Understanding Neural Networks: A Perspective on Representability and Interpretability

Haiyu Zhang

Department of Mathematics
Southern University of Science and Technology

May 22, 2025

Deep learning: Alchemy or science?

Ali Rahimi, an artificial intelligence (AI) researcher at Google in San Francisco, California, criticized his field in December and received a 40-second standing ovation for it. Speaking at an AI conference, Rahimi argued that machine learning algorithms where computers learn through trial and error have become a form of "alchemy". **He claimed that researchers do not understand why some algorithms succeed while others fail, nor do they have rigorous criteria for selecting one AI architecture over another**.

Outline

- **Representability of Neural Networks**
  Approximation Theory of MLP, RNN, Transformer, etc.
- **Interpretability of Neural Networks**
  Training Dynamics and Generalization

Basic idea of Machine learning: Curve fitting

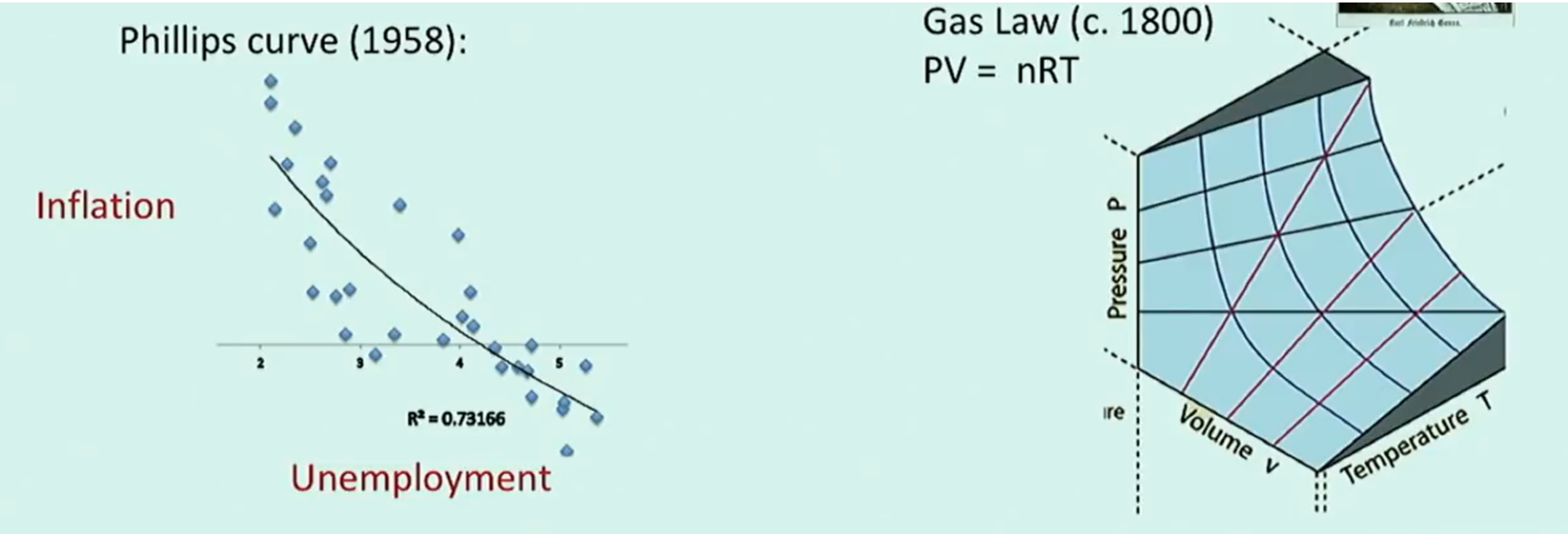

Figure: Surface fitting ("Learning patterns in data")

Neural networks are regarded as high-dimensional function approximators designed to fit the data.

Basic shallow network

Consider the mapping

$$x \mapsto \sum_{j=1}^{m} a_j \sigma(w_j^T x + b_j) \tag{1}$$

- σ: nonlinear activation function.
  Eg. ReLU $z \mapsto \max\{0, z\}$, sigmoid $z \mapsto \frac{1}{1+\exp(-z)}$.
- m: width of the hidden layer
- $((a_j, w_j, b_j))_{j=1}^m$: trainable parameters

Define weight matrix $W \in \mathbb{R}^{m \times d}$ and bias vector $v \in \mathbb{R}^m$ as $W_j := w_j^T$ and $v_j := b_j$. The first layer computes

$$h := \sigma(Wx + b) \in \mathbb{R}^m \quad (\sigma \text{ applied coordinate-wise}),$$

the second computes $h \mapsto a^T h$.

Approximation theory: univariate version

Theorem (Univariate Approximation)

Suppose $g : \mathbb{R} \to \mathbb{R}$ is ρ-Lipschitz. For any $\epsilon > 0$, there exists a 2-layer network f with $\lceil \frac{\rho}{\epsilon} \rceil$ threshold nodes $z \mapsto \mathbf{1}[z \geq 0]$ so that

$$\sup_{x \in [0,1]} |f(x) - g(x)| \leq \epsilon.$$

Proof:(Use step function to localize the target function)

Single hidden layer networks

Consider unbounded width networks with one hidden layer:

$$F_{\sigma,d,m} := F_{d,m} := \left\{ x \mapsto a^T \sigma(Wx + b) : a \in \mathbb{R}^m,\ W \in \mathbb{R}^{m \times d},\ b \in \mathbb{R}^m \right\}.$$

$$F_{\sigma,d} := F_d := \bigcup_{m \geq 0} F_{\sigma,d,m}.$$

Note that $F_{\sigma,m,1}$ denotes networks with a single node, and $F_{\sigma,d}$ is the linear span (in function space) of single-node networks.

Stone-Weierstrass theorem

Definition (Universal Approximator)

A class of functions F is a **universal approximator** over a compact set S if for every continuous function g and target accuracy $\epsilon > 0$, there exists $f \in F$ with

$$\sup_{x \in S} |f(x) - g(x)| \leq \epsilon.$$

Stone-Weierstrass theorem

Theorem (Stone–Weierstrass)

Let functions F be given as follows.

1. Each $f \in F$ is continuous.
2. For every x, there exists $f \in F$ with $f(x) \neq 0$.
3. For every $x \neq x'$, there exists $f \in F$ with $f(x) \neq f(x')$ (F separates points).
4. F is closed under multiplication and vector space operations (F is an algebra).

Then F is a universal approximator: for every continuous $g : \mathbb{R}^d \to \mathbb{R}$ and $\epsilon > 0$, there exists $f \in F$ with

$$\sup_{x \in [0,1]^d} |f(x) - g(x)| \leq \epsilon.$$

Apply it to single hidden layer networks....

We have $F_{\cos,d}$ and $F_{\exp,d}$ are universal approximators. (Check the conditions for Stone-Weierstrass thm!)
Futhermore, we have **general activation universal approximation thm**:

Theorem (Hornik et al., 1989)

Suppose $\sigma : \mathbb{R} \to \mathbb{R}$ is sigmoidal, i.e. it is continuous, and

$$\lim_{z \to -\infty} \sigma(z) = 0, \quad \lim_{z \to +\infty} \sigma(z) = 1.$$

Then $F_{\sigma,d}$ is universal.

Remark:

- ReLU is a "qualified" activation.
- $m \sim O(\frac{1}{\epsilon^d})$ implies the curse of dimensionality.

Infinite-width shallow networks

An **infinite-width shallow network** is characterized by a signed measure ν over weight vectors in $\mathbb{R}^p$:

$$x \mapsto \int \sigma(w^\top x)\, d\nu(w).$$

The mass of ν is the total positive and negative weight mass assigned by

$$|\nu|(\mathbb{R}^p) = \nu_-(\mathbb{R}^p) + \nu_+(\mathbb{R}^p).$$

Barron's theorem

The quantity

$$\int \|\nabla df(w)\| \, dw = 2\pi \int \|w\| \cdot |\hat{f}(w)| \, dw$$

is the **Barron norm** of a function f. The corresponding **Barron class** with norm C is

$$\mathcal{F}_C := \left\{ f : \mathbb{R}^d \to \mathbb{R} : \hat{f} \text{ exists}, \int \|\nabla df(w)\| \, dw \leq C \right\}.$$

Barron's theorem

Theorem (Barron, 1993)

Any continuous function f in Barron class can be approximated by a neural network g with a single hidden layer containing $\mathcal{O}\left(\frac{C^2}{\epsilon}\right)$ hidden units such that $\forall x$ in the domain of f:

$$\mathbb{E}_{x\sim\mu}\left[\|f(x)-g(x)\|^2\right] \leq \epsilon,$$

where C is a constant and μ is the measure (distribution) from which x is picked.

Sketch of the proof: Step 1: Approximate f by a infinite-width shallow networks with nonlinearity cosine (use inverse fourier transform)
Step 2: Probabilistic sampling (Maurey's Lemma, 1981)
Step 3: Turning cosine into other sigmoidal nonlinearity.

Barron's theorem

Remark:

- The assumption for a function in Barron class is quite strong. it requires the Fourier transform $f(\omega)$ to decay sufficiently fast as ω increases.
- If the upperbound of the barron norm $C \sim O(d)$, then the curse of dimensionality can be mitigated.
- The sampling in Step 2 suggests that a finite-width neural network can be viewed as a finite sample (realization) drawn from the underlying stochastic process defined by the infinite-width limit. This insight is closely aligned with later theoretical developments such as the Neural Tangent Kernel (NTK) and Bayesian Neural Networks.

ReLU Deep neural networks

Take the hat function as an example.
We can write the hat function $h : [0,1] \to [0,1]$ as a neural network with 2 layers and 2 neurons:

$$h(x) = 2\sigma(x) - 4\sigma\left(x - \frac{1}{2}\right) = \begin{cases} 2x, & \text{if } 0 \le x < \frac{1}{2}, \\ 2(1-x), & \text{if } \frac{1}{2} \le x \le 1, \end{cases}$$

Observation (Telgarsky, 2016)

For the n-fold composition $h^n(x) := h \circ \cdots \circ h$:

- It generates a sawtooth function with 2^n spikes.
- It consists of 2^n affine linear pieces using only $2n$ neurons.

Deep ReLU networks achieve exponential efficiency in creating linear regions compared to shallow networks.

Width vs depth

Theorem

Let $f^*(x) = x^2$. For any $\epsilon > 0$, there exists a neural network $\tilde{f}$, whose depth and width are $\mathcal{O}(\log(1/\epsilon))$ and $\mathcal{O}(1)$, respectively, such that:

$$\sup_{x \in [0,1]} |\tilde{f}(x) - f^*(x)| \leq \epsilon.$$

A deep architecture could use parameters more efficiently and requires exponentially fewer parame ters to express certain families of functions than a shallow architecture.

Width vs depth

Theorem (Yarotsky, 2017)

Let $d, L \in \mathbb{N}$ with $L \geq 2$, and let $g \in C^2([0,1]^d)$ be a function that is not affine linear. Then there exists a constant $c \in (0, \infty)$ with the following property: For every $\varepsilon \in (0,1)$ and every ReLU neural network architecture $\mathfrak{a} = (N, \sigma) = ((d, N_1, \ldots, N_{L-1}, 1), \sigma)$ with L layers and

$$\|N\|_1 \leq c \cdot \varepsilon^{-1/(2(L-1))},$$

then

$$\inf_{\theta \in \mathbb{R}^{P(N)}} \|\Phi_{\mathfrak{a}}(\cdot, \theta) - g\|_{L^\infty([0,1]^d)} \geq \varepsilon.$$

As the depth L increases, the required width decrease (exponentially) for the same error.

Model comparison of representability

Sparse averaging task

For sparsity q, problem dimension d', and input dimension $d = d' + q + 1$, consider an input $X = (x_1, \ldots, x_N) \in \mathbb{R}^{N \times d}$ with $x_i = (z_i; y_i; i)$ for $z_i \in \mathbb{B}^{d'}$ and $y_i \in \binom{[N]}{q}$. Let the q-sparse average be:

$$qSA(X) = \left(\frac{1}{q} \sum_{j=1}^{q} z_{y_{i,j}} \right)_{i \in [N]}.$$

For accuracy $\epsilon > 0$, a function $f : \mathbb{R}^{N \times d} \to \mathbb{R}^{N \times d'}$ ϵ-approximates qSA if for all X,

$$\max_{i \in [N]} \| f(X)_i - qSA(X)_i \|_2 \leq \epsilon.$$

Model comparison of representability

Here, we have **three different architectures**:

- FC (fully connected neural network)
  $f(x) = \sigma(Wx + b), x \in \mathbb{R}^{Nd}, W \in \mathbb{R}^{m \times Nd}, \sigma : \mathbb{R}^m \to \mathbb{R}^{Nd'}$
- RNN (recurrent neural network)
  $(f(X)_i, h_i) = g_i(x_i, h_{i-1}),$
  $X \in \mathbb{R}^{N \times d}, h_i \in \{0,1\}^m, g_i : \mathbb{R}^d \times \{0,1\}^m \to \mathbb{R}^{d'} \times \{0,1\}^m.$
- Tansformer
  $f_{Q,K,V}(X) = \mathrm{softmax}\left(XQK^\top X^\top\right) XV,$
  $X \in \mathbb{R}^{N \times d}, Q, K \in \mathbb{R}^{d \times m}, V \in \mathbb{R}^{d \times d'}$

We consider the qSA implementation by transformer **efficient** since the dimension of the model parameters grows with $\mathrm{poly}(q, d, \log N)$, whereas the other two are **inefficient** since their parameter dimension grows as $\mathrm{poly}(q, d, N)$.

Model comparison of representability

Theorem

For any $\varepsilon \in (0,1)$, any memory-bounded algorithm that ε-approximates qSA (for $q = 1$ and $d' = 1$) must have memory

$$m \geq \frac{N-1}{2}.$$

Sketch of the proof

1. Transforms the ability of a memory-limited algorithm to approximate qSA into a communication protocol for solving the DISJ problem.
2. By leveraging the communication complexity lower bound ($\Omega(n)$), it follows that the required memory must grow at least linearly.

(Interesting proof using communication complexity. See Thm 11 in Representational Strengths and Limitations of Transformers.)

Summary

These work aims to

- analyze and explain, from the perspective of function approximation, why neural networks perform well on a wide range of tasks, and provide a theoretical guarantee for the approximation capability of neural networks.
- quantify the relationship between approximation error, data size, and model complexity, in order to derive a scaling law, which also provides guidance for model selection.

In practice.....

Given the architecture of a model and the parameters θ to be trained, we have

- θ^*: true parameter
- θ_1: the theoretical optimal parameter within the function class represented by the architecture
- θ_2: the theoretically optimal parameter that can be obtained using the training data and loss function
- θ_3: the parameter obtained in practice using an optimization algorithm (e.g., iterative methods), which may not be a global optimum

While our goal is to obtain the optimal parameter θ^*, in practice we typically end up with a suboptimal solution θ_3. The discrepancy between the two can introduce significant errors that impact model performance. Approximation theory can provide valuable insight and theoretical guidance, though real-world scenarios are often much more complex.

The interpretability problem

Models train millions of parameters to achieve input-output behaviors. But it's difficult to know the "meaning" of the parameters and what happens during the training process.

Weight space of CNN

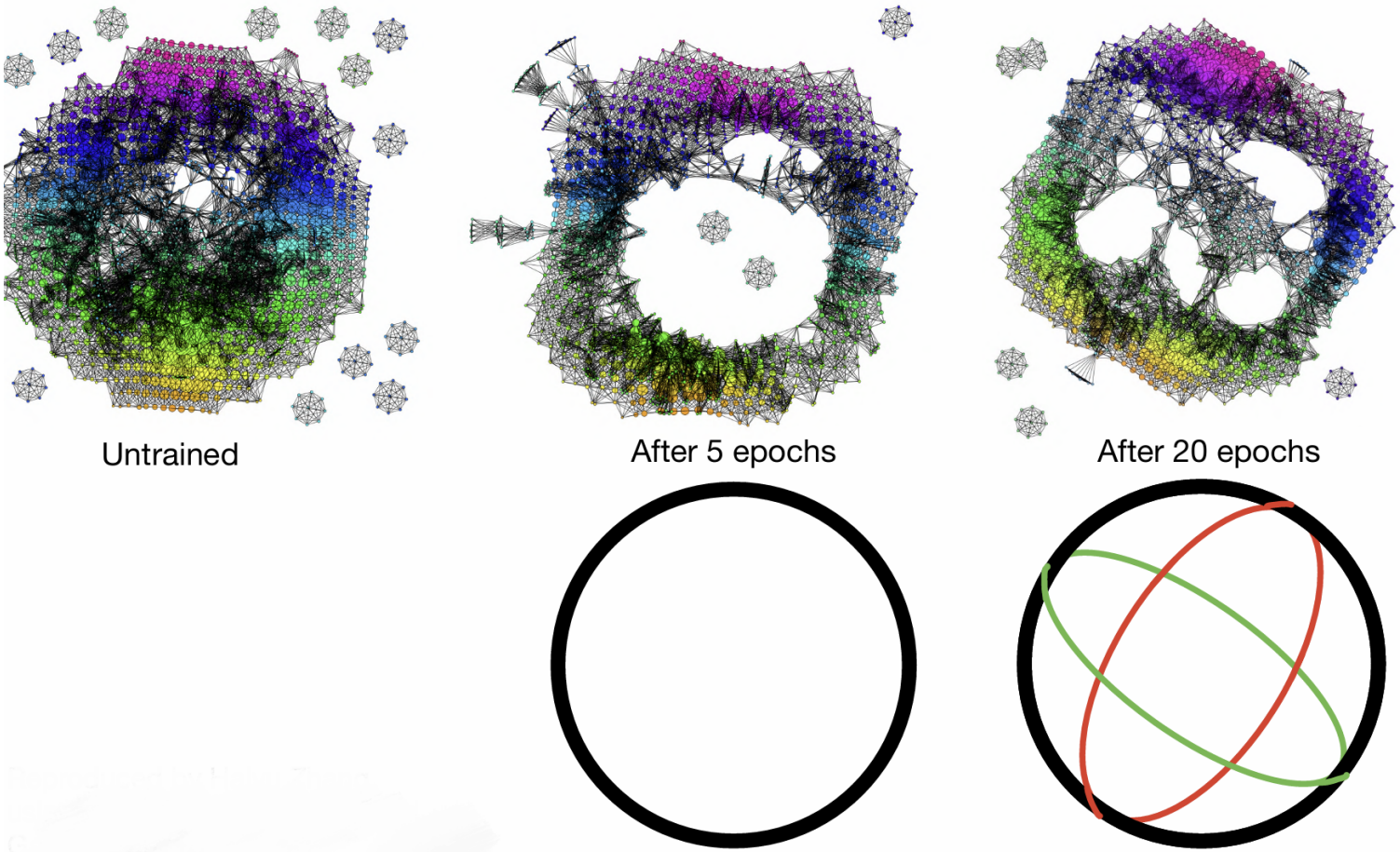

Figure: Emergence of cycles during the training process

Grokking

Grokking is a phenomenon where a neural network, after initially overfitting the training data, suddenly achieves generalization to unseen data after extended training. Despite poor validation performance for a long time, the model eventually "understands" the task and begins to generalize well.

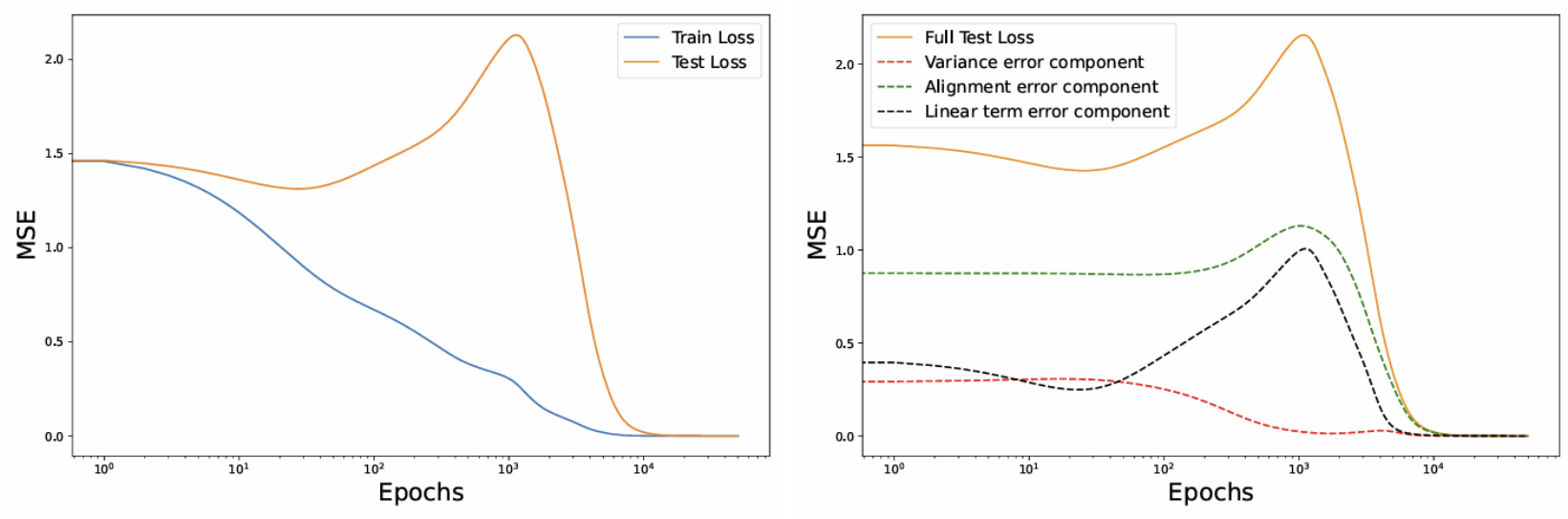

Figure: Grokking as the transition from lazy to rich training dynamics, Kumar et al.

Grokking

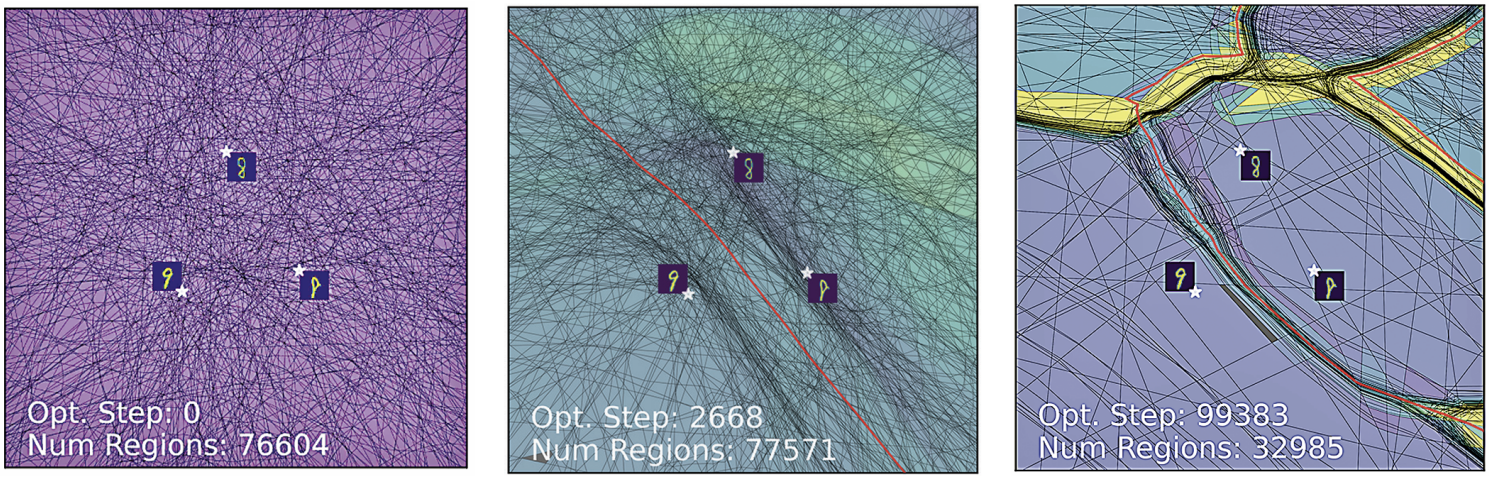

Figure 9. SplineCam visualization of a slice of the input space defined by three training MNIST digits being classified by a four-layer MLP of width 200. The false color map (vividis) encodes the two-norm of the A_ω matrix defined on each tile according to purple (low), green (medium), yellow (high). The decision boundary is depicted in red. (Adapted from [HBB24].)

Figure: On the Geometry of Deep Learning, Balestriero et al.

Local complexity (LC) : the number of tiles in a neighborhood V around a point x in the input space.

Information bottleneck theory (Naftali Tishby et al.,1999)

Models are trying to solve the optimization problem:

$$\min_{T} I(X;T) - \beta I(T;Y),$$

where:

- X is the input random variable,
- Y is the target variable,
- T is the learned representation (a bottleneck variable),
- $I(A;B)$ denotes the mutual information between A and B,
- $\beta > 0$ controls the trade-off between compression and prediction.

In the early training phase, the network **memorizes** the input (increasing $I(X;T)$), In later phases, stochastic gradient descent (SGD) leads to **compression** (decreasing $I(X;T)$), while retaining useful predictive information (maintaining $I(T;Y)$).

Experiments

Weight tracking and feature analysis in consonants classification experiments.

Thanks for your listening!