Deep Learning Meets Sparse Regularization

Rahul Parhi
ECE, UCSD

Mathematics of Machine Learning Session
CMS Winter Meeting
30 November 2024

A Brief History of Neural Networks and AI

1943: McCulloch and Pitts had the vision to introduce artificial intelligence to the world.

BULLETIN OF
MATHEMATICAL BIOPHYSICS
VOLUME 5, 1943

A LOGICAL CALCULUS OF THE
IDEAS IMMANENT IN NERVOUS ACTIVITY

WARREN S. MCCULLOCH AND WALTER PITTS

FROM THE UNIVERSITY OF ILLINOIS, COLLEGE OF MEDICINE,
DEPARTMENT OF PSYCHIATRY AT THE ILLINOIS NEUROPSYCHIATRIC INSTITUTE,
AND THE UNIVERSITY OF CHICAGO

1958: Rosenblatt implemented the first perceptron for learning.

Psychological Review
Vol. 65, No. 6, 1958

THE PERCEPTRON: A PROBABILISTIC MODEL FOR
INFORMATION STORAGE AND ORGANIZATION
IN THE BRAIN [1]

F. ROSENBLATT

Cornell Aeronautical Laboratory

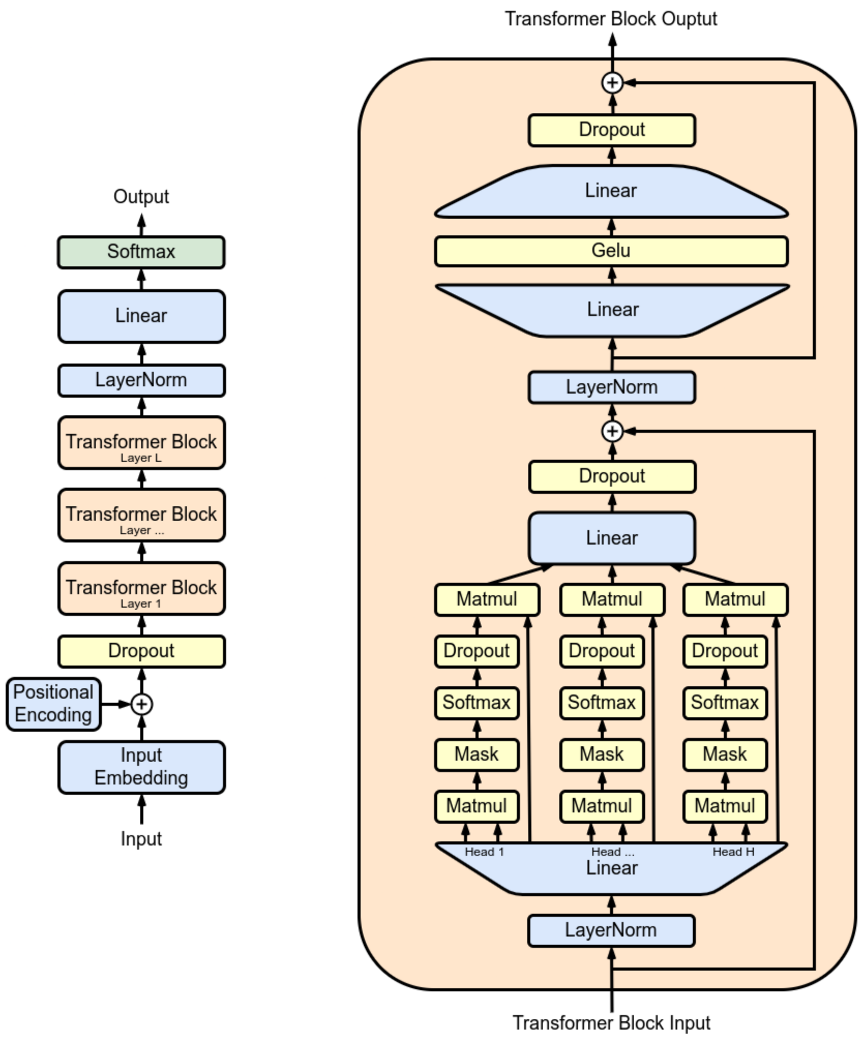
1986: Rumelhart, Hinton, and Williams studied backpropagation for training multilayer perceptrons.

Learning representations by back-propagating errors

David E. Rumelhart*, Geoffrey E. Hinton† & Ronald J. Williams*

* Institute for Cognitive Science, C-015, University of California, San Diego, La Jolla, California 92093, USA
† Department of Computer Science, Carnegie-Mellon University, Pittsburgh, Philadelphia 15213, USA

The World Is Now Based on Neural Networks



Large language models (LLMs) like generative pre-trained transformers (GPT) have taken the world by storm.

- ChatGPT
- Claude

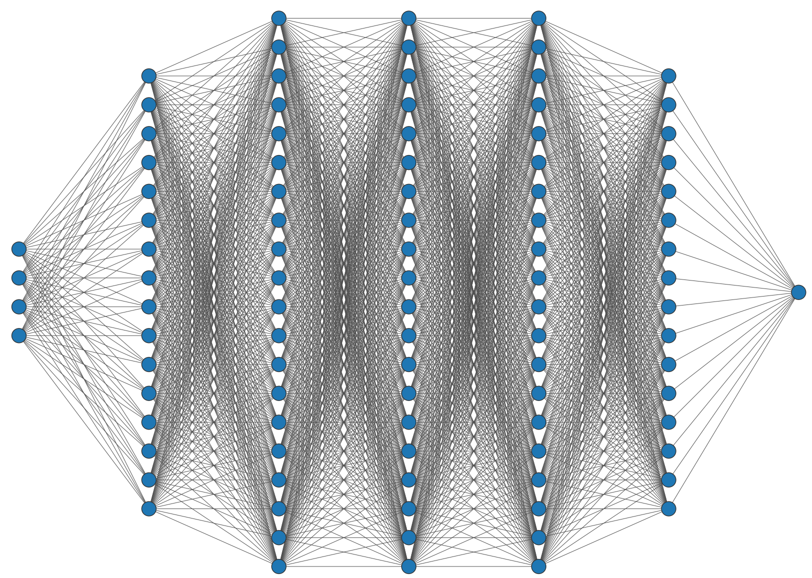
Do we even understand why neural networks work?

[PDF] Improving language understanding by generative pre-training
A Radford, K Narasimhan, T Salimans, I Sutskever
Natural language understanding comprises a wide range of diverse tasks such as textual entailment, question answering, semantic similarity assessment, and document …
☆ Save 99 Cite Cited by 6469 Related articles

Today's Talk

Understanding **analytic properties** of **trained** neural networks.

$\boldsymbol{x}$



$f_{\boldsymbol{\theta}}(\boldsymbol{x})$

parameterized by a vector $\boldsymbol{\theta} \in \mathbb{R}^P$ of neural network **weights**

Neural network training problem for the data $\{(\boldsymbol{x}_n, y_n)\}_{n=1}^N$.

$$\min_{\boldsymbol{\theta} \in \mathbb{R}^P} \underbrace{\sum_{n=1}^{N} \mathcal{L}(y_n, f_{\boldsymbol{\theta}}(\boldsymbol{x}_n))}_{\text{data fidelity}} + \underbrace{\frac{\lambda}{2} \|\boldsymbol{\theta}\|_2^2}_{\text{regularization}}$$

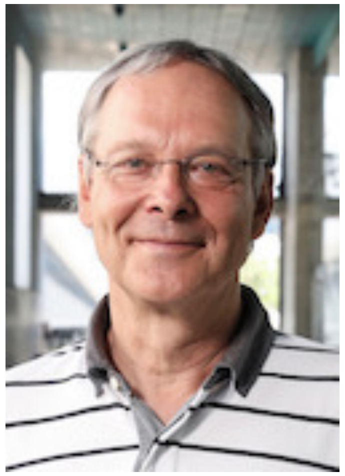
← Tikhonov regularization "weight decay"

We will be **agnostic** to the optimization algorithm.

Collaborators

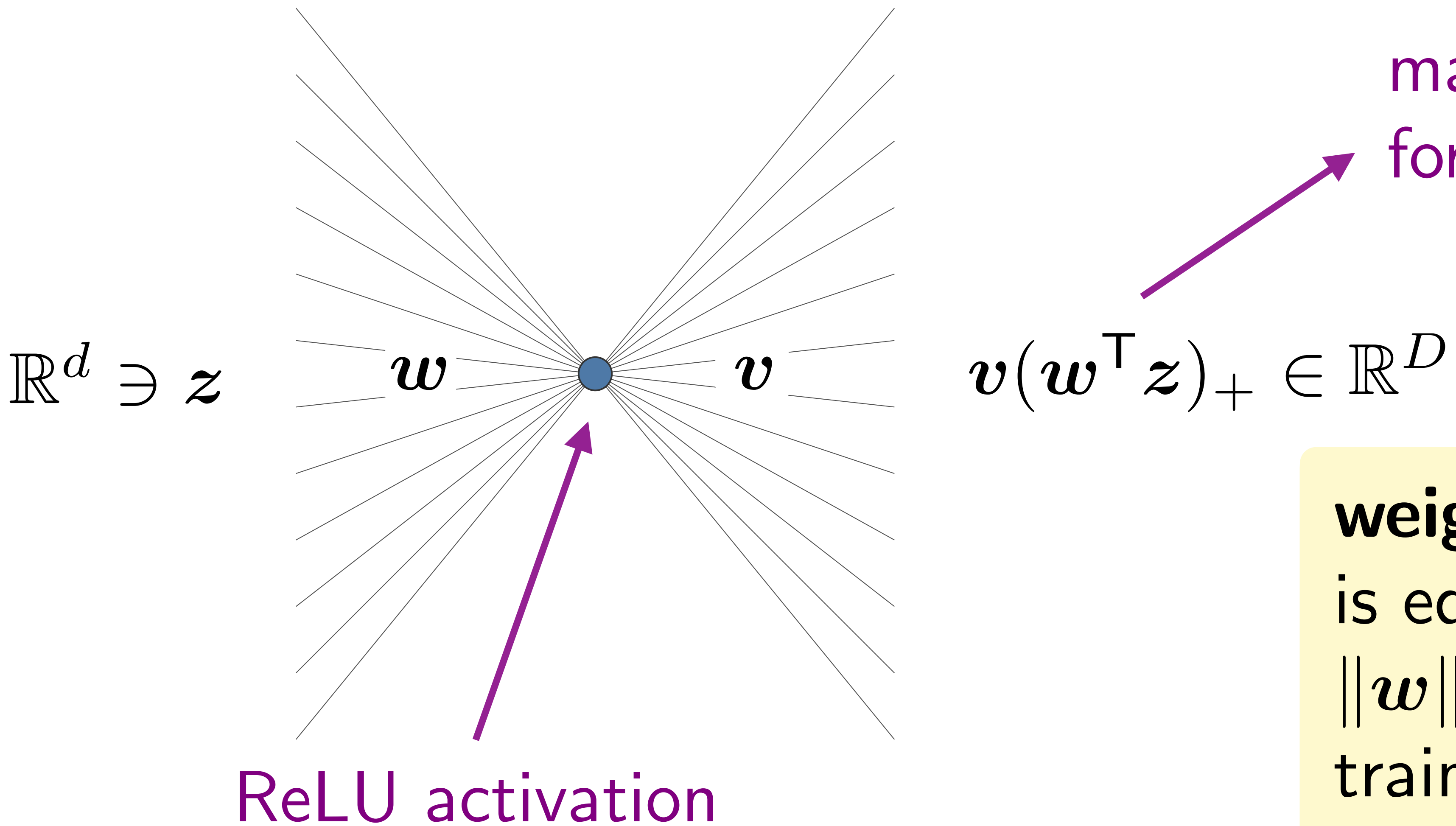
Rob Nowak

Ron DeVore



Michael Unser

Neural Balance in Deep Neural Networks



weight decay in training is equivalent to adding $\|\boldsymbol{w}\|_2^2 + \|\boldsymbol{v}\|_2^2$ to the training objective

Neural Balance Theorem

If a DNN is trained with weight decay, then the 2-norms of the input and output weights to each ReLU neuron must be **balanced**.

$$\|\boldsymbol{w}\|_2 = \|\boldsymbol{v}\|_2$$

P. and Nowak (2023)

Neural Balance

The ReLU activation is **homogeneous**

$$\boldsymbol{v}(\boldsymbol{w}^\top \boldsymbol{z})_+ = \gamma^{-1}\boldsymbol{v}(\gamma\boldsymbol{w}^\top \boldsymbol{z})_+, \quad \text{for any } \gamma > 0.$$

At a global minimizer of the weight decay objective, $\|\boldsymbol{v}\|_2 = \|\boldsymbol{w}\|_2$.

Proof. The solution to

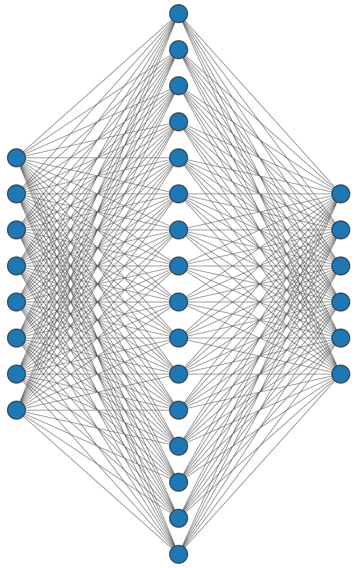
$$\min_{\gamma>0} \|\gamma^{-1}\boldsymbol{v}\|_2 + \|\gamma\boldsymbol{w}\|_2$$

is $\gamma = \sqrt{\|\boldsymbol{v}\|_2/\|\boldsymbol{w}\|_2}$. □

At a global minimizer, $\dfrac{\|\boldsymbol{v}\|_2^2 + \|\boldsymbol{w}\|_2^2}{2} = \|\boldsymbol{v}\|_2\|\boldsymbol{w}\|_2$.

Grandvalet (1998, ICANN)
Neyshabur et al. (2015, ICLR Workshop)

Secret Sparsity of Weight Decay



$$f_{\boldsymbol{\theta}}(\boldsymbol{x}) = \sum_{k=1}^{K} \boldsymbol{v}_k (\boldsymbol{w}_k^{\mathsf{T}} \boldsymbol{x})_+$$

$$\boldsymbol{\theta} = \{(\boldsymbol{w}_k, \boldsymbol{v}_k)\}_{k=1}^{K}$$

weight decay

$$\min_{\boldsymbol{\theta}=\{(\boldsymbol{w}_k,\boldsymbol{v}_k)\}_{k=1}^{K}} \sum_{n=1}^{N} \mathcal{L}(\boldsymbol{y}_n, f_{\boldsymbol{\theta}}(\boldsymbol{x}_n)) + \frac{\lambda}{2} \sum_{k=1}^{K} \|\boldsymbol{v}_k\|_2^2 + \|\boldsymbol{w}_k\|_2^2$$

path-norm

$$\min_{\boldsymbol{\theta}=\{(\boldsymbol{w}_k,\boldsymbol{v}_k)\}_{k=1}^{K}} \sum_{n=1}^{N} \mathcal{L}(\boldsymbol{y}_n, f_{\boldsymbol{\theta}}(\boldsymbol{x}_n)) + \lambda \sum_{k=1}^{K} \|\boldsymbol{v}_k\|_2 \|\boldsymbol{w}_k\|_2$$

multitask lasso

$$\min_{\substack{\boldsymbol{\theta}=\{(\boldsymbol{w}_k,\boldsymbol{v}_k)\}_{k=1}^{K} \\ \|\boldsymbol{w}_k\|_2=1}} \sum_{n=1}^{N} \mathcal{L}(\boldsymbol{y}_n, f_{\boldsymbol{\theta}}(\boldsymbol{x}_n)) + \lambda \sum_{k=1}^{K} \|\boldsymbol{v}_k\|_2$$

Rebalancing

What Kinds of Functions Do Neural Networks Learn?

Path-Norm and Neural Banach Spaces

$$\mathring{\mathcal{V}} = \left\{ f(\boldsymbol{x}) = \sum_{k=1}^{K} \boldsymbol{v}_k (\boldsymbol{w}_k^\mathsf{T} \boldsymbol{x})_+ \,:\, \boldsymbol{v}_k \in \mathbb{R}^D, \boldsymbol{w}_k \in \mathbb{R}^d, K \in \mathbb{N} \right\}$$

finite-width **vector-valued** networks

The path-norm is a **valid norm** on $\mathring{\mathcal{V}}$:

$$\|f\|_{\mathcal{V}} = \sum_{k=1}^{K} \|\boldsymbol{v}_k\|_2 \|\boldsymbol{w}_k\|_2$$

The "completion" of $\mathring{\mathcal{V}}$ (in an appropriate sense) is a Banach space. It is the Banach space $\mathcal{V}$ of all functions of the form

$$f(\boldsymbol{x}) = \int_{\mathbb{S}^{d-1}} (\boldsymbol{w}^\mathsf{T} \boldsymbol{x})_+ \, \mathrm{d}\boldsymbol{\nu}(\boldsymbol{w}).$$

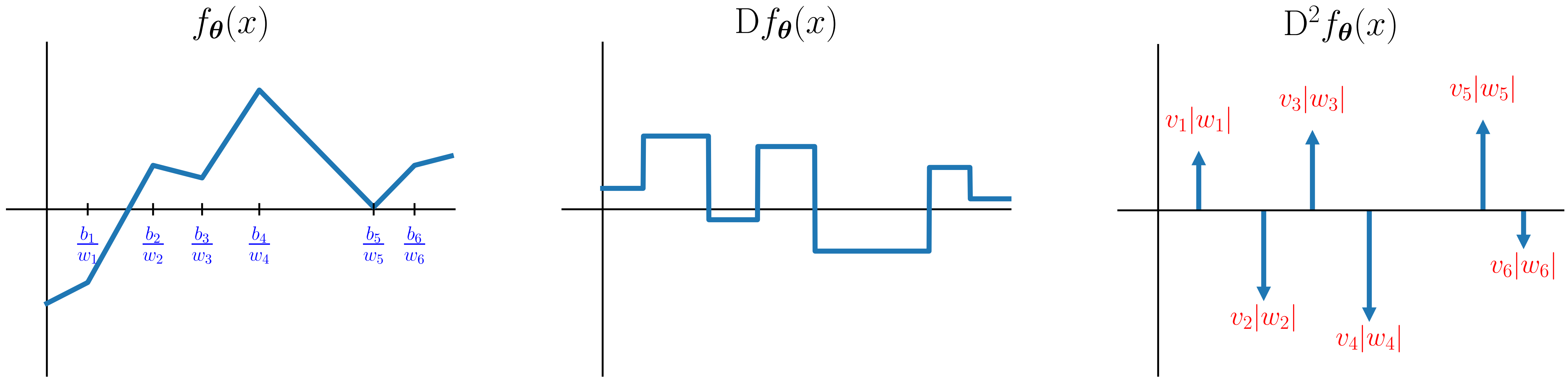
vector-valued measure

"output weights"

Barron (1993, IEEE TIT)
Bach (2017, JMLR)
Ongie et al. (2020, ICLR)
Shenouda, **P.**, Lee, and Nowak (2024, JMLR)

Path-Norm and Derivatives

$$f_{\boldsymbol{\theta}}(x) = \sum_{k=1}^{K} v_k (w_k x - b_k)_+$$



$$\text{path-norm}(f_{\boldsymbol{\theta}}) = \sum_{k=1}^{K} |v_k||w_k| = \int_{-\infty}^{\infty} |\mathrm{D}^2 f_{\boldsymbol{\theta}}(x)| \, \mathrm{d}x$$

More rigorously:
total variation of $\mathrm{D} f_{\boldsymbol{\theta}}$

"How do infinite width bounded norm networks look in function space?"
Pedro Savarese, Itay Evron, Daniel Soudry, and Nathan Srebro
Conference on Learning Theory (2019)

Weight Decay $= \mathrm{TV}(\mathrm{D}f)$-Regularization

$$\min_{\boldsymbol{\theta}=\{(w_k,v_k)\}_{k=1}^K} \sum_{n=1}^N \mathcal{L}(y_n, f_{\boldsymbol{\theta}}(x_n)) + \frac{\lambda}{2}\sum_{k=1}^K |v_k|^2 + |w_k|^2$$

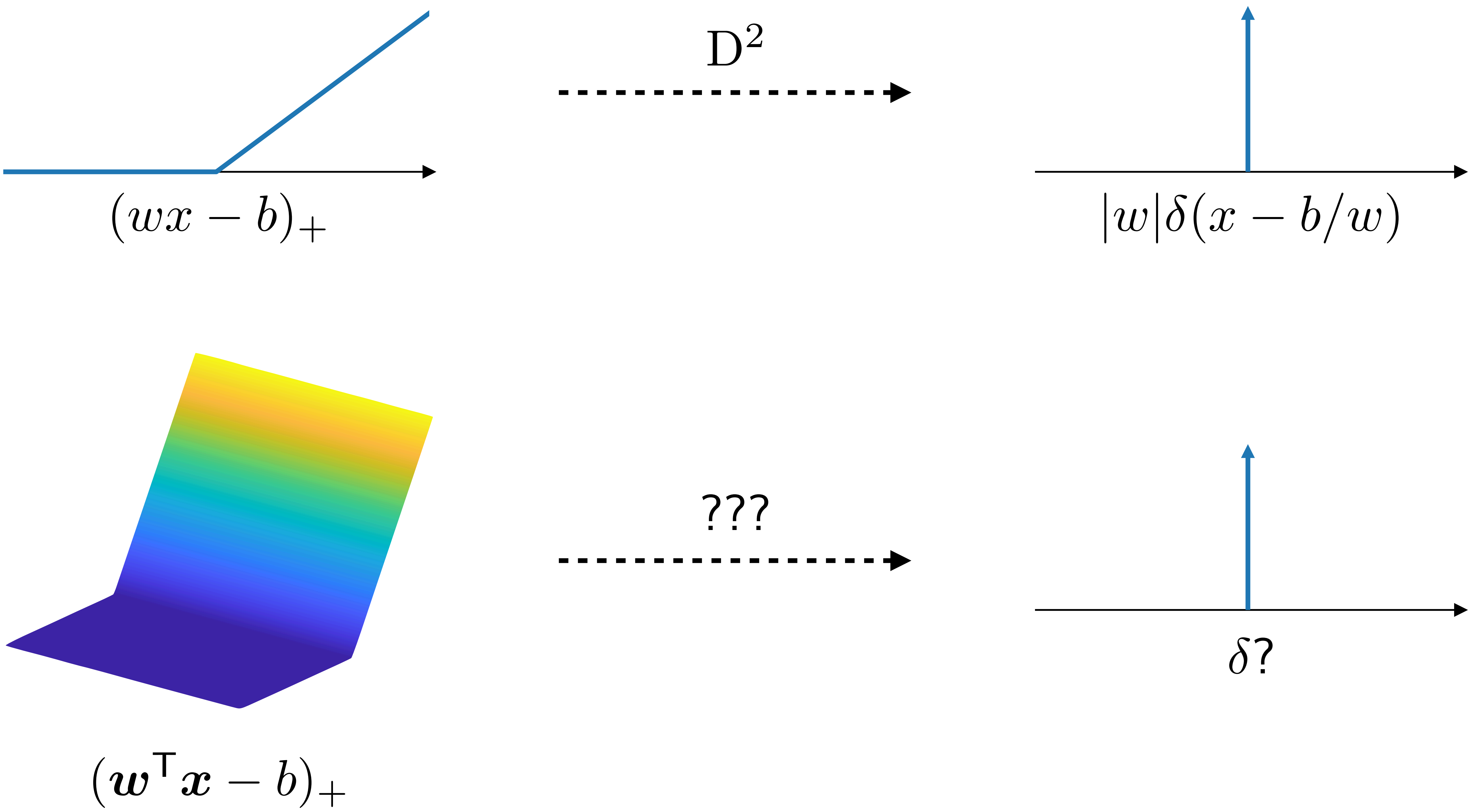
$$\min_{\boldsymbol{\theta}=\{(w_k,v_k)\}_{k=1}^K} \sum_{n=1}^N \mathcal{L}(y_n, f_{\boldsymbol{\theta}}(x_n)) + \lambda\sum_{k=1}^K |v_k||w_k|$$

$$\min_{\boldsymbol{\theta}=\{(w_k,v_k)\}_{k=1}^K} \sum_{n=1}^N \mathcal{L}(y_n, f_{\boldsymbol{\theta}}(x_n)) + \lambda\, \mathrm{TV}(\mathrm{D}\, f_{\boldsymbol{\theta}})$$

→ $\mathrm{TV}^2(f_{\boldsymbol{\theta}})$

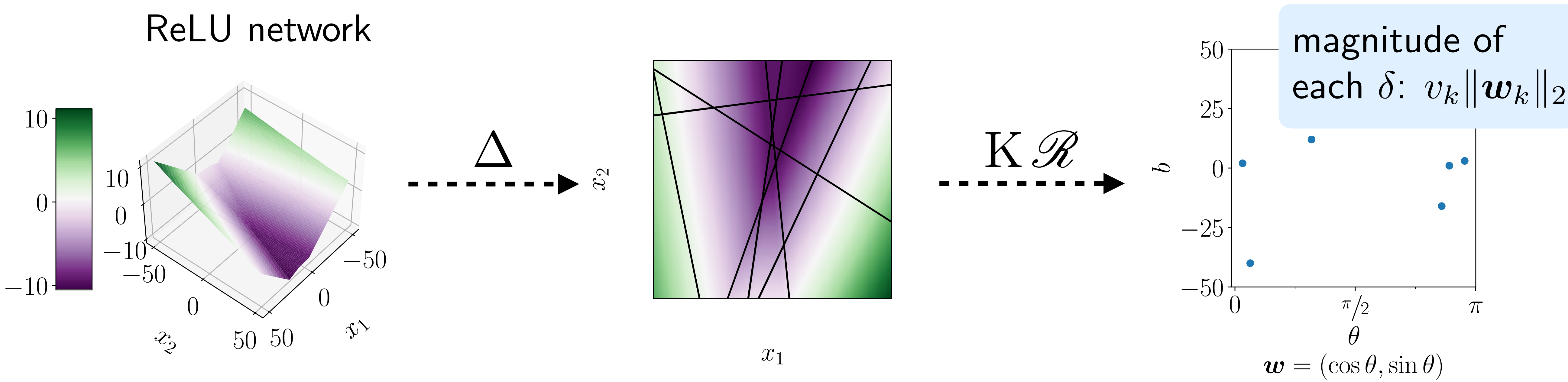
BV^2 is the space of all functions with $\mathrm{TV}^2(f) = \|\mathrm{D}^2 f\|_{\mathcal{M}} < \infty$.

What About the Multivariate Case?



Multivariate Extension: The Radon Transform

$f_{\boldsymbol{\theta}}$ → **differentiate twice** → Dirac "lines" along activation thresholds → **filtered Radon transform** → δ at each neuron weight/bias



$$\text{path-norm}(f_{\boldsymbol{\theta}}) = \sum_{k=1}^{K} |v_k| \|\boldsymbol{w}_k\|_2 = \|\mathrm{K}\mathscr{R}\Delta f_{\boldsymbol{\theta}}\|_{\mathcal{M}}$$

second-order Radon-domain total variation

"A function space view of bounded norm infinite width ReLU nets: The multivariate case"
Greg Ongie, Rebecca Willett, Daniel Soudry, and Nathan Srebro
International Conference on Learning Representations (2020)

The Neural Banach Space $\mathscr{R}\mathrm{BV}^2$

Radon-domain TV^2: $\mathscr{R}\,\mathrm{TV}^2(f) := \|\mathrm{K}\,\mathscr{R}\Delta f\|_{\mathcal{M}}$

total variation of the measure $\mathrm{K}\,\mathscr{R}\Delta f$

$\mathrm{K}\,\mathscr{R}$ = filtered Radon transform

$\widehat{\mathrm{K}g}(\omega) \propto |\omega|^{d-1}\widehat{g}(\omega)$

$$\Delta = \sum_{k=1}^{d} \frac{\partial^2}{\partial x_k^2} = \text{Laplacian operator}$$

Average measure of **sparsity** of second derivatives along each **direction** in $\mathbb{R}^d$.

$\mathscr{R}\,\mathrm{BV}^2$ is the space of all functions on $\mathbb{R}^d$ with $\mathscr{R}\,\mathrm{TV}^2(f) < \infty$.

Banach, not Hilbert!

P. and Nowak (2021, Journal of Machine Learning Research)

A Banach Space Representer Theorem

Neural Network Representer Theorem (**P.** and Nowak 2021)

For any data set $\{(\boldsymbol{x}_n, y_n)\}_{n=1}^N$ and lower semicontinuous $\mathcal{L}(\cdot, \cdot)$, there exists a solution to

$$\min_{f \in \mathscr{R}\,\mathrm{BV}^2} \sum_{n=1}^N \mathcal{L}(y_n, f(\boldsymbol{x}_n)) + \lambda \mathscr{R}\,\mathrm{TV}^2(f), \quad \lambda > 0,$$

that admits a representation of the form

$$f_{\mathrm{ReLU}}(\boldsymbol{x}) = \sum_{k=1}^K v_k \boxed{(\boldsymbol{w}_k^\mathsf{T} \boldsymbol{x} - b_k)_+} + \boxed{\boldsymbol{w}_0^\mathsf{T} \boldsymbol{x} + b_0,} \quad \boxed{K < N.}$$

ReLU neurons — skip connection — sparse solution

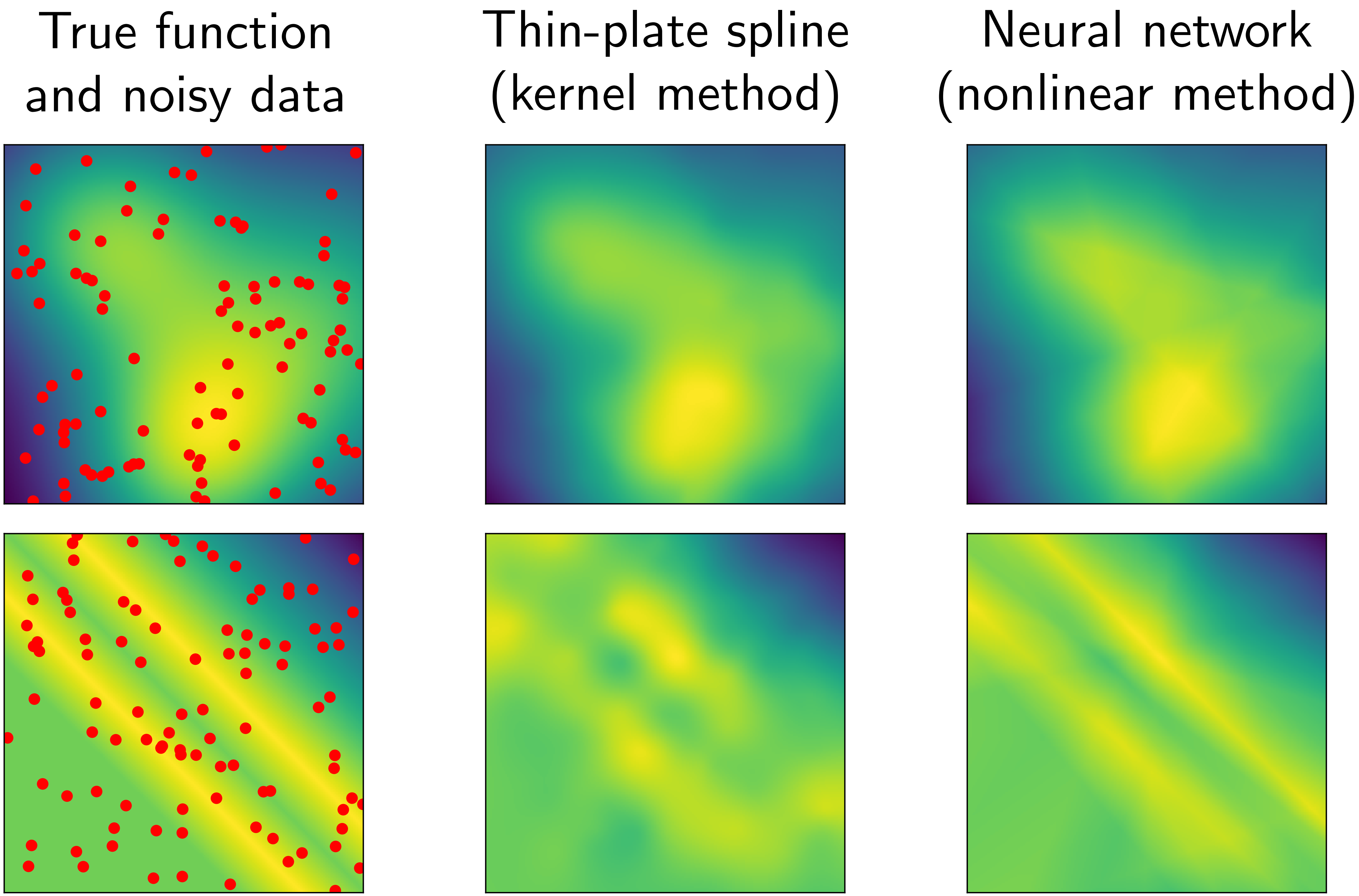
Training a **sufficiently parameterized** neural network ($K \geq N$) with weight decay (to a global minimizer) is a solution to the Banach space problem.

Neural networks learn $\mathscr{R}\,\mathrm{BV}^2$-functions.

P. and Nowak (2021, Journal of Machine Learning Research)

Why Do Neural Networks Work Well in High-Dimensional Problems?

Neural Networks Adapt to Directional Smoothness



Variation in only a **few directions** is a defining characteristic of $\mathscr{R}\,\mathrm{BV}^2$.

P. and Nowak (2023, IEEE Transactions on Information Theory)

Breaking the Curse of Dimensionality?

Given $f \in \mathscr{R}\,\mathrm{BV}^2$, there exists a finite-width ReLU network f_K with K neurons such that

$-\alpha$

$$\|f - f_K\|_{L^\infty(\Omega)} = O(K^{-\frac{1}{2}-\frac{3}{2d}}) = O(K^{-\frac{1}{2}}).$$

Barron (1993)
Matoušek (1996)
Bach (2017)
Siegel (2023)

By the inequality of Carl (1981), this implies

$-\frac{1}{\alpha}$

$$\log \mathcal{N}(\delta, U(\mathscr{R}\,\mathrm{BV}^2), \|\cdot\|_{L^\infty(\Omega)}) = \widetilde{O}(\delta^{-\frac{2d}{d+3}}) = \widetilde{O}(\delta^{-2}).$$

unit ball

Approximation rates and metric entropies **do not grow** with the input dimension d.

Minimax Optimality of Neural Networks

Suppose that $\{\boldsymbol{x}_n\}_{n=1}^N$ are i.i.d. uniform on a bounded domain $\Omega \subset \mathbb{R}^d$. If $y_n = f^\star(\boldsymbol{x}_n) + \varepsilon_n$ with $\mathscr{R}\,\mathrm{TV}^2(f^\star) < \infty$, then any solution to

$$f_{\mathrm{ReLU}} \in \arg\min_{\boldsymbol{\theta}} \sum_{n=1}^N \mathcal{L}(y_n, f_{\boldsymbol{\theta}}(\boldsymbol{x}_n)) + \frac{\lambda}{2} \sum_{k=1}^K |v_k|^2 + \|\boldsymbol{w}_k\|_2^2$$

weight decay objective

satisfies

no curse

$$\mathbf{E}\|f^\star - f_{\mathrm{ReLU}}\|_{L^2(\Omega)}^2 = \widetilde{O}(N^{-\frac{d+3}{2d+3}}) = \widetilde{O}(N^{-\frac{1}{2}}).$$

minimax rate

Linear methods (thin-plate splines, kernel methods, neural tangent kernels, etc.) **necessarily** suffer the curse of dimensionality.

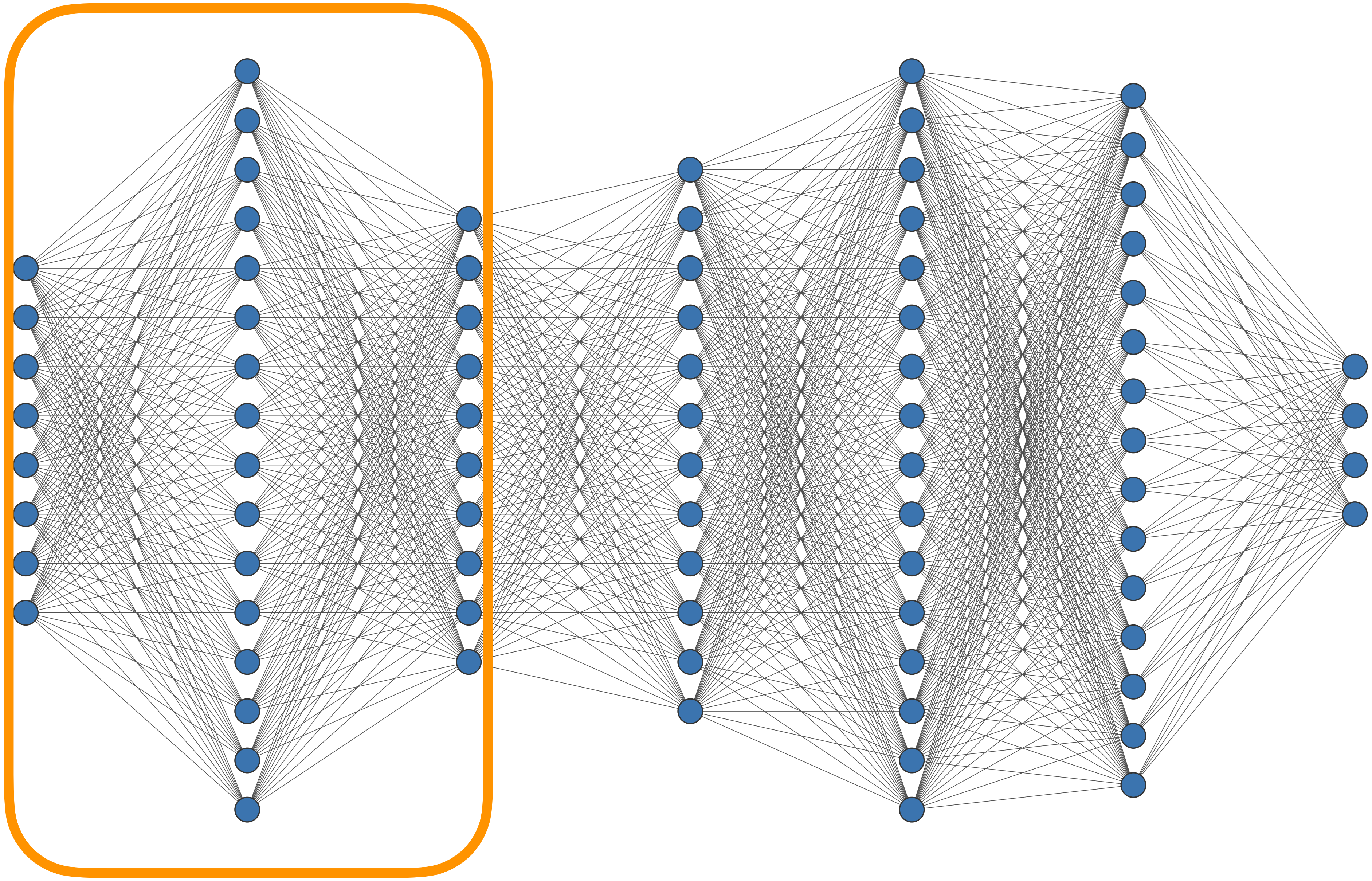
Linear minimax lower bound: $N^{-\frac{3}{d+3}}$

the curse

P. and Nowak (2023, IEEE Transactions on Information Theory)

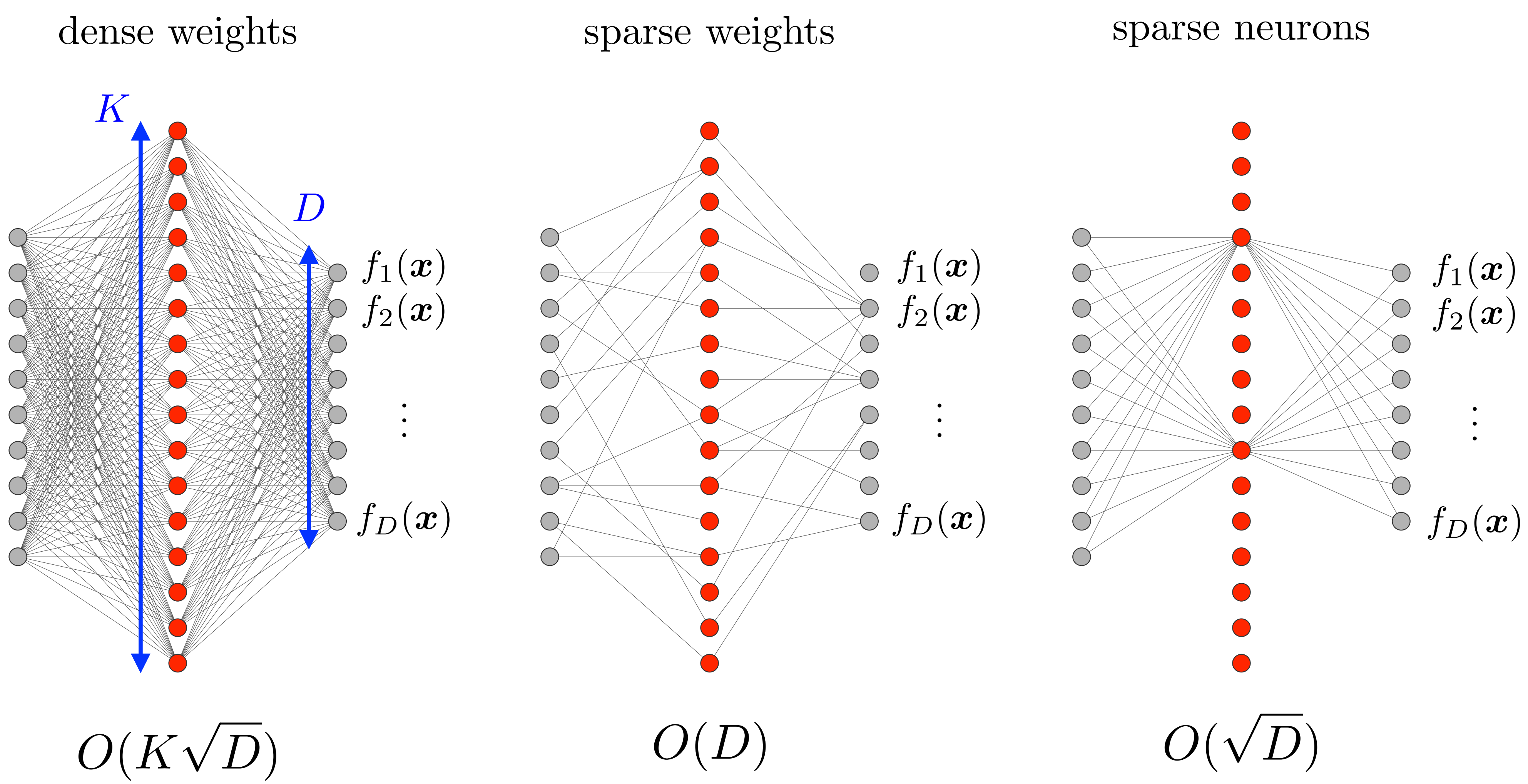
What Does All of This Mean for Learning With **Deep** Neural Networks?

Layers of Vector-Valued Shallow Networks



Deep Neural Networks are **Layers** of Shallow Vector-Valued Networks

The Structured Sparsity of Weight Decay



Weight decay favors outputs that "share" neurons (sparse neurons)

Weight Decay Promotes Neuron Sharing

$$\min_{f \in \mathscr{R}\mathrm{BV}^2} \left(J(f) := \sum_{n=1}^{N} \mathcal{L}(\boldsymbol{y}_n, f(\boldsymbol{x}_n)) + \lambda \mathscr{R}\mathrm{TV}^2(f) \right)$$

$\mathscr{R}\mathrm{TV}^2$ regularization
$\Longleftrightarrow$
path-norm regularization
$\Longleftrightarrow$
weight decay

Neuron Sharing Theorem (Shenouda, P., Lee and Nowak 2024)

Consider **one layer** of a deep neural network

$$f(\boldsymbol{x}) = \sum_{k=1}^{K} \boldsymbol{v}_k (\boldsymbol{w}_k^\mathsf{T} \boldsymbol{x})_+.$$

There exists $\delta > 0$ such that, if $\angle(\boldsymbol{w}_1, \boldsymbol{w}_2) < \delta$, then the neural network that *shares neurons* has a strictly smaller objective value. That is,

$$\widetilde{f}(\boldsymbol{x}) = f(\boldsymbol{x}) - \boldsymbol{v}_1(\boldsymbol{w}_1^\mathsf{T} \boldsymbol{x}) + \widetilde{\boldsymbol{v}}_1(\boldsymbol{w}_2^\mathsf{T} \boldsymbol{x})$$

satisfies $J(\tilde{f}) < J(f)$.

Shenouda, **P.**, Lee, and Nowak (2024, JMLR)

Summary

ReLU neural networks are optimal solutions to data-fitting problems in **new function spaces**:

- Radon-domain **bounded variation** spaces
- **Banach**, not Hilbert
- immune to the **curse of dimensionality**
- solutions are **sparse**/**narrow**
- solutions are **adaptive** to spatial and directional varying smoothness
- weight decay is secretly **sparsity-promoting** regularization scheme
- weight decay promotes **neuron sharing** in **deep neural networks**

Open Problems

What are the fundamental limits of **shallow** networks?

- $\mathscr{R}\mathrm{BV}^2$ does not capture everything DeVore, Nowak, **P.** and Siegel (2025, ACHA)
- Characterization of the **approximation spaces** of shallow networks?

  $\Longrightarrow$ In 1D, these are **Besov spaces** Petrushev (1986)
- **Quantitative** depth separation results?

What kinds of functions do **structured neural architectures** learn?

- Orthogonal weight normalization and pooling layers

  P. and Unser (2025, SIAM J. Math. Data Sci.)

  $\Longrightarrow$ New theory about the distributional k-plane transform

  P. and Unser (2024, SIAM J. Math. Anal.)
- Attention mechanisms and **transformers**?

Questions?