta > 0$ be arbitrarily small

- First layer: $\mathcal{O}(\log d)$ steps of spherical SGD on W_1 with correlation loss, batch-size $\mathcal{O}(d^{1+\epsilon+\delta})$, vanishing step-size:

$$W_1 = Z(W^\star) + o_d(1), z_i \sim U(S_1)$$

Main theorem

$$\hat{f}(\mathbf{x}) = \mathbf{w_3}^\top \sigma\left(W_2 \sigma(W_1 \mathbf{x} + b_1) + b_2\right)$$

$$W_1 \in \mathbb{R}^{p_1 \times d}, W_2 \in \mathbb{R}^{p_2 \times p_1}, \mathbf{w_3} \in \mathbb{R}^{\mathbf{p_3}}$$

Let $\delta > 0$ be arbitrarily small

- First layer: $\mathcal{O}(\log d)$ steps of spherical SGD on W_1 with correlation loss, batch-size $\mathcal{O}(d^{1+\epsilon+\delta})$, vanishing step-size:

$$W_1 = Z(W^\star) + o_d(1), z_i \sim U(S_1)$$

- Second layer: upon reinitializing $W_2 = \mathbf{0}_{d \times d}$, one step of pre-conditioned step with batch-size $\mathcal{O}(d^{k\epsilon+\delta}),\ p_1 = \mathcal{O}(d^{k\epsilon+\delta})$:

Main theorem

$$\hat{f}(\mathbf{x}) = \mathbf{w_3}^\top \sigma\left(W_2\sigma(W_1\mathbf{x} + b_1) + b_2\right)$$

$$W_1 \in \mathbb{R}^{p_1 \times d}, W_2 \in \mathbb{R}^{p_2 \times p_1}, \mathbf{w_3} \in \mathbb{R}^{\mathbf{p_3}}$$

Let $\delta > 0$ be arbitrarily small

- First layer: $\mathcal{O}(\log d)$ steps of spherical SGD on W_1 with correlation loss, batch-size $\mathcal{O}(d^{1+\epsilon+\delta})$, vanishing step-size:

$$W_1 = Z(W^\star) + o_d(1), z_i \sim U(S_1)$$

- Second layer: upon reinitializing $W_2 = \mathbf{0}_{d\times d}$, one step of pre-conditioned step with batch-size $\mathcal{O}(d^{k\epsilon+\delta})$, $p_1 = \mathcal{O}(d^{k\epsilon+\delta})$:

$$\mathbf{h_i}(\mathbf{x}) = \mathbf{W_2}\sigma(\mathbf{W_1}\mathbf{x}) = \mathbf{c}\mathbf{w}_\mathbf{3}^\mathbf{i}\mathbf{h}^\star(\mathbf{x}) + \mathbf{o_d}(\mathbf{1})$$

Main theorem

$$\hat{f}(\mathbf{x}) = \mathbf{w_3}^\top \sigma\left(W_2\sigma(W_1\mathbf{x} + b_1) + b_2\right)$$

$$W_1 \in \mathbb{R}^{p_1 \times d}, W_2 \in \mathbb{R}^{p_2 \times p_1}, \mathbf{w_3} \in \mathbb{R}^{\mathbf{p_3}}$$

Let $\delta > 0$ be arbitrarily small

- First layer: $\mathcal{O}(\log d)$ steps of spherical SGD on W_1 with correlation loss, batch-size $\mathcal{O}(d^{1+\epsilon+\delta})$, vanishing step-size:

$$W_1 = Z(W^\star) + o_d(1), z_i \sim U(S_1)$$

- Second layer: upon reinitializing $W_2 = \mathbf{0}_{d\times d}$, one step of pre-conditioned step with batch-size $\mathcal{O}(d^{k\epsilon+\delta}),\ p_1 = \mathcal{O}(d^{k\epsilon+\delta})$:

$$\mathbf{h_i}(\mathbf{x}) = \mathbf{W_2}\sigma(\mathbf{W_1}\mathbf{x}) = \mathbf{c}\mathbf{w_3^i}\mathbf{h}^\star(\mathbf{x}) + \mathbf{o_d}(\mathbf{1})$$

- Third layer: Ridge regression on $\mathbf{w_3}$ with samples $\mathcal{O}(d^\delta)$, $p_2 = \mathcal{O}(d^\delta)$:

Main theorem

$$\hat{f}(\mathbf{x}) = \mathbf{w_3}^\top \sigma\left(W_2\sigma(W_1\mathbf{x} + b_1) + b_2\right)$$

$$W_1 \in \mathbb{R}^{p_1 \times d}, W_2 \in \mathbb{R}^{p_2 \times p_1}, \mathbf{w_3} \in \mathbb{R}^{\mathbf{p_3}}$$

Let $\delta > 0$ be arbitrarily small

- First layer: $\mathcal{O}(\log d)$ steps of spherical SGD on W_1 with correlation loss, batch-size $\mathcal{O}(d^{1+\epsilon+\delta})$, vanishing step-size:

$$W_1 = Z(W^\star) + o_d(1), z_i \sim U(S_1)$$

- Second layer: upon reinitializing $W_2 = \mathbf{0}_{d\times d}$, one step of pre-conditioned step with batch-size $\mathcal{O}(d^{k\epsilon+\delta})$, $p_1 = \mathcal{O}(d^{k\epsilon+\delta})$:

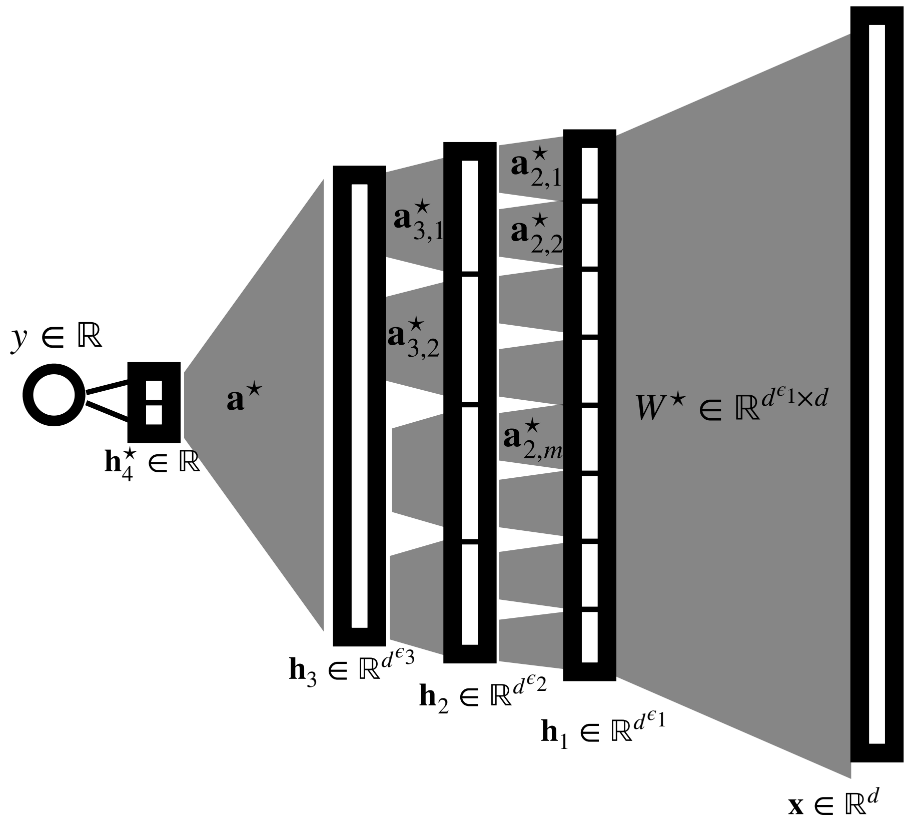
$$\mathbf{h_i}(\mathbf{x}) = \mathbf{W_2}\sigma(\mathbf{W_1}\mathbf{x}) = \mathbf{c}\mathbf{w_3^i}\mathbf{h}^\star(\mathbf{x}) + \mathbf{o_d}(\mathbf{1})$$

- Third layer: Ridge regression on $\mathbf{w_3}$ with samples $\mathcal{O}(d^\delta)$, $p_2 = \mathcal{O}(d^\delta)$:

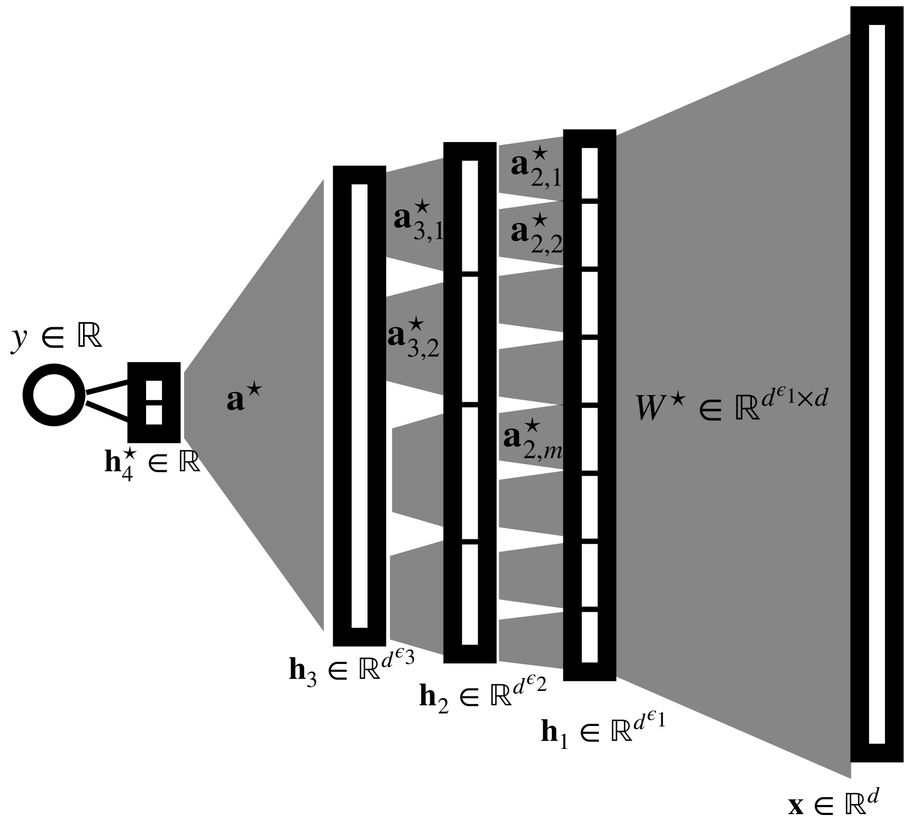
$$\hat{f}(\mathbf{x}) = \mathbf{f}^\star(\mathbf{x}) + \mathbf{o_d}(\mathbf{1})$$

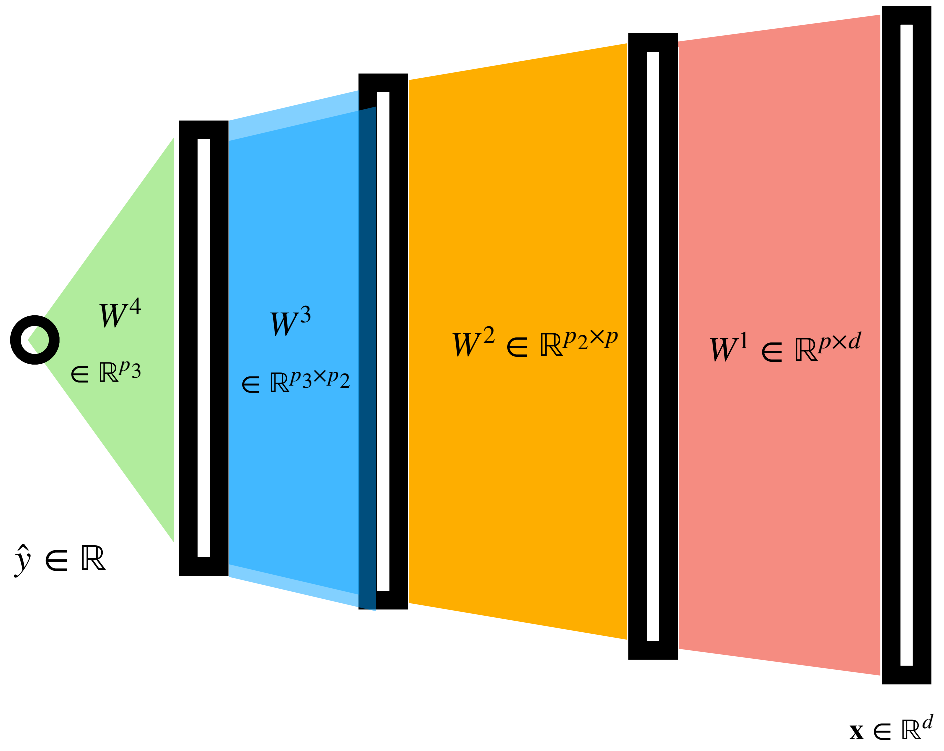
General Depth

General Depth

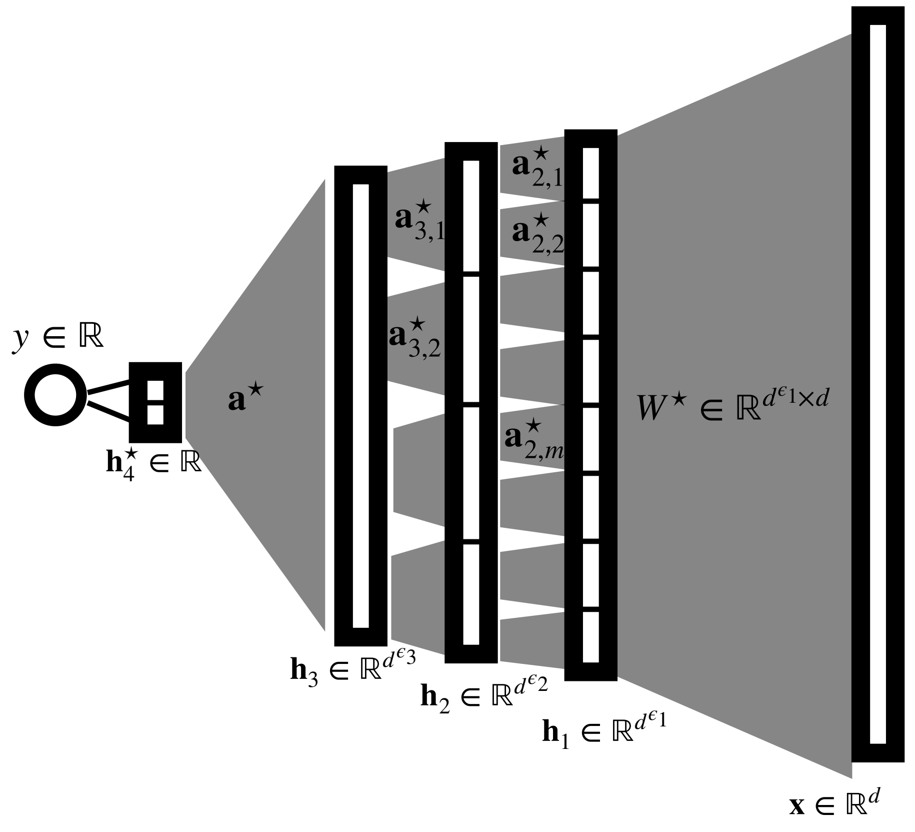


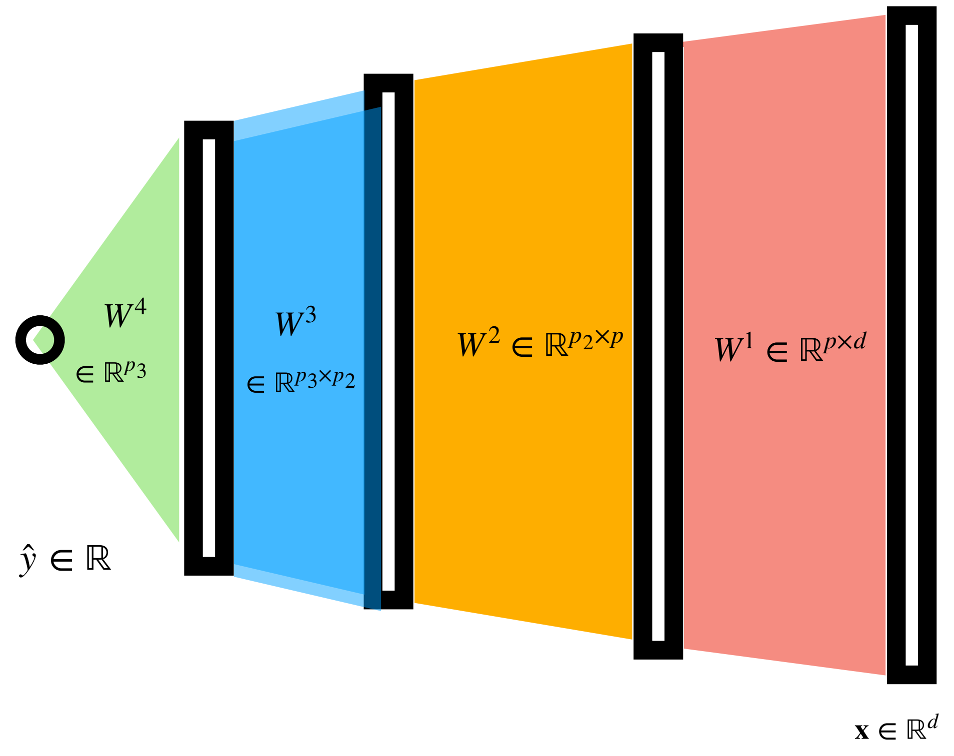
General Depth





General Depth

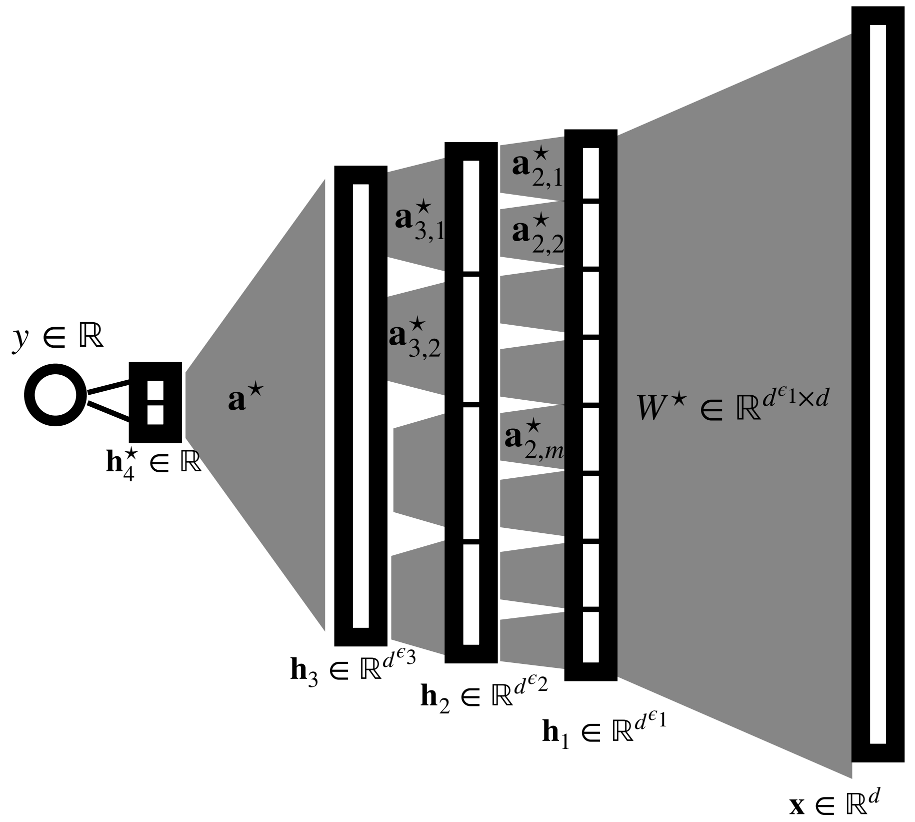


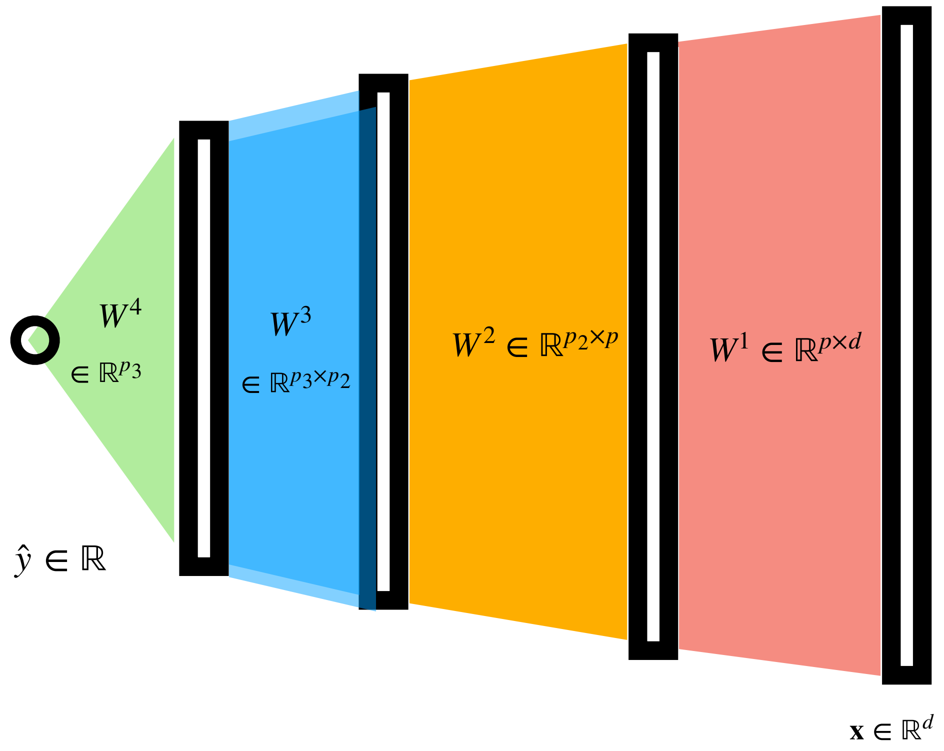


- Conditional on perfect spherical recovery for W_{L-2}, the same picture holds for the last two layers.

- Key idea: Features are independent by tree-structure, asymptotically Gaussian, and maintain nice tails (hypercontractivity).

General Depth





- Conditional on perfect spherical recovery for W_{L-2}, the same picture holds for the last two layers.

- Key idea: Features are independent by tree-structure, asymptotically Gaussian, and maintain nice tails (hypercontractivity).

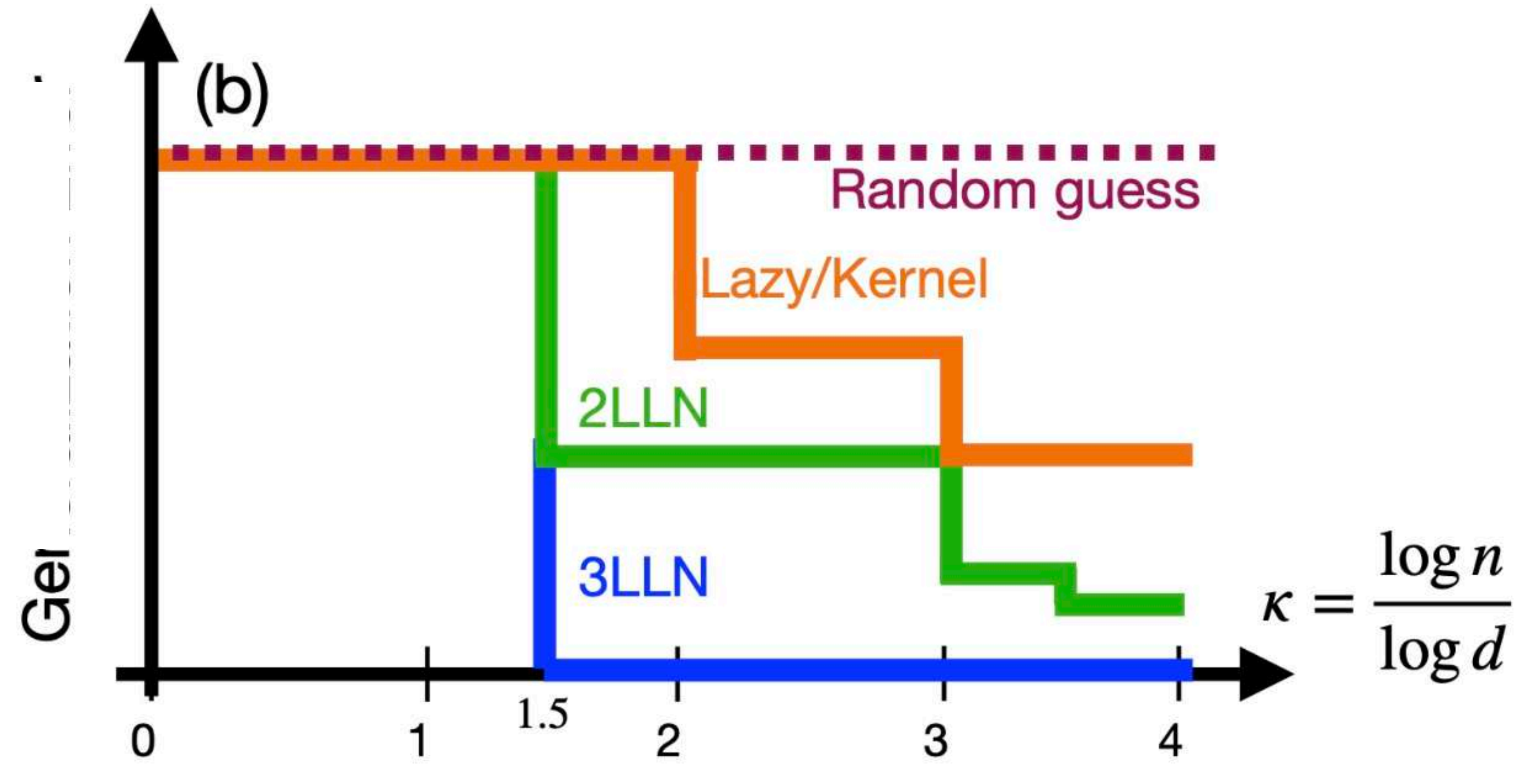
Theorem 2. *For $L \in \mathbb{N}$, let $f^\star(\mathbf{x})$ denote a target as in Eq. (6) with $r = 1$, and let δ', δ be arbitrary reals satisfying $0 < \delta < \delta' < 1$. Consider a model of the form $\hat{f}_\theta(\mathbf{x}) = \mathbf{w}_L^\top \sigma(W_{L-1}\sigma(W h^\star_{L-1}(\mathbf{x})))$ with $W \in \mathbb{R}^{p_{L-2}\times d^{\varepsilon_{L-2}}}$ having $p_{L-2} = \Theta(d^{k\varepsilon_{\ell-2}+\delta'})$ rows independently sampled as $\mathbf{w}_i \sim U(\mathbb{S}_{d^{\varepsilon_{L-2}}}(1))$. Under Ass. 1-3, after a single step of pre-conditioned SGD on W_{L-1} with batch-size $\Theta(d^{k\varepsilon_{\ell-2}+\delta})$, step-size $\Theta(\sqrt{p_{L-1}})$, the pre-activations $h_{L-1}(\mathbf{x}) := W_{L-1}\sigma(W h^\star_{L-1}(\mathbf{x}))$ satisfy, for a constant $c > 0$:*

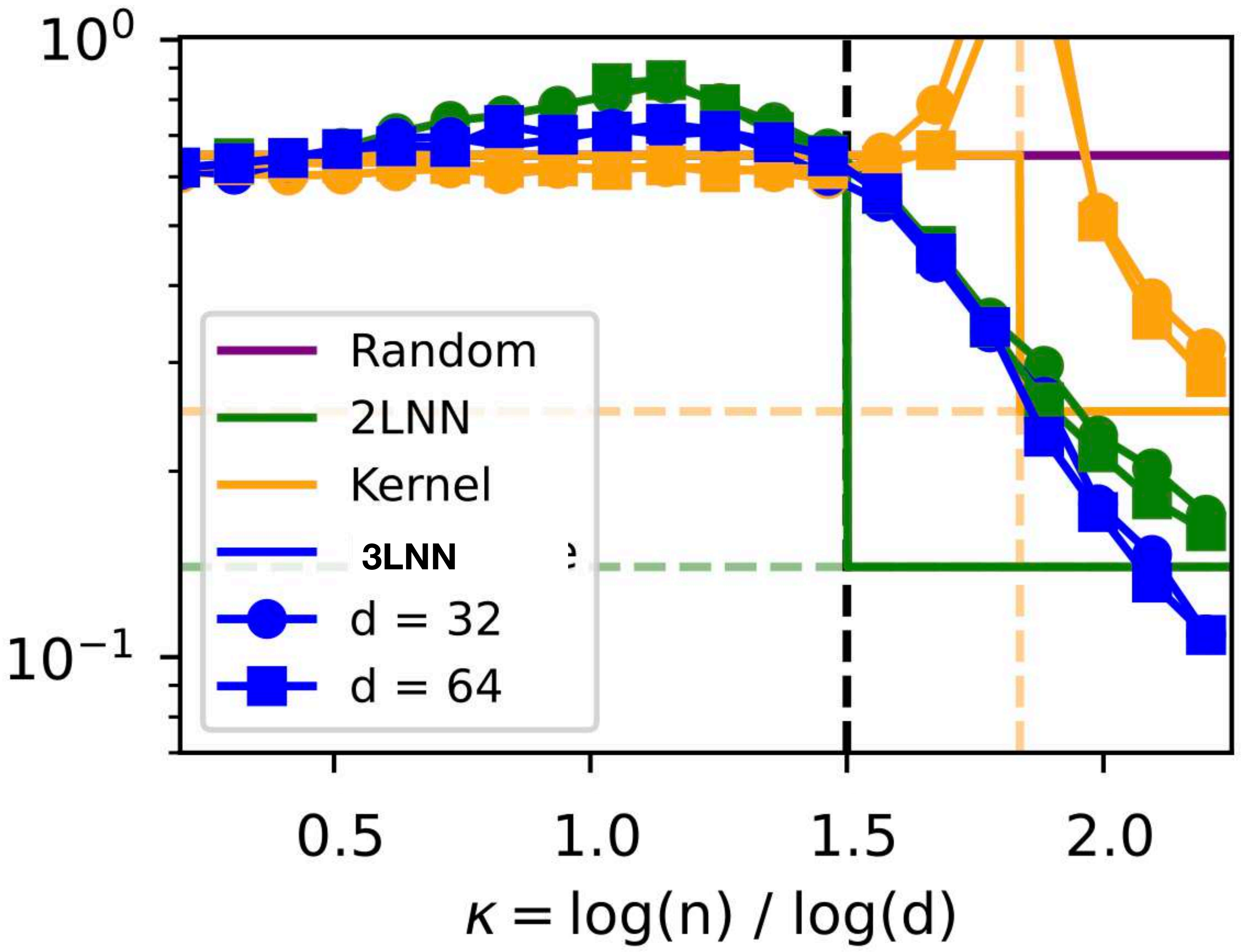
$$h_{L-1}(\mathbf{x}) = c\mathbf{w}_L h^\star_L(\mathbf{x}) + o_d(1), \tag{22}$$

Example

$$f^\star(\mathbf{x}) = \tanh\left(\frac{\mathbf{a}^{\star\top} P_3(W^\star \mathbf{x})}{\sqrt{d^{\varepsilon_1 = 1/2}}}\right)$$

$$P_3(x) = \mathrm{He}_2(x) + \mathrm{He}_3(x)$$





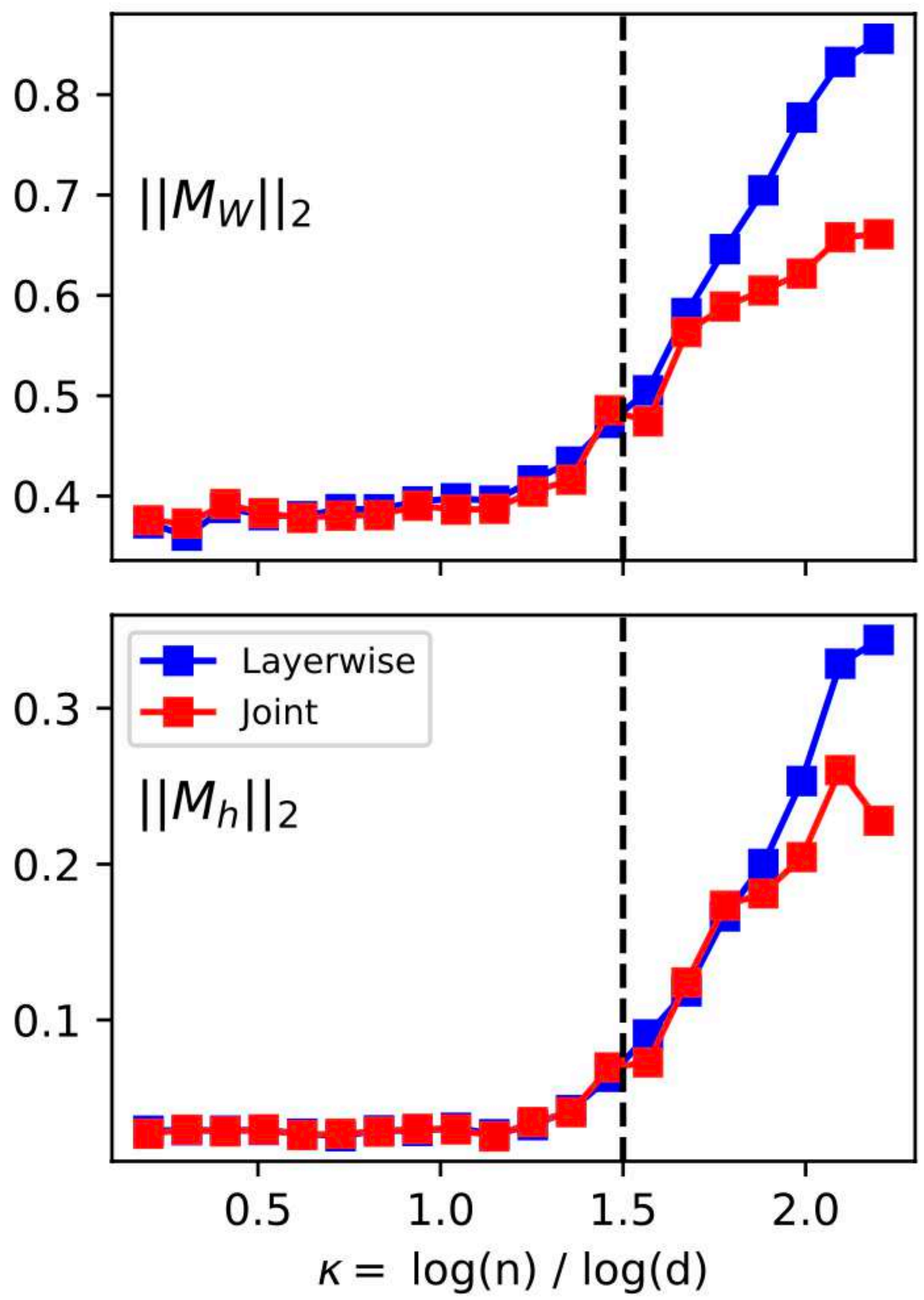
Overlaps in parameter and function space

$$f^\star(\mathbf{x}) = \tanh\left(\frac{\mathbf{a}^{\star\top} P_3\left(W^\star \mathbf{x}\right)}{\sqrt{d^{\varepsilon_1 = 1/2}}}\right)$$

$$P_3(x) \;=\; \mathrm{He}_2(x) + \mathrm{He}_3(x)$$

$$M_W = \frac{W_1 W^\star}{\|W_1\|_2},$$

$$M_h = \frac{\mathbb{E}[\mathbf{h}(\mathbf{z})\mathbf{h}^\star(\mathbf{z})]}{\sqrt{\mathbb{E}[\mathbf{h}(\mathbf{z})^2]}}.$$



V

Key ideas and proof sketches

Recursive Expansion of the Target

Recursive Expansion of the Target

$$f^\star(\mathbf{x}) = g^\star\left(\frac{\mathbf{a}^{\star\top} P_k\left(W^\star \mathbf{x}\right)}{\sqrt{d^\varepsilon}}\right), \mathbf{x} \in \mathbb{R}^d$$

Recursive Expansion of the Target

$$f^\star(\mathbf{x}) = g^\star\left(\frac{\mathbf{a}^{\star\top} P_k\left(W^\star \mathbf{x}\right)}{\sqrt{d^\varepsilon}}\right), \ \mathbf{x} \in \mathbb{R}^d \qquad h^\star = \mathbf{a}^\star \cdot \frac{P_k\left(\mathbf{x}_\star\right)}{\sqrt{d^\varepsilon}} \in \mathbb{R}$$

Recursive Expansion of the Target

$$f^\star(\mathbf{x}) = g^\star\left(\frac{\mathbf{a}^{\star\top} P_k\left(W^\star \mathbf{x}\right)}{\sqrt{d^\varepsilon}}\right), \ \mathbf{x} \in \mathbb{R}^d \qquad h^\star = \mathbf{a}^\star \cdot \frac{P_k\left(\mathbf{x}_\star\right)}{\sqrt{d^\varepsilon}} \in \mathbb{R}$$

$$g^\star(h_\star(\mathbf{x})) \approx \mu_1 \mathbf{h}^\star(\mathbf{x}) + \mu_2^\star \mathbf{He}_2(\mathbf{h}^\star(\mathbf{x})) + \dots$$

Recursive Expansion of the Target

$$f^{\star}(\mathbf{x}) = g^{\star}\left(\frac{\mathbf{a}^{\star\top} P_k\left(W^{\star}\mathbf{x}\right)}{\sqrt{d^{\varepsilon}}}\right), \mathbf{x} \in \mathbb{R}^d \qquad h^{\star} = \mathbf{a}^{\star} \cdot \frac{P_k\left(\mathbf{x}_{\star}\right)}{\sqrt{d^{\varepsilon}}} \in \mathbb{R}$$

$$g^{\star}(h_{\star}(\mathbf{x})) \approx \mu_1 \mathbf{h}^{\star}(\mathbf{x}) + \mu_2^{\star}\mathbf{He}_2(\mathbf{h}^{\star}(\mathbf{x})) + \dots$$

!!! $He_k(h^{\star}(\mathbf{x}))$ can contribute low-degree terms

Composition of Hermites

Composition of Hermites

$$h(\mathbf{x}) = \frac{\mathbf{1}}{\sqrt{\mathbf{d}^{\epsilon}}} \sum_{\mathbf{i}=\mathbf{1}}^{\mathbf{d}^{\epsilon}} \mathrm{He}_{\mathbf{k}}(\langle \mathbf{w}_{\mathbf{i}}^{\star}, \mathbf{x} \rangle)$$

Composition of Hermites

$$h(\mathbf{x}) = \frac{\mathbf{1}}{\sqrt{\mathbf{d}^{\epsilon}}} \sum_{\mathbf{i}=\mathbf{1}}^{\mathbf{d}^{\epsilon}} \mathrm{He}_{\mathbf{k}}(\langle \mathbf{w}_{\mathbf{i}}^{\star}, \mathbf{x} \rangle)$$

$$\mathrm{He}_m(h(\mathbf{x})) \approx \frac{\mathbf{1}}{\sqrt{\mathbf{m}\mathbf{d}^{\mathbf{m}\epsilon_1}}} \sum_{\text{distinct subsets}} \prod_{\mathbf{s_i}} \mathrm{He}_{\mathbf{k}}(\langle \mathbf{w}_{\mathbf{s_i}}^{\star}, \mathbf{x} \rangle)$$

Composition of Hermites

$$h(\mathbf{x}) = \frac{\mathbf{1}}{\sqrt{\mathbf{d}^{\epsilon}}} \sum_{\mathbf{i=1}}^{\mathbf{d}^{\epsilon}} \mathrm{He}_{\mathbf{k}}(\langle \mathbf{w}_{\mathbf{i}}^{\star}, \mathbf{x} \rangle)$$

$$\mathrm{He}_m(h(\mathbf{x})) \approx \frac{\mathbf{1}}{\sqrt{\mathbf{md}^{\mathbf{m}\epsilon_1}}} \sum_{\text{distinct subsets}} \prod_{\mathbf{s_i}} \mathrm{He}_{\mathbf{k}}(\langle \mathbf{w}_{\mathbf{s_i}}^{\star}, \mathbf{x} \rangle)$$

$$g^{\star}(h_{\star}(\mathbf{x})) \approx \mu_{\mathbf{1}} \mathbf{h}^{\star}(\mathbf{x}) + \underbrace{\mu_{\mathbf{2}}^{\star} \mathbf{He}_{\mathbf{2}}(\mathbf{h}^{\star}(\mathbf{x})) + \ldots}_{\text{High Hermite-degree}}$$

Composition of Hermites

$$h(\mathbf{x}) = \frac{\mathbf{1}}{\sqrt{\mathbf{d}^{\epsilon}}} \sum_{\mathbf{i=1}}^{\mathbf{d}^{\epsilon}} \mathrm{He}_{\mathbf{k}}(\langle \mathbf{w}_{\mathbf{i}}^{\star}, \mathbf{x} \rangle)$$

$$\mathrm{He}_m(h(\mathbf{x})) \approx \frac{\mathbf{1}}{\sqrt{\mathbf{m}\mathbf{d}^{\mathbf{m}\epsilon_1}}} \sum_{\text{distinct subsets}} \prod_{\mathbf{s_i}} \mathrm{He}_{\mathbf{k}}(\langle \mathbf{w}_{\mathbf{s_i}}^{\star}, \mathbf{x} \rangle)$$

$$g^{\star}(h_{\star}(\mathbf{x})) \approx \mu_{\mathbf{1}} \mathbf{h}^{\star}(\mathbf{x}) + \underbrace{\mu_{\mathbf{2}}^{\star} \mathbf{He}_{\mathbf{2}}(\mathbf{h}^{\star}(\mathbf{x})) + \dots}_{\text{High Hermite-degree}}$$

Composition of Hermites

$$h(\mathbf{x}) = \frac{\mathbf{1}}{\sqrt{\mathbf{d}^{\epsilon}}} \sum_{\mathbf{i}=\mathbf{1}}^{\mathbf{d}^{\epsilon}} \mathrm{He}_{\mathbf{k}}(\langle \mathbf{w}_{\mathbf{i}}^{\star}, \mathbf{x} \rangle)$$

$$\mathrm{He}_m(h(\mathbf{x})) \approx \frac{\mathbf{1}}{\sqrt{\mathbf{m}\mathbf{d}^{\mathbf{m}\epsilon_1}}} \sum_{\text{distinct subsets}} \prod_{\mathbf{s}_\mathbf{i}} \mathrm{He}_{\mathbf{k}}(\langle \mathbf{w}_{\mathbf{s}_\mathbf{i}}^{\star}, \mathbf{x} \rangle)$$

$$g^{\star}(h_{\star}(\mathbf{x})) \approx \mu_{\mathbf{1}} \mathbf{h}^{\star}(\mathbf{x}) + \underbrace{\mu_{\mathbf{2}}^{\star} \mathbf{He}_{\mathbf{2}}(\mathbf{h}^{\star}(\mathbf{x})) + \ldots}_{\text{High Hermite-degree}}$$

Composition of Hermites

$$h(\mathbf{x}) = \frac{\mathbf{1}}{\sqrt{\mathbf{d}^{\epsilon}}} \sum_{\mathbf{i=1}}^{\mathbf{d}^{\epsilon}} \mathrm{He}_{\mathbf{k}}(\langle \mathbf{w}_{\mathbf{i}}^{\star}, \mathbf{x} \rangle)$$

$$\mathrm{He}_m(h(\mathbf{x})) \approx \frac{\mathbf{1}}{\sqrt{\mathbf{md^{m\epsilon_1}}}} \sum_{\text{distinct subsets}} \prod_{\mathbf{s_i}} \mathrm{He}_{\mathbf{k}}(\langle \mathbf{w}_{\mathbf{s_i}}^{\star}, \mathbf{x} \rangle)$$

$$g^{\star}(h_{\star}(\mathbf{x})) \approx \mu_{\mathbf{1}} \mathbf{h}^{\star}(\mathbf{x}) + \underbrace{\mu_{\mathbf{2}}^{\star} \mathbf{He}_{\mathbf{2}}(\mathbf{h}^{\star}(\mathbf{x})) + \dots}_{\text{High Hermite-degree}}$$

- The dominant term in the expansion is along $h^{\star}(\mathbf{x})$.

Composition of Hermites

$$h(\mathbf{x}) = \frac{\mathbf{1}}{\sqrt{\mathbf{d}^{\epsilon}}} \sum_{\mathbf{i=1}}^{\mathbf{d}^{\epsilon}} \mathrm{He}_{\mathbf{k}}(\langle \mathbf{w}_{\mathbf{i}}^{\star}, \mathbf{x} \rangle)$$

$$\mathrm{He}_m(h(\mathbf{x})) \approx \frac{\mathbf{1}}{\sqrt{\mathbf{m}\mathbf{d}^{\mathbf{m}\epsilon_1}}} \sum_{\text{distinct subsets}} \prod_{\mathbf{s_i}} \mathrm{He}_{\mathbf{k}}(\langle \mathbf{w}_{\mathbf{s_i}}^{\star}, \mathbf{x} \rangle)$$

$$g^{\star}(h_{\star}(\mathbf{x})) \approx \mu_{\mathbf{1}} \mathbf{h}^{\star}(\mathbf{x}) + \underbrace{\mu_{\mathbf{2}}^{\star} \mathbf{He}_{\mathbf{2}}(\mathbf{h}^{\star}(\mathbf{x})) + \ldots}_{\text{High Hermite-degree}}$$

- The dominant term in the expansion is along $h^{\star}(\mathbf{x})$.
- Generalization of the "approximate Stein's Lemma" in Nichani et al. 2023.

Composition of Hermites

$$h(\mathbf{x}) = \frac{\mathbf{1}}{\sqrt{\mathbf{d}^{\epsilon}}} \sum_{\mathbf{i=1}}^{\mathbf{d}^{\epsilon}} \mathrm{He}_{\mathbf{k}}(\langle \mathbf{w}_{\mathbf{i}}^{\star}, \mathbf{x} \rangle)$$

$$\mathrm{He}_m(h(\mathbf{x})) \approx \frac{\mathbf{1}}{\sqrt{\mathbf{m}\mathbf{d}^{\mathbf{m}\epsilon_1}}} \sum_{\text{distinct subsets}} \prod_{\mathbf{s_i}} \mathrm{He}_{\mathbf{k}}(\langle \mathbf{w}_{\mathbf{s_i}}^{\star}, \mathbf{x} \rangle)$$

$$g^{\star}(h_{\star}(\mathbf{x})) \approx \mu_{\mathbf{1}} \mathbf{h}^{\star}(\mathbf{x}) + \underbrace{\mu_{\mathbf{2}}^{\star} \mathbf{He}_{\mathbf{2}}(\mathbf{h}^{\star}(\mathbf{x})) + \ldots}_{\text{High Hermite-degree}}$$

- The dominant term in the expansion is along $h^{\star}(\mathbf{x})$.
- Generalization of the "approximate Stein's Lemma" in Nichani et al. 2023.

Recovery by the first layer

Recovery by the first layer

$$g^\star(h_\star(\mathbf{x})) \approx \mu_1 \mathbf{h}^\star(\mathbf{x}) + \mu_2^\star \mathbf{He}_2(\mathbf{h}^\star(\mathbf{x})) + \dots$$

Recovery by the first layer

$$g^\star(h_\star(\mathbf{x})) \approx \mu_1 \mathbf{h}^\star(\mathbf{x}) + \mu_2^\star \mathbf{He}_2(\mathbf{h}^\star(\mathbf{x})) + \dots$$

Updates to W_1 ≈ SGD on

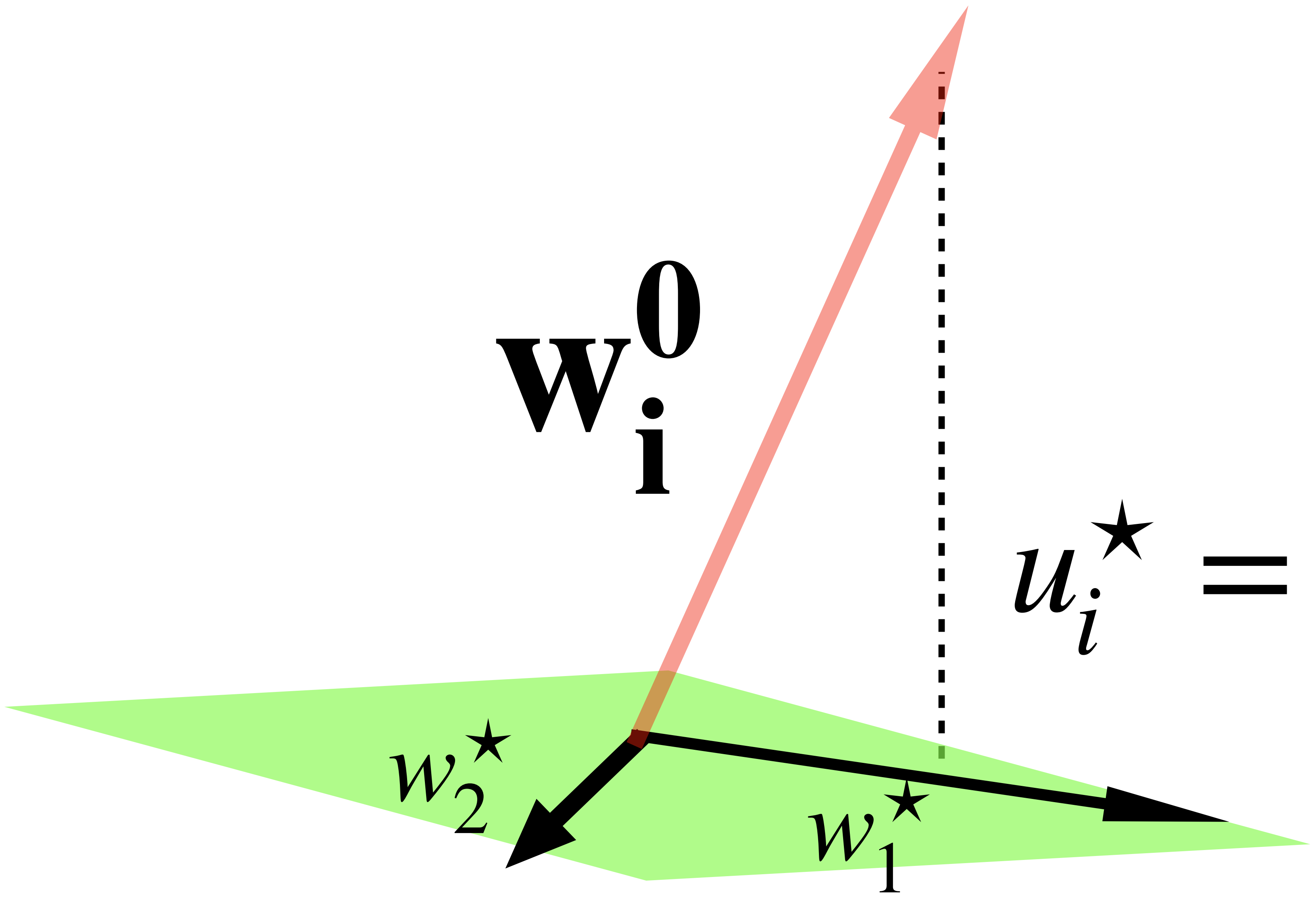
$$h^\star = \mathbf{a}^\star \cdot \frac{P_k\left(W^\star \mathbf{x}\right)}{\sqrt{d^\varepsilon}} \in \mathbb{R}$$

Recovery by the first layer

$$g^\star(h_\star(\mathbf{x})) \approx \mu_1 \mathbf{h}^\star(\mathbf{x}) + \mu_2^\star \mathbf{He}_2(\mathbf{h}^\star(\mathbf{x})) + \dots$$

Updates to $W_1 \approx$ SGD on

$$h^\star = \mathbf{a}^\star \cdot \frac{P_k\left(W^\star \mathbf{x}\right)}{\sqrt{d^\varepsilon}} \in \mathbb{R}$$



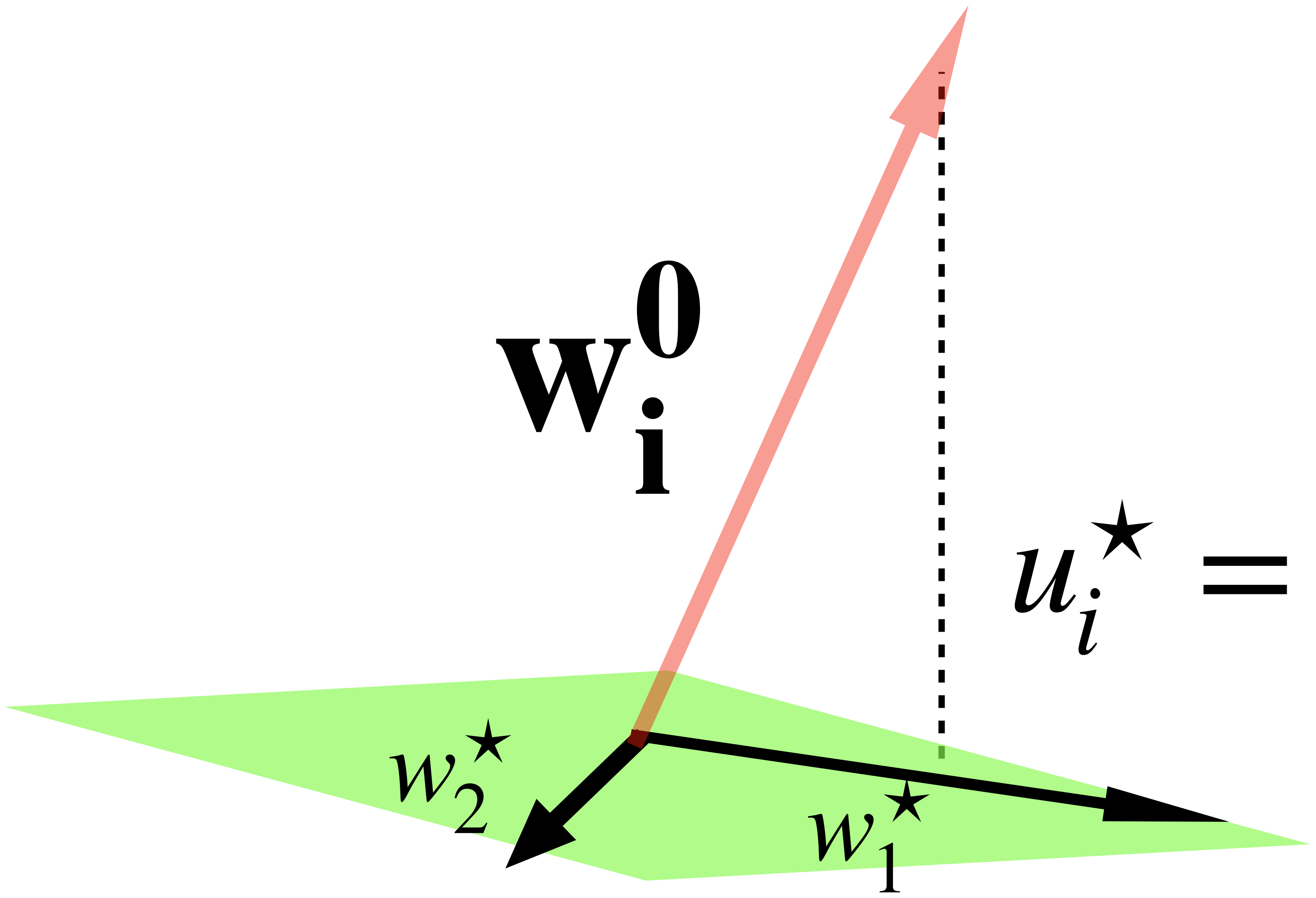
$$u_i^\star = \frac{P_{W^\star} w_i^0}{\|P_{W^\star} w_i^0\|}$$

Recovery by the first layer

$$g^\star(h_\star(\mathbf{x})) \approx \mu_1 \mathbf{h}^\star(\mathbf{x}) + \mu_2^\star \mathbf{He_2}(\mathbf{h}^\star(\mathbf{x})) + \dots$$

Updates to $W_1 \approx$ SGD on

$$h^\star = \mathbf{a}^\star \cdot \frac{P_k\left(W^\star \mathbf{x}\right)}{\sqrt{d^\varepsilon}} \in \mathbb{R}$$



$$u_i^\star = \frac{P_{W^\star} w_i^0}{\|P_{W^\star} w_i^0\|}$$

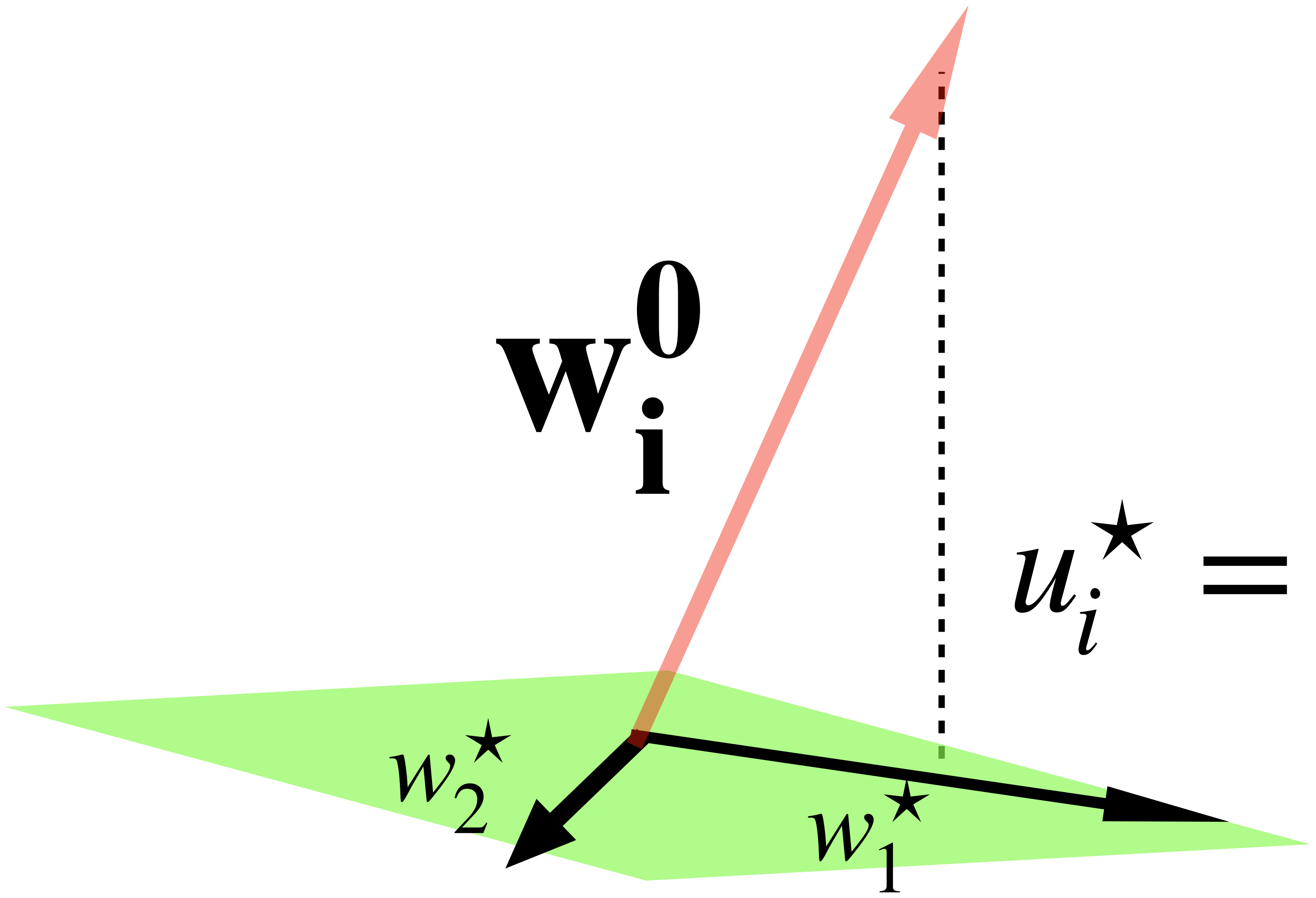
Gradient dominated by initial direction

Recovery by the first layer

$$g^\star(h_\star(\mathbf{x})) \approx \mu_1 \mathbf{h}^\star(\mathbf{x}) + \mu_2^\star \mathbf{He_2}(\mathbf{h}^\star(\mathbf{x})) + \dots$$

Updates to $W_1 \approx$ SGD on

$$h^\star = \mathbf{a}^\star \cdot \frac{P_k\left(W^\star \mathbf{x}\right)}{\sqrt{d^\varepsilon}} \in \mathbb{R}$$



$$u_i^\star = \frac{P_{W^\star} w_i^0}{\|P_{W^\star} w_i^0\|}$$

$$\frac{d\mathbf{w_i}}{dt} \propto u_i^\star + \text{noise}$$

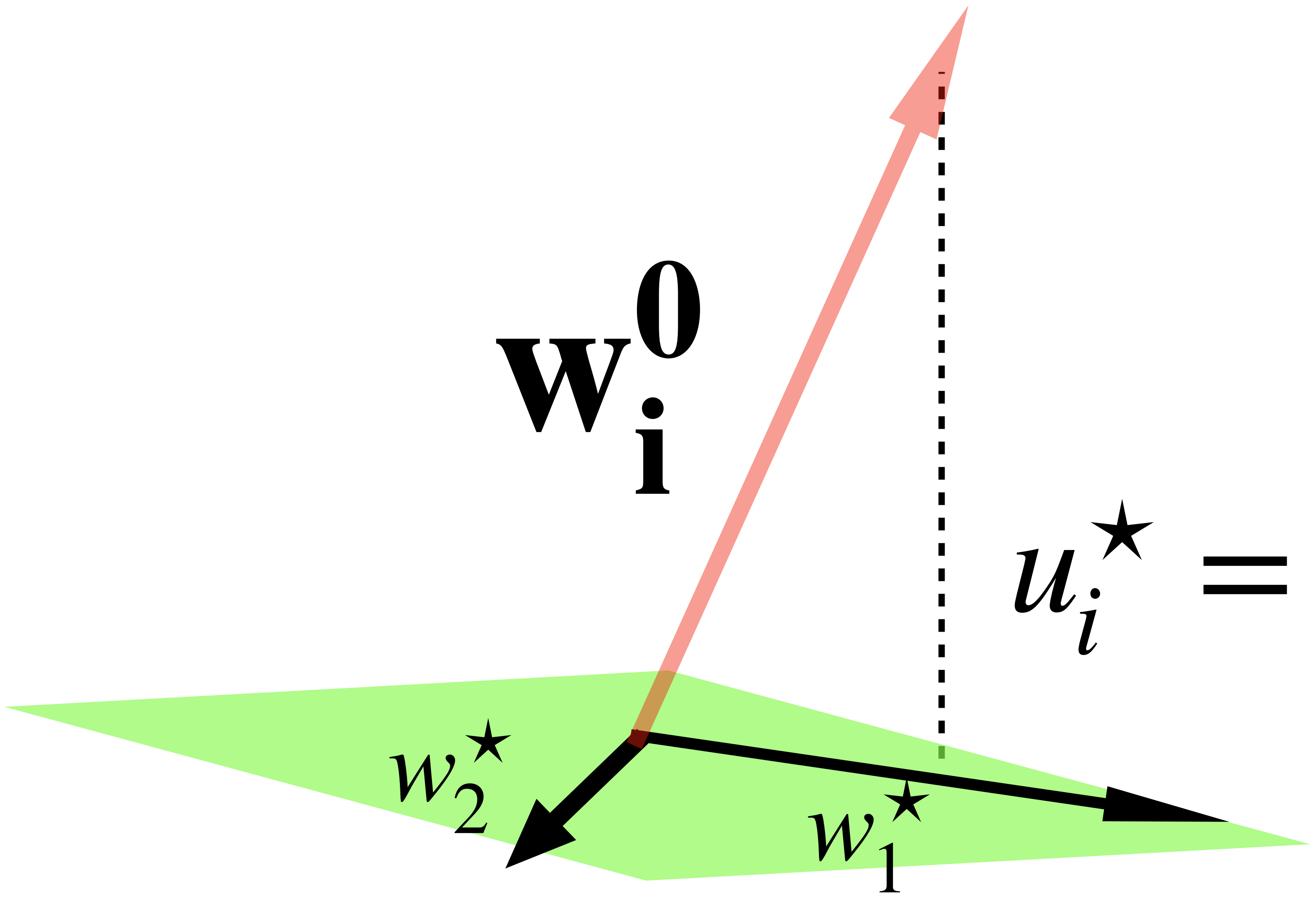
Gradient dominated by initial direction

Recovery by the first layer

$$g^\star(h_\star(\mathbf{x})) \approx \mu_1 \mathbf{h}^\star(\mathbf{x}) + \mu_2^\star \mathbf{He}_2(\mathbf{h}^\star(\mathbf{x})) + \dots$$

Updates to $W_1 \approx$ SGD on

$$h^\star = \mathbf{a}^\star \cdot \frac{P_k\left(W^\star \mathbf{x}\right)}{\sqrt{d^\varepsilon}} \in \mathbb{R}$$



$$u_i^\star = \frac{P_{W^\star} w_i^0}{\|P_{W^\star} w_i^0\|}$$

$$\frac{d\mathbf{w_i}}{dt} \propto u_i^\star + \text{noise}$$

drift+martingale

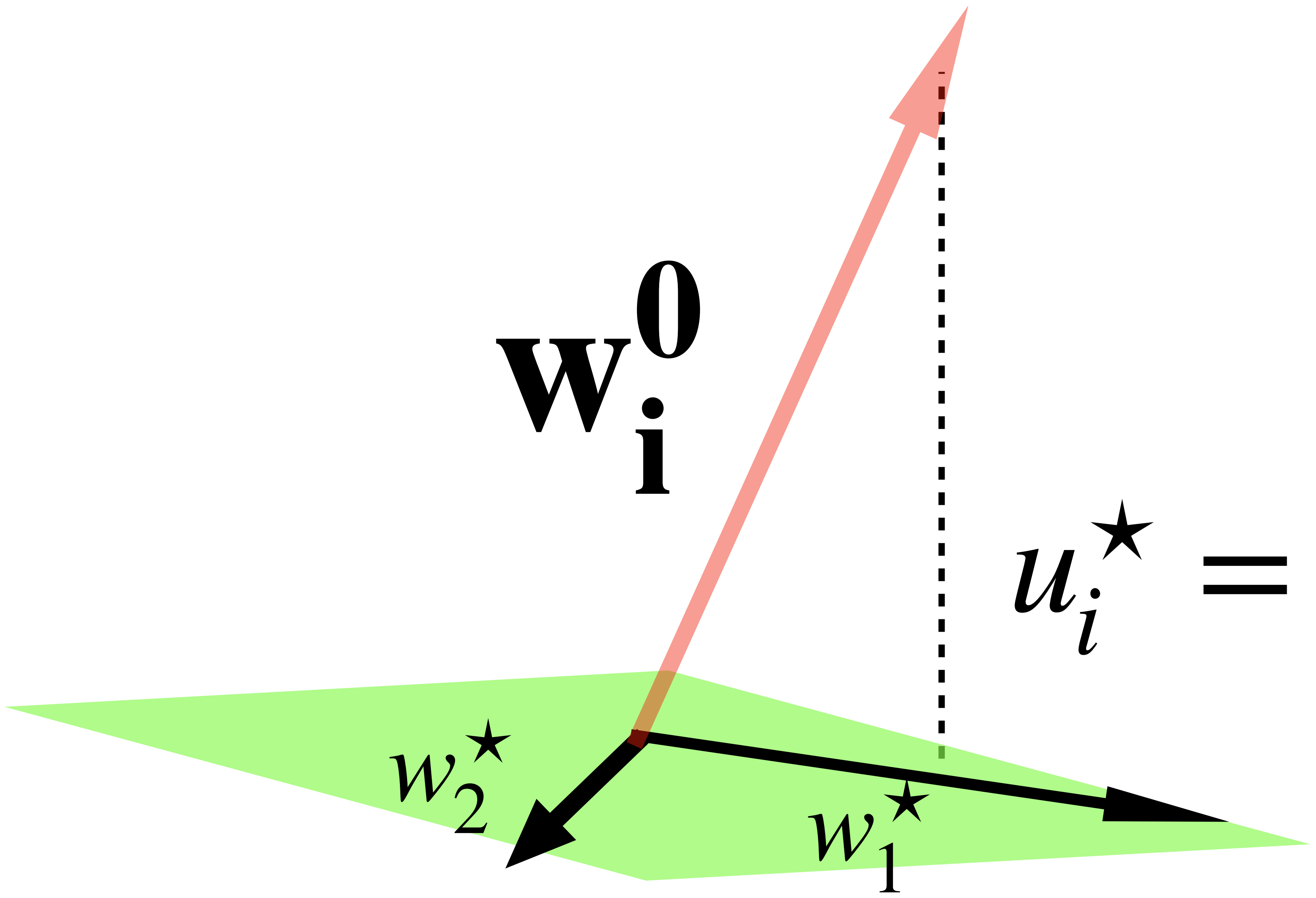
Gradient dominated by initial direction

Recovery by the first layer

$$g^\star(h_\star(\mathbf{x})) \approx \mu_1 \mathbf{h}^\star(\mathbf{x}) + \mu_2^\star \mathbf{He}_2(\mathbf{h}^\star(\mathbf{x})) + \ldots$$

Updates to W_1 ≈ SGD on

$$h^\star = \mathbf{a}^\star \cdot \frac{P_k\left(W^\star \mathbf{x}\right)}{\sqrt{d^\varepsilon}} \in \mathbb{R}$$



$$u_i^\star = \frac{P_{W^\star} w_i^0}{\|P_{W^\star} w_i^0\|}$$

$$\frac{d\mathbf{w_i}}{dt} \propto u_i^\star + \text{noise}$$

drift+martingale

Gradient dominated by initial direction

higher-order terms supressed by vanishing specialization along $w_i^\star$ + vanishing step size.

Recovery by the second layer

Recovery by the second layer

$$\mathbf{h}(\mathbf{x}) = W_2\sigma(\mathbf{W_1}\mathbf{x}) \in \mathbb{R}^{p_2}, X \in \mathbb{R}^{n\times d}$$

Recovery by the second layer

$$\mathbf{h}(\mathbf{x}) = W_2\sigma(\mathbf{W_1x}) \in \mathbb{R}^{p_2}, X \in \mathbb{R}^{n\times d}$$

Gradient updates

Recovery by the second layer

$$\mathbf{h}(\mathbf{x}) = W_2\sigma(\mathbf{W_1x}) \in \mathbb{R}^{p_2}, X \in \mathbb{R}^{n\times d}$$

Gradient updates $\approx$

Recovery by the second layer

$$\mathbf{h}(\mathbf{x}) = W_2\sigma(\mathbf{W_1x}) \in \mathbb{R}^{p_2}, X \in \mathbb{R}^{n\times d}$$

Gradient updates $\approx$ **Projections on the conjugate Kernel***

*Nichani et al. 2023

Recovery by the second layer

$$\mathbf{h}(\mathbf{x}) = W_2\sigma(\mathbf{W_1x}) \in \mathbb{R}^{p_2}, X \in \mathbb{R}^{n\times d}$$

Gradient updates $\approx$ **Projections on the conjugate Kernel***

$$\Delta W_2^i \propto w_i^3\sigma(W_1X^\top)f^\star(X) \odot \sigma'(\mathbf{h_i^t}(\mathbf{X}))$$

*Nichani et al. 2023

Recovery by the second layer

$$\mathbf{h}(\mathbf{x}) = W_2\sigma(\mathbf{W_1x}) \in \mathbb{R}^{p_2}, X \in \mathbb{R}^{n\times d}$$

Gradient updates $\approx$ **Projections on the conjugate Kernel***

$$\Delta h^t_{i,2}(\mathbf{x}) \approx \langle \sigma(\mathbf{W_1x}), \mathbf{\Delta W_i} \rangle$$

$\Delta W_2^i \propto w_i^3 \sigma(W_1 X^\top) f^\star(X) \odot \sigma'(\mathbf{h_i^t}(\mathbf{X}))$ →

*Nichani et al. 2023

Recovery by the second layer

$$\mathbf{h}(\mathbf{x}) = W_2\sigma(\mathbf{W_1x}) \in \mathbb{R}^{p_2}, X \in \mathbb{R}^{n\times d}$$

Gradient updates $\approx$ **Projections on the conjugate Kernel***

$$\Delta h^t_{i,2}(\mathbf{x}) \approx \langle \sigma(\mathbf{W_1x}), \mathbf{\Delta W_i} \rangle$$

$$\Delta W_2^i \propto w_i^3 \sigma(W_1 X^\top) f^\star(X) \odot \sigma'(\mathbf{h_i^t}(\mathbf{X})) \rightarrow \mathbf{\Delta h_i}(\mathbf{x}) \propto \mathbf{w_i^3} \sigma(\mathbf{W_1x})^\top \sigma(\mathbf{W_1X}^\top)\mathbf{f}^\star(\mathbf{X}) \odot \sigma'(\mathbf{h_i^t}(\mathbf{X}))$$

Conjugate Kernel: $\sigma(\mathbf{W_1x})^\top \sigma(\mathbf{W_1X}^\top)$

Perturbed target: $\mathbf{f}^\star(\mathbf{X}) \odot \sigma'(\mathbf{h_i^t}(\mathbf{X}))$

*Nichani et al. 2023

Recovery by the second layer

$$\mathbf{h}(\mathbf{x}) = W_2\sigma(\mathbf{W_1x}) \in \mathbb{R}^{p_2}, X \in \mathbb{R}^{n\times d}$$

Gradient updates $\approx$ **Projections on the conjugate Kernel***

$$\Delta h^t_{i,2}(\mathbf{x}) \approx \langle \sigma(\mathbf{W_1x}), \mathbf{\Delta W_i} \rangle$$

$$\Delta W_2^i \propto w_i^3 \sigma(W_1 X^\top) f^\star(X) \odot \sigma'(\mathbf{h^t_i}(\mathbf{X})) \rightarrow \mathbf{\Delta h_i(x)} \propto \mathbf{w_i^3}\sigma(\mathbf{W_1x})^\top \sigma(\mathbf{W_1X}^\top)\mathbf{f}^\star(\mathbf{X}) \odot \sigma'(\mathbf{h^t_i}(\mathbf{X}))$$

Conjugate Kernel: $\sigma(\mathbf{W_1x})^\top \sigma(\mathbf{W_1X}^\top)$

Perturbed target: $\mathbf{f}^\star(\mathbf{X}) \odot \sigma'(\mathbf{h^t_i}(\mathbf{X}))$

$$\mathbf{\Delta h_i(x)} \propto \mathbf{w_i^3}\sigma(\mathbf{W_1x})^\top \sigma(\mathbf{W_1X}^\top)\mathbf{f}^\star(\mathbf{X})$$

*Nichani et al. 2023

Dimension reduction in action

Dimension reduction in action

$$K_{CK}(\mathbf{x}, \mathbf{x}') = \sigma(\mathbf{W}_1 \mathbf{x})^\top \sigma(\mathbf{W}_1 \mathbf{x}')$$

Dimension reduction in action

$$K_{CK}(\mathbf{x}, \mathbf{x}') = \sigma(\mathbf{W_1 x})^\top \sigma(\mathbf{W_1 x}')$$

$$K_{CK}(\mathbf{x}, \mathbf{x}') = \sum_{\mathbf{i}=1}^{\infty} \mathbf{u_i}(\mathbf{x})\mathbf{u_i}(\mathbf{x}')$$

Dimension reduction in action

$$K_{CK}(\mathbf{x}, \mathbf{x}') = \sigma(\mathbf{W_1 x})^\top \sigma(\mathbf{W_1 x}')$$

$$K_{CK}(\mathbf{x}, \mathbf{x}') = \sum_{\mathbf{i=1}}^{\infty} \mathbf{u_i}(\mathbf{x})\mathbf{u_i}(\mathbf{x}')$$

Dimension reduction in action

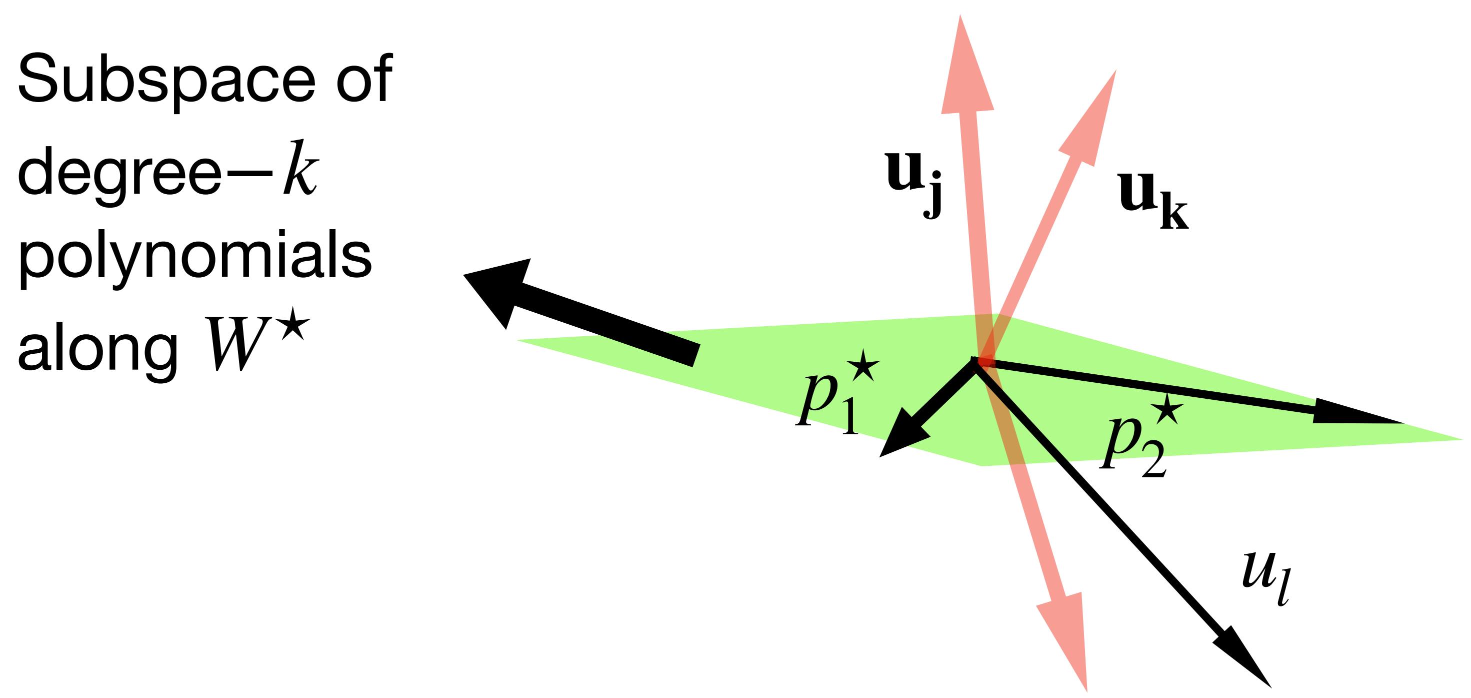
$$K_{CK}(\mathbf{x}, \mathbf{x}') = \sigma(\mathbf{W_1 x})^\top \sigma(\mathbf{W_1 x}')$$

$$K_{CK}(\mathbf{x}, \mathbf{x}') = \sum_{\mathbf{i=1}}^{\infty} \mathbf{u_i}(\mathbf{x})\mathbf{u_i}(\mathbf{x}')$$

Dimension reduction in action

$$K_{CK}(\mathbf{x}, \mathbf{x}') = \sigma(\mathbf{W_1 x})^\top \sigma(\mathbf{W_1 x}')$$

$$K_{CK}(\mathbf{x}, \mathbf{x}') = \sum_{\mathbf{i=1}}^{\infty} \mathbf{u_i}(\mathbf{x})\mathbf{u_i}(\mathbf{x}')$$

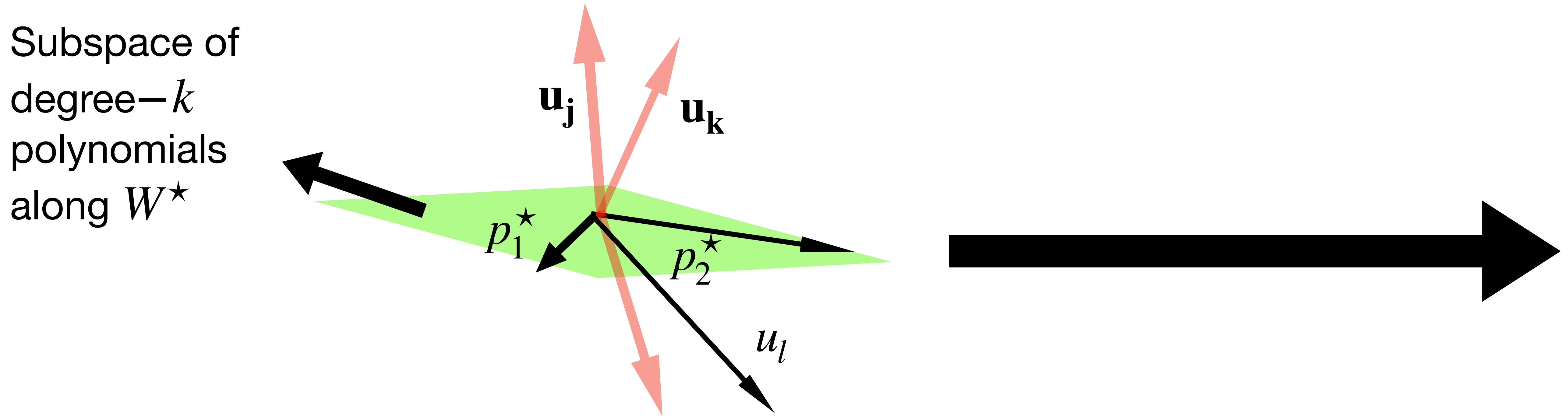


$K_{CK}(\mathbf{x}, \mathbf{x}')$ before training W_1

Dimension reduction in action

$$K_{CK}(\mathbf{x}, \mathbf{x}') = \sigma(\mathbf{W_1 x})^\top \sigma(\mathbf{W_1 x}')$$

$$K_{CK}(\mathbf{x}, \mathbf{x}') = \sum_{\mathbf{i=1}}^{\infty} \mathbf{u_i}(\mathbf{x})\mathbf{u_i}(\mathbf{x}')$$

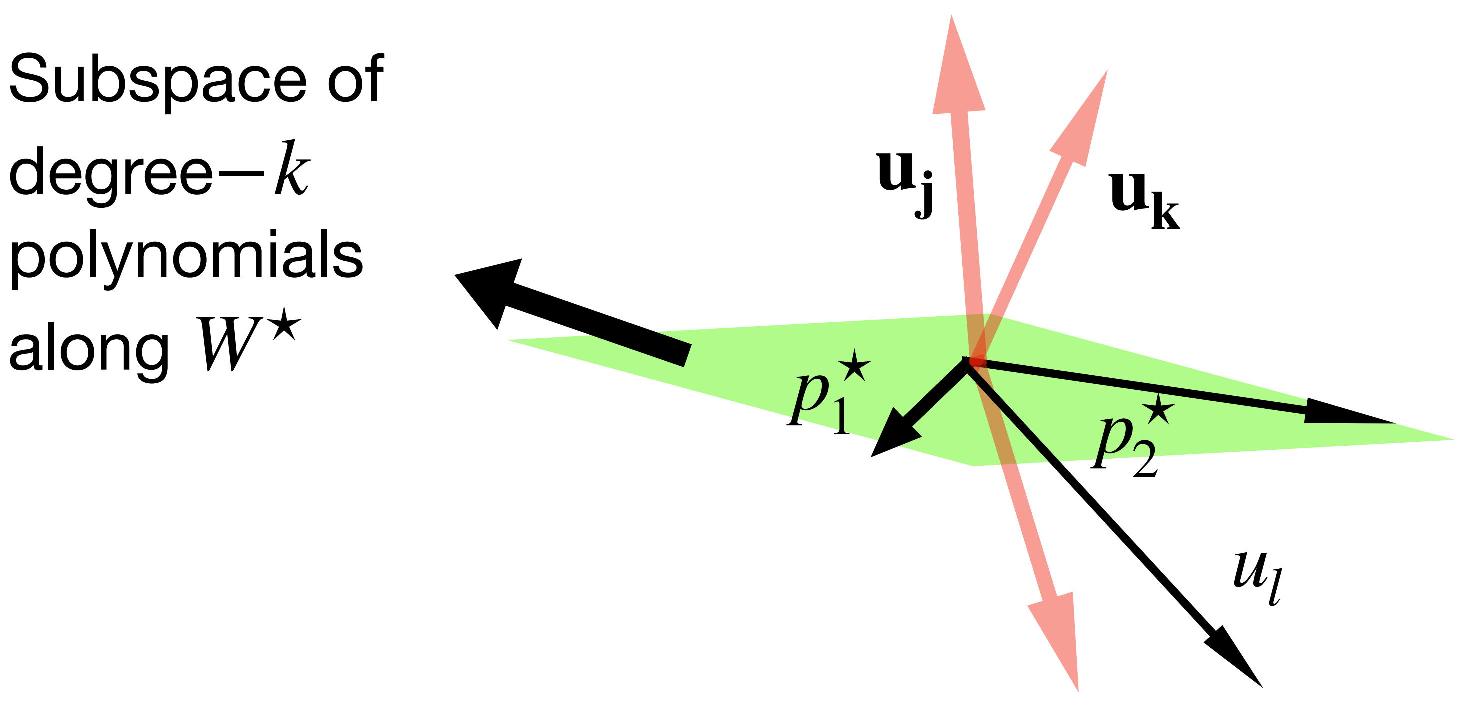


$K_{CK}(\mathbf{x}, \mathbf{x}')$ before training W_1

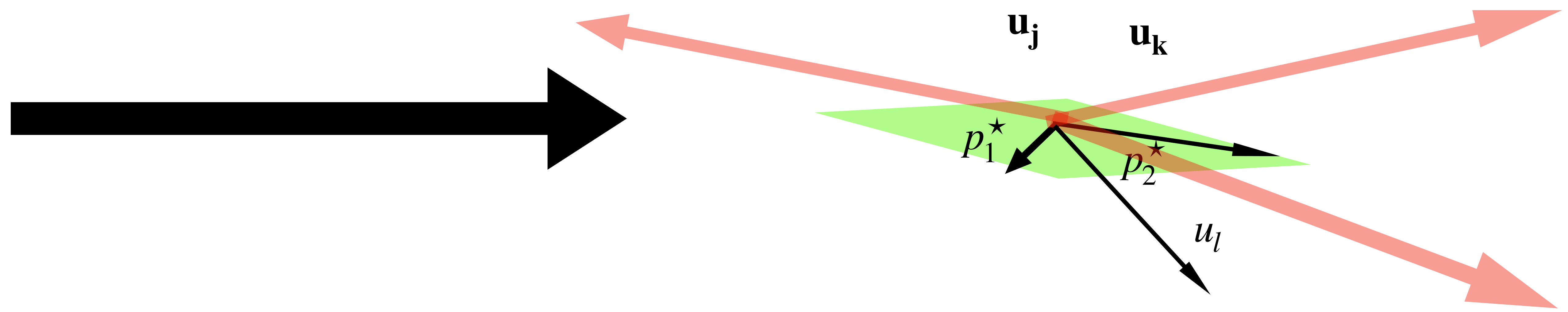
Dimension reduction in action

$$K_{CK}(\mathbf{x}, \mathbf{x}') = \sigma(\mathbf{W_1 x})^\top \sigma(\mathbf{W_1 x}')$$

$$K_{CK}(\mathbf{x}, \mathbf{x}') = \sum_{\mathbf{i=1}}^{\infty} \mathbf{u_i}(\mathbf{x})\mathbf{u_i}(\mathbf{x}')$$



$K_{CK}(\mathbf{x}, \mathbf{x}')$ before training W_1



$K_{CK}(\mathbf{x}, \mathbf{x}')$ after training W_1

Dimension reduction in action

$$K_{CK}(\mathbf{x}, \mathbf{x}') = \sigma(\mathbf{W_1 x})^\top \sigma(\mathbf{W_1 x}')$$

$$K_{CK}(\mathbf{x}, \mathbf{x}') = \sum_{\mathbf{i=1}}^{\infty} \mathbf{u_i}(\mathbf{x})\mathbf{u_i}(\mathbf{x}')$$

$K_{CK}(\mathbf{x}, \mathbf{x}')$ after training W_1

$K_{CK}(\mathbf{x}, \mathbf{x}')$ before training W_1

Caveats:

- Conjugate Kernel ill-conditioned, $\lambda_k = \dfrac{1}{d^{\epsilon k}} \implies \mathcal{O}(d^{\epsilon k})$ steps for convergence $\implies \mathcal{O}(d^{2\epsilon k})$ sample complexity.
- Fix: Pre-conditioning: $W_2 = W_2 - \eta(\frac{1}{n}\sigma(W_1 X^\top)\sigma(W_1 X)^\top)^{-1} \nabla_{W_2}\mathscr{L}$

Spike+ bulk decomposition of Gram matrix

Spike+ bulk decomposition of Gram matrix

$$\mathbf{x}_\star = \mathbf{W}^\star \mathbf{x}$$

Spike+ bulk decomposition of Gram matrix

$$\mathbf{x}_\star = \mathbf{W}^\star \mathbf{x}$$

$$h^\star = \mathbf{a}^\star \cdot \frac{P_k(\mathbf{x}_\star)}{\sqrt{d^\varepsilon}} \in \mathbb{R}$$

Spike+ bulk decomposition of Gram matrix

$$\mathbf{x}_\star = \mathbf{W}^\star \mathbf{x} \qquad h^\star = \mathbf{a}^\star \cdot \frac{P_k\left(\mathbf{x}_\star\right)}{\sqrt{d^\varepsilon}} \in \mathbb{R}$$

$$K \approx \kappa_0^2 \mathbf{1}\mathbf{1}^T + \kappa_1^2 \frac{X_\star X_\star^T}{d} + \kappa_2^2 \frac{H_2(X_\star)H_2(X_\star)^T}{d^2} + \ldots + \kappa_k^2 \frac{H_k(X_\star)H_k(X_\star)^T}{d^k} + \kappa_{k+1}^2 \frac{H_{k+1}(X_\star)H_{k+1}(X_\star)^T}{d^{k+1}} + \ldots$$

Spike+ bulk decomposition of Gram matrix

$$\mathbf{x}_\star = \mathbf{W}^\star \mathbf{x} \qquad h^\star = \mathbf{a}^\star \cdot \frac{P_k\left(\mathbf{x}_\star\right)}{\sqrt{d^\varepsilon}} \in \mathbb{R}$$

$$K \approx \kappa_0^2 \mathbf{1}\mathbf{1}^T + \kappa_1^2 \frac{X_\star X_\star^T}{d} + \kappa_2^2 \frac{H_2(X_\star)H_2(X_\star)^T}{d^2} + \ldots + \kappa_k^2 \frac{H_k(X_\star)H_k(X_\star)^T}{d^k} + \kappa_{k+1}^2 \frac{H_{k+1}(X_\star)H_{k+1}(X_\star)^T}{d^{k+1}} + \ldots$$

$$n = O(d^{k\epsilon+\delta})$$

Spike+ bulk decomposition of Gram matrix

$$\mathbf{x}_\star = \mathbf{W}^\star \mathbf{x} \qquad h^\star = \mathbf{a}^\star \cdot \frac{P_k\left(\mathbf{x}_\star\right)}{\sqrt{d^\varepsilon}} \in \mathbb{R}$$

$$K \approx \kappa_0^2 \mathbf{1}\mathbf{1}^T + \kappa_1^2 \frac{X_\star X_\star^T}{d} + \kappa_2^2 \frac{H_2(X_\star)H_2(X_\star)^T}{d^2} + \ldots + \kappa_k^2 \frac{H_k(X_\star)H_k(X_\star)^T}{d^k} + \kappa_{k+1}^2 \frac{H_{k+1}(X_\star)H_{k+1}(X_\star)^T}{d^{k+1}} + \ldots$$

$$n = O(d^{k\epsilon+\delta})$$

Generalization error of random features and kernel methods: hypercontractivity and kernel matrix concentration

Song Mei[*] Theodor Misiakiewicz[†] Andrea Montanari[†‡]

Spike+ bulk decomposition of Gram matrix

$$\mathbf{x}_\star = \mathbf{W}^\star \mathbf{x} \qquad h^\star = \mathbf{a}^\star \cdot \frac{P_k\left(\mathbf{x}_\star\right)}{\sqrt{d^\varepsilon}} \in \mathbb{R}$$

$$K \approx \kappa_0^2 \mathbf{1}\mathbf{1}^T + \kappa_1^2 \frac{X_\star X_\star^T}{d} + \kappa_2^2 \frac{H_2(X_\star)H_2(X_\star)^T}{d^2} + \ldots + \kappa_k^2 \frac{H_k(X_\star)H_k(X_\star)^T}{d^k} + \kappa_{k+1}^2 \frac{H_{k+1}(X_\star)H_{k+1}(X_\star)^T}{d^{k+1}} + \ldots$$

Concentrates to informative spikes

Noise/ Identity

$$n = O(d^{k\epsilon+\delta})$$

Generalization error of random features and kernel methods: hypercontractivity and kernel matrix concentration

Song Mei*, Theodor Misiakiewicz†, Andrea Montanari†‡

Spike+ bulk decomposition of Gram matrix

$$\mathbf{x}_\star = \mathbf{W}^\star \mathbf{x} \qquad h^\star = \mathbf{a}^\star \cdot \frac{P_k(\mathbf{x}_\star)}{\sqrt{d^\varepsilon}} \in \mathbb{R}$$

$$K \approx \kappa_0^2 \mathbf{1}\mathbf{1}^T + \kappa_1^2 \frac{X_\star X_\star^T}{d} + \kappa_2^2 \frac{H_2(X_\star)H_2(X_\star)^T}{d^2} + \ldots + \kappa_k^2 \frac{H_k(X_\star)H_k(X_\star)^T}{d^k} + \kappa_{k+1}^2 \frac{H_{k+1}(X_\star)H_{k+1}(X_\star)^T}{d^{k+1}} + \ldots$$

$$n = O(d^{k\epsilon+\delta})$$

Concentrates to informative spikes

Noise/ Identity

dominant low-degree term along $h^\star(\mathbf{x})$


Generalization error of random features and kernel methods: hypercontractivity and kernel matrix concentration

Song Mei[*] Theodor Misiakiewicz[†] Andrea Montanari[†‡]

Spike+ bulk decomposition of Gram matrix

$$\mathbf{x}_\star = \mathbf{W}^\star \mathbf{x} \qquad h^\star = \mathbf{a}^\star \cdot \frac{P_k(\mathbf{x}_\star)}{\sqrt{d^\varepsilon}} \in \mathbb{R}$$

$$K \approx \kappa_0^2 \mathbf{1}\mathbf{1}^T + \kappa_1^2 \frac{X_\star X_\star^T}{d} + \kappa_2^2 \frac{H_2(X_\star)H_2(X_\star)^T}{d^2} + \ldots + \kappa_k^2 \frac{H_k(X_\star)H_k(X_\star)^T}{d^k} + \kappa_{k+1}^2 \frac{H_{k+1}(X_\star)H_{k+1}(X_\star)^T}{d^{k+1}} + \ldots$$

$$n = O(d^{k\epsilon+\delta})$$

Concentrates to informative spikes

Noise/ Identity

dominant low-degree term along $h^\star(\mathbf{x})$

Generalization error of random features and kernel methods: hypercontractivity and kernel matrix concentration

Song Mei*, Theodor Misiakiewicz†, Andrea Montanari†‡

Caveat: With Gaussian inputs, some radial degree k polynomials require less than $O(d^k)$ samples.

Fitting the last layer

Fitting the last layer

$$\mathbf{h}(\mathbf{x}) = W_2\sigma(\mathbf{W_1}\mathbf{x}) \approx c\mathbf{w_3}\mathbf{h}^\star(\mathbf{x})$$

Fitting the last layer

$$\mathbf{h}(\mathbf{x}) = W_2\sigma(\mathbf{W_1x}) \approx c\mathbf{w_3h}^\star(\mathbf{x})$$

- Reduction to Kernel on low-dimensional features.
- $K(\mathbf{x_1}, \mathbf{x_2}) = \mathbb{E}_{\mathbf{w}\sim\mathcal{N}(\mathbf{0},\mathbf{1})}[\sigma(\mathbf{cwh}^\star(\mathbf{x_1}) + \mathbf{b})\sigma(\mathbf{cwh}^\star(\mathbf{x_2}) + \mathbf{b})]$.
- $\hat{W}_3 \approx$ KRR$(K(\cdot), \mathbf{X}, \mathbf{y})$.
- Sample complexity/width now dimension-independent.

Takeaways

Takeaways

Hierarchical functions with robustness w.r.t intermediate features allow exploitation of depth through dimension reduction

Takeaways

Hierarchical functions with robustness w.r.t intermediate features allow exploitation of depth through dimension reduction

Do we need narrowing of networks? No, consider:

$$f^\star(\mathbf{x}) = g^\star\left(\frac{{\mathbf{a}^\star_1}^\top P_2\left(W_1^\star \mathbf{x}\right)}{\sqrt{d}}, \cdots, \frac{{\mathbf{a}^\star_m}^\top P_2\left(W_m^\star \mathbf{x}\right)}{\sqrt{d}}\right), \mathbf{x} \in \mathbb{R}^d, m = \mathcal{O}(d)$$

Takeaways

Hierarchical functions with robustness w.r.t intermediate features allow exploitation of depth through dimension reduction

Do we need narrowing of networks? No, consider:

$$f^\star(\mathbf{x}) = g^\star\left(\frac{\mathbf{a}_1^{\star\top} P_2\left(W_1^\star \mathbf{x}\right)}{\sqrt{d}}, \cdots, \frac{\mathbf{a}_m^{\star\top} P_2\left(W_m^\star \mathbf{x}\right)}{\sqrt{d}}\right), \mathbf{x} \in \mathbb{R}^d, m = \mathcal{O}(d)$$

"approximately independent" $\mathcal{O}(d)$ features in $\mathcal{O}(d^2)$ space

Thanks to my collaborators!