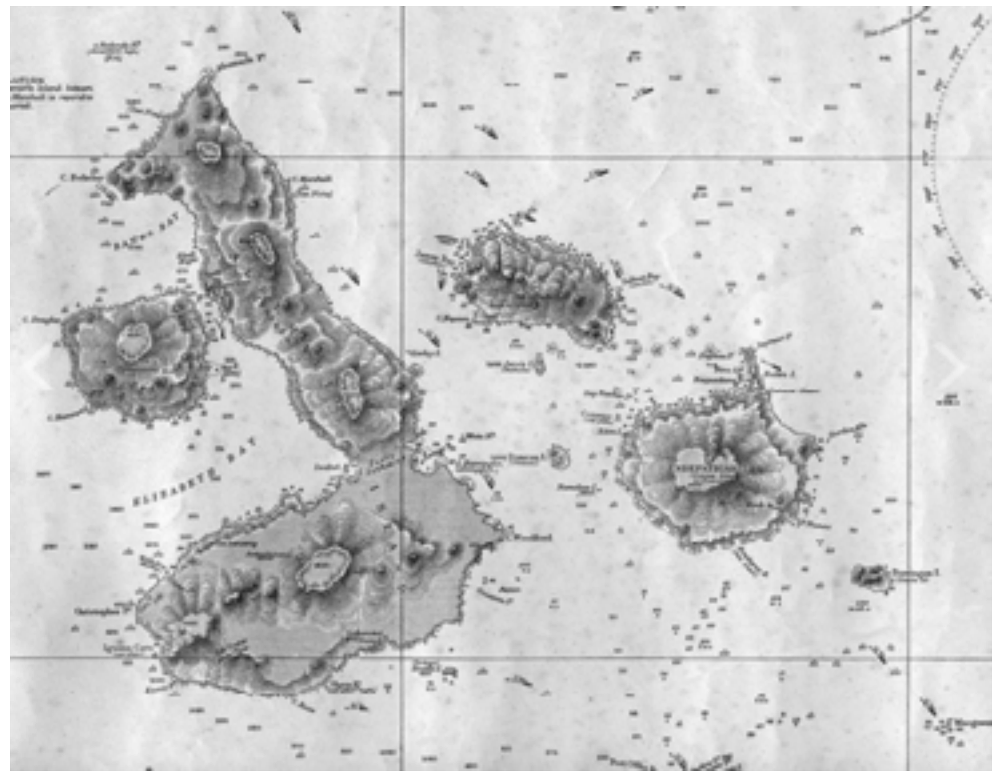
Patterns of Feature Learning and Their Sample Complexity

Zohar Ringel | Cargese, Statistical Mechanics & Machine Learning : Moving Forward | Aug 2025

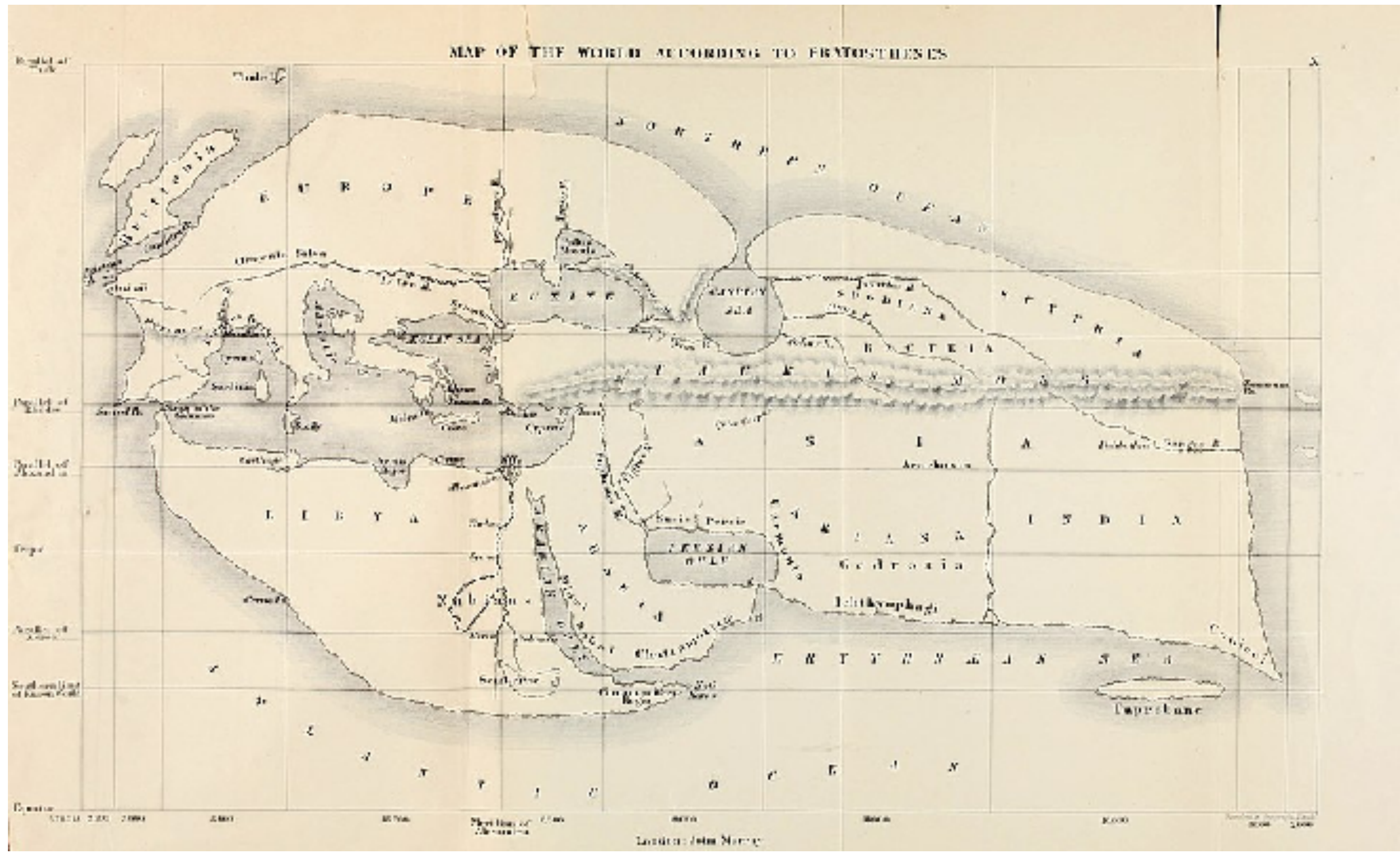
On Rigor in Science (Borges)

Inspired a talk by Solla's at KITP

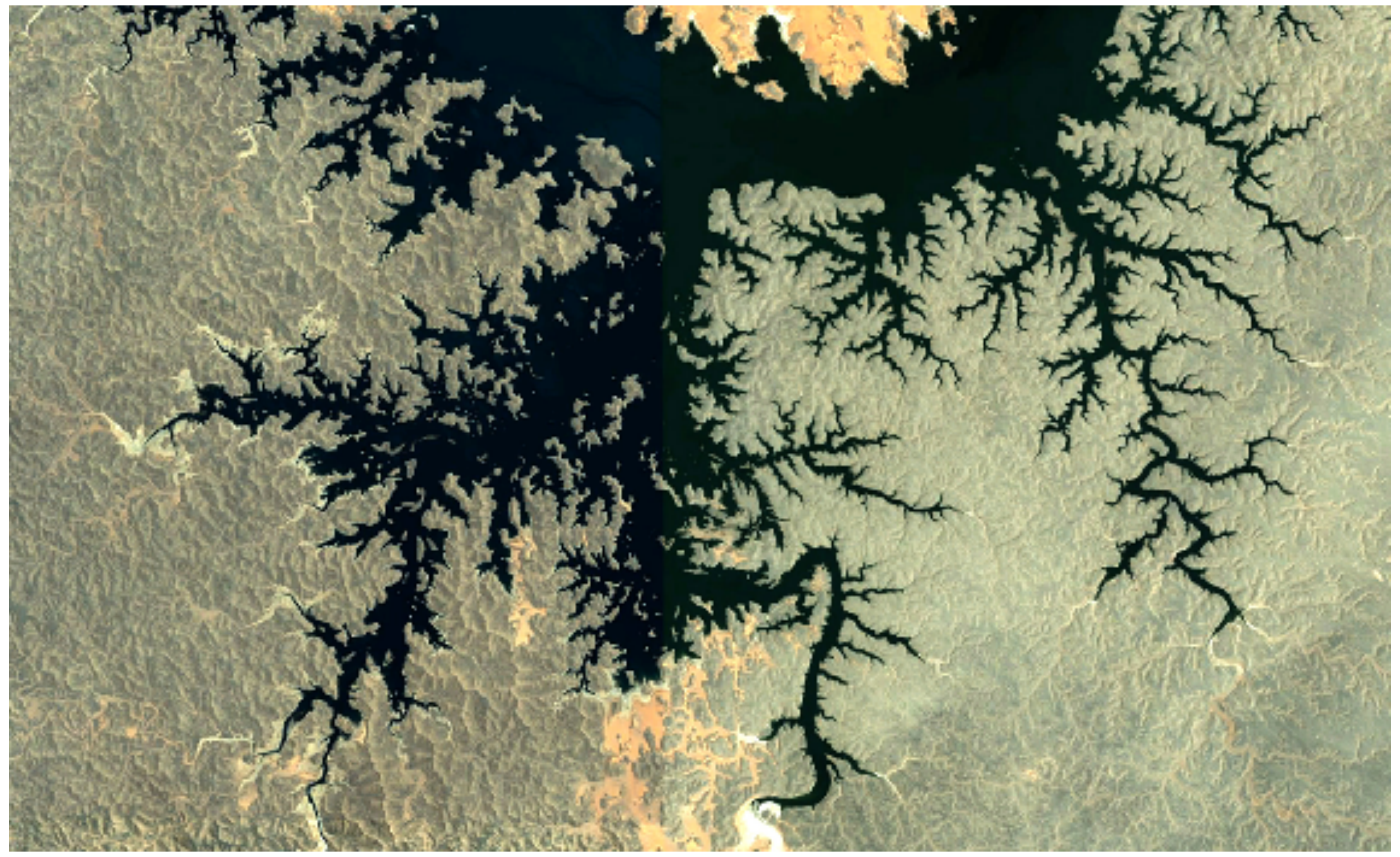
- If Science of Deep Learning is a map, how exact should it be?



Toy Models



Heuristics

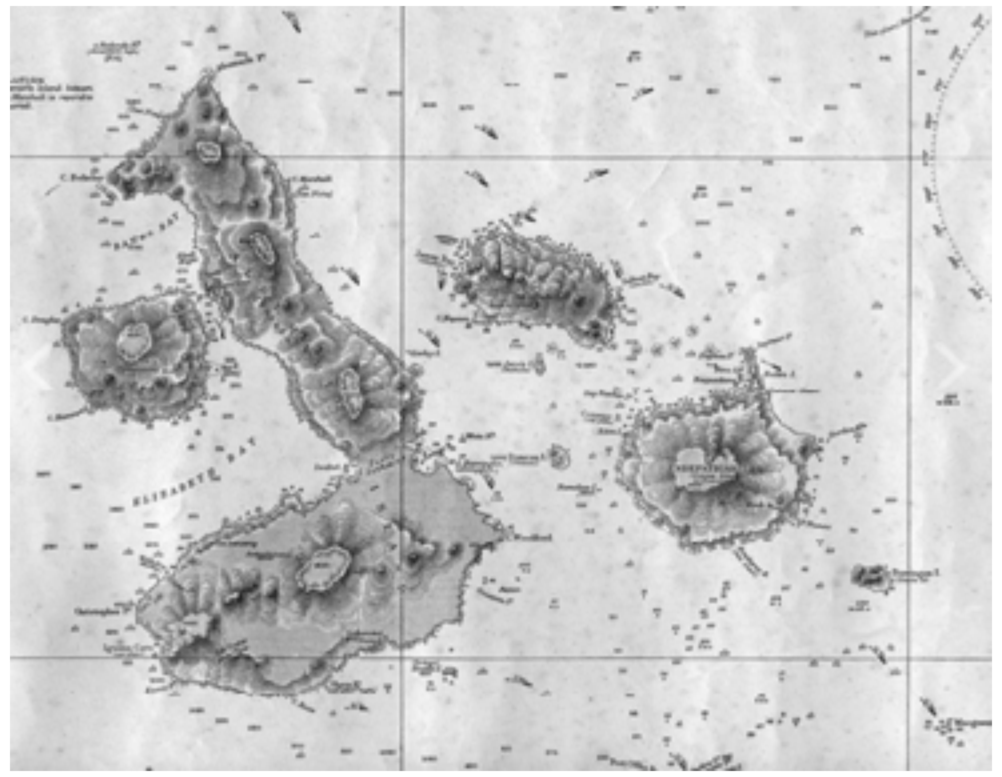


Scaling

On Rigor in Science (Borges)

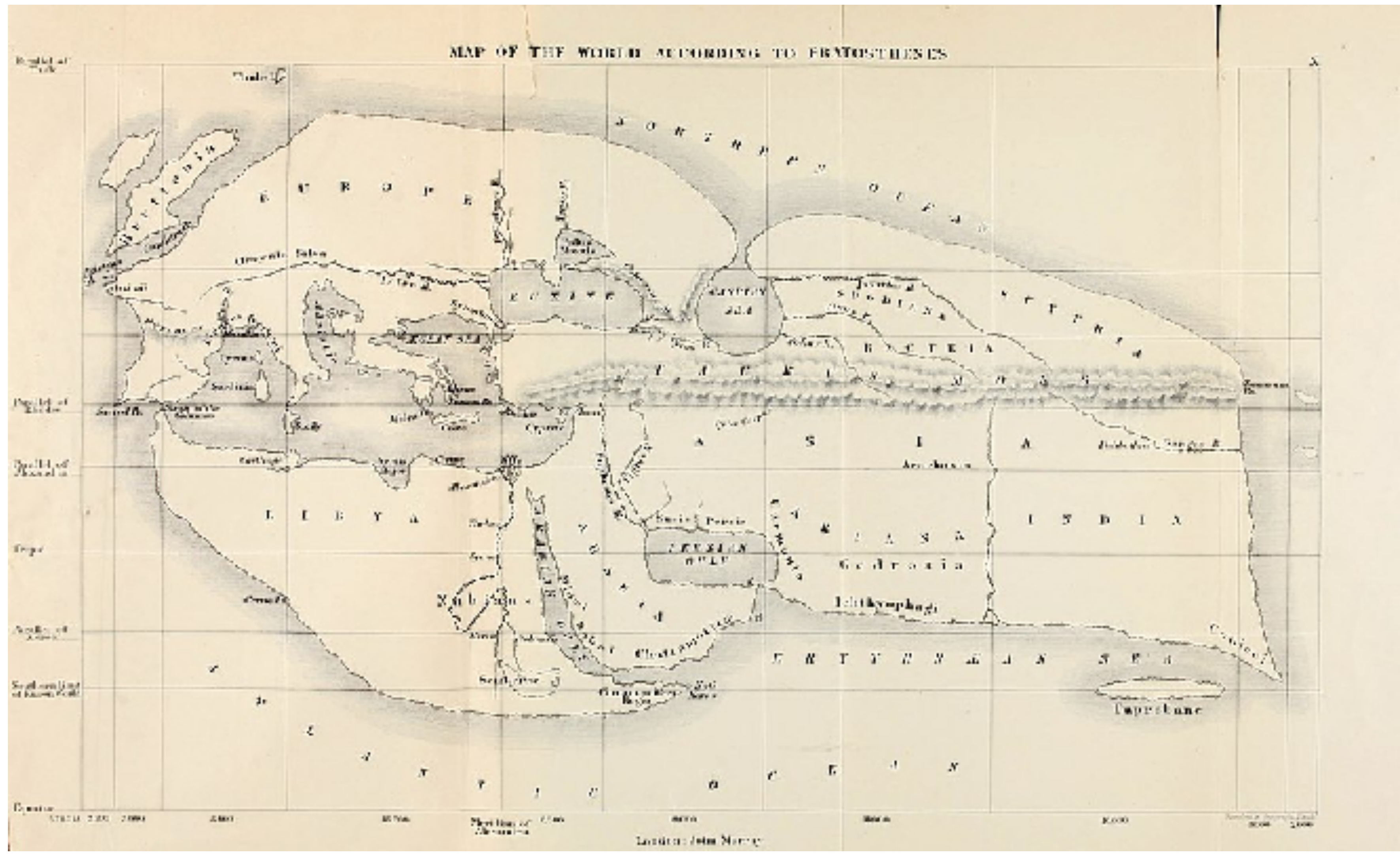
Inspired a talk by Solla's at KITP

- If Science of Deep Learning is a map, how exact should it be?

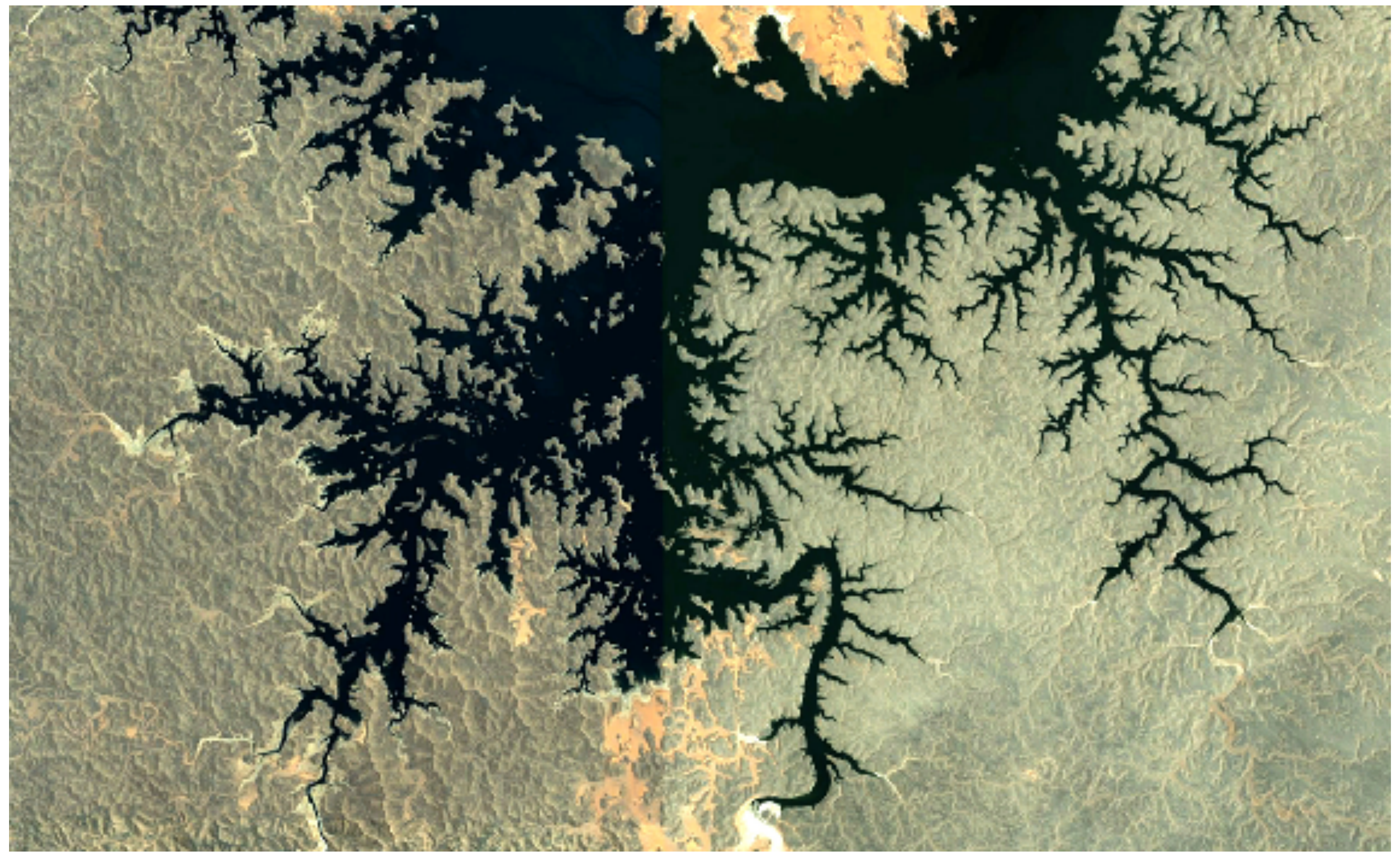


Toy Models

Kernel Adaptation + Generalization + Sample-Complexity-Change in 2+ trainable layer networks



Heuristics



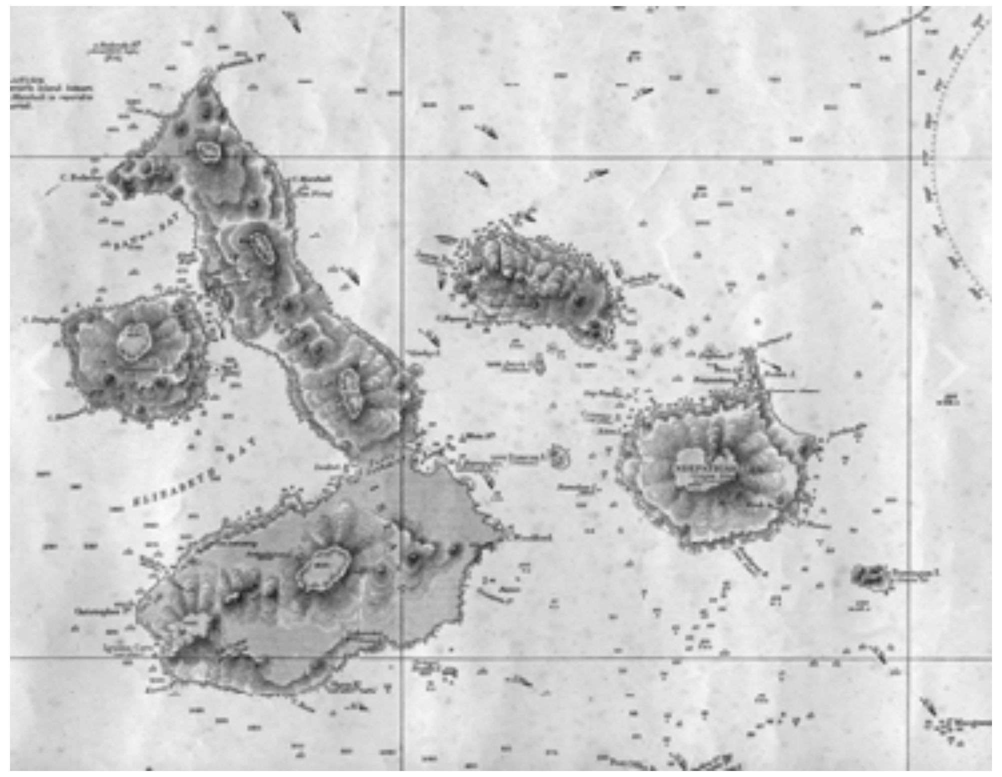
Scaling

Ringel et. al. Applications of Statistical Field Theory in Deep Learning (2025)

On Rigor in Science (Borges)

Inspired a talk by Solla's at KITP

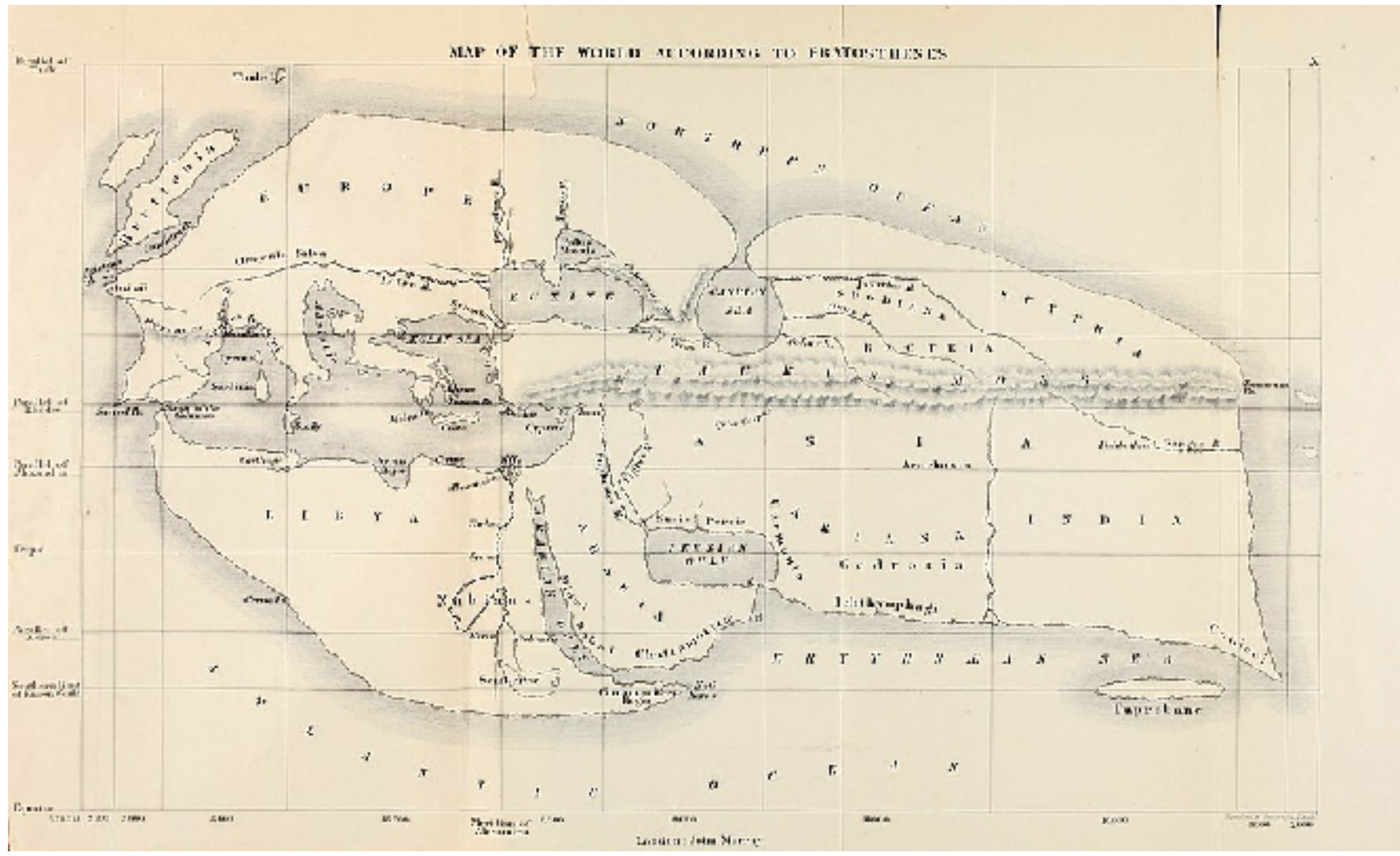
- If Science of Deep Learning is a map, how exact should it be?



Toy Models

Kernel Adaptation + Generalization + Sample-Complexity-Change in 2+ trainable layer networks

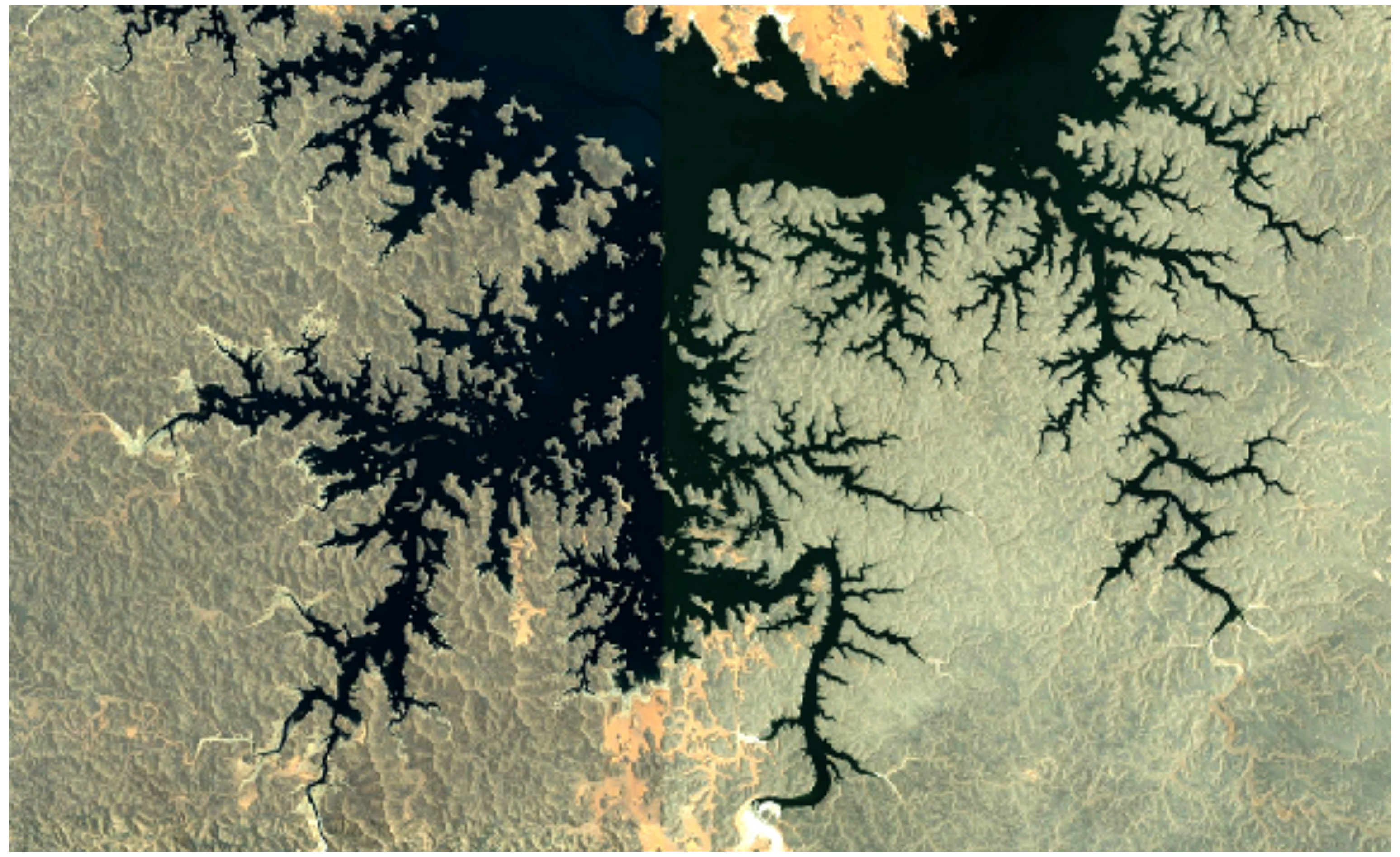
Ringel et. al. Applications of Statistical Field Theory in Deep Learning (2025)



Heuristics

How to guess sample complexity and feature learning patterns in deeper networks

Rubin et. al. Patterns of Feature Learning and Their Sample Complexity (TBP)

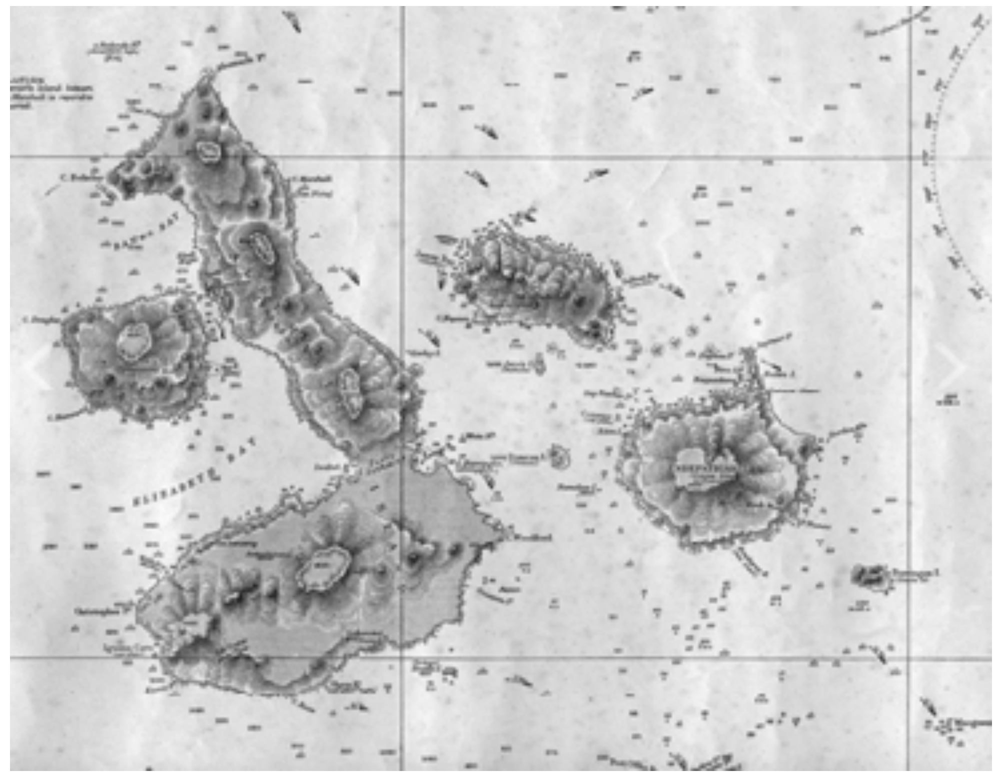


Scaling

On Rigor in Science (Borges)

Inspired a talk by Solla's at KITP

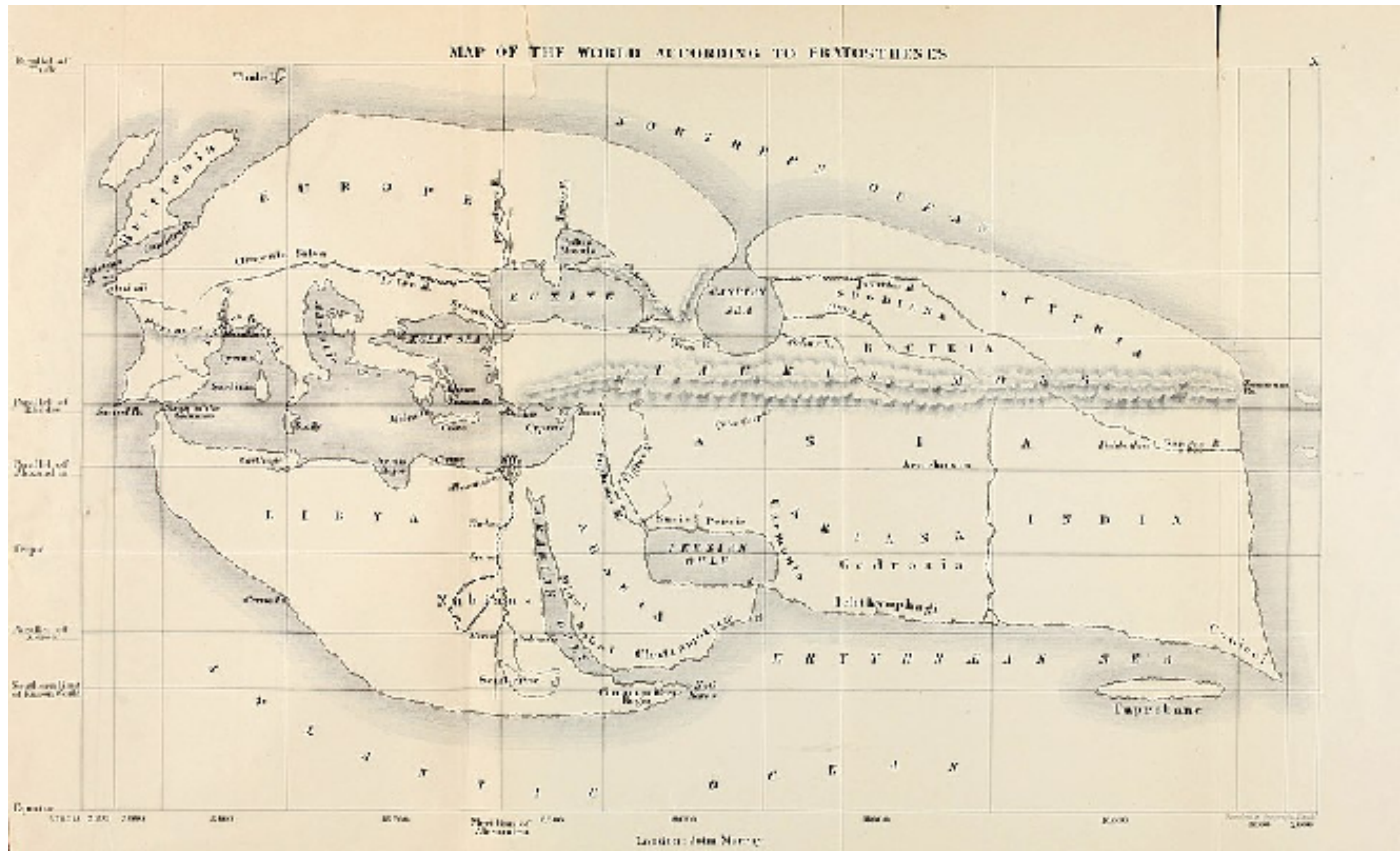
- If Science of Deep Learning is a map, how exact should it be?



Toy Models

Kernel Adaptation + Generalization + Sample-Complexity-Change in 2+ trainable layer networks

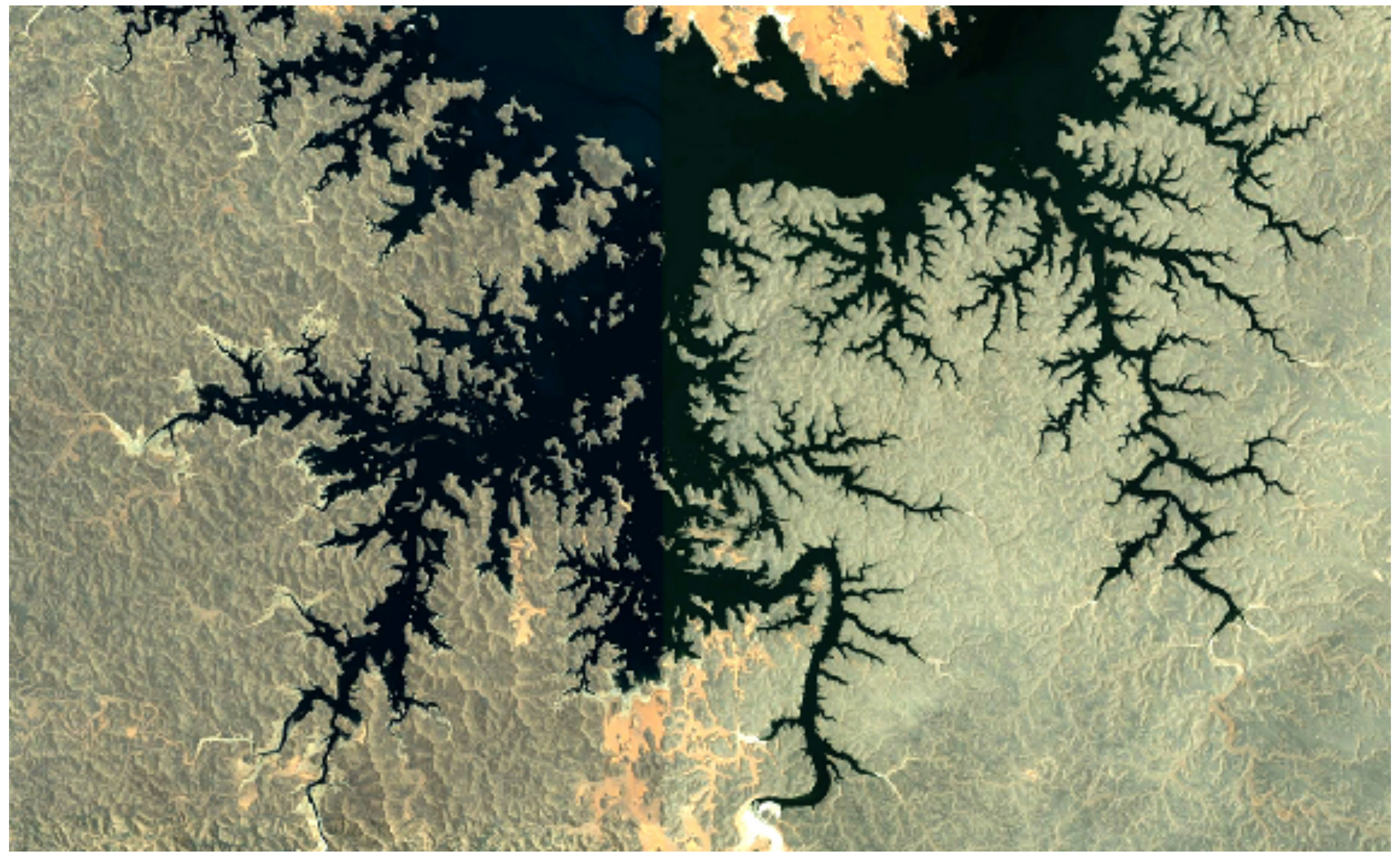
Ringel et. al. Applications of Statistical Field Theory in Deep Learning (2025)



Heuristics

How to guess sample complexity and feature learning patterns in deeper networks

Rubin et. al. Patterns of Feature Learning and Their Sample Complexity (TBP)



Scaling

Renormalization Group Analysis of Feature Learning Effects of power law distributed data

Howard et. al. Wilsonian RG of NNGPs (2025)

Gorka et. al. RG flows, Universality and Irrelevance in Overparametrized Deep Neural Networks (TBP)

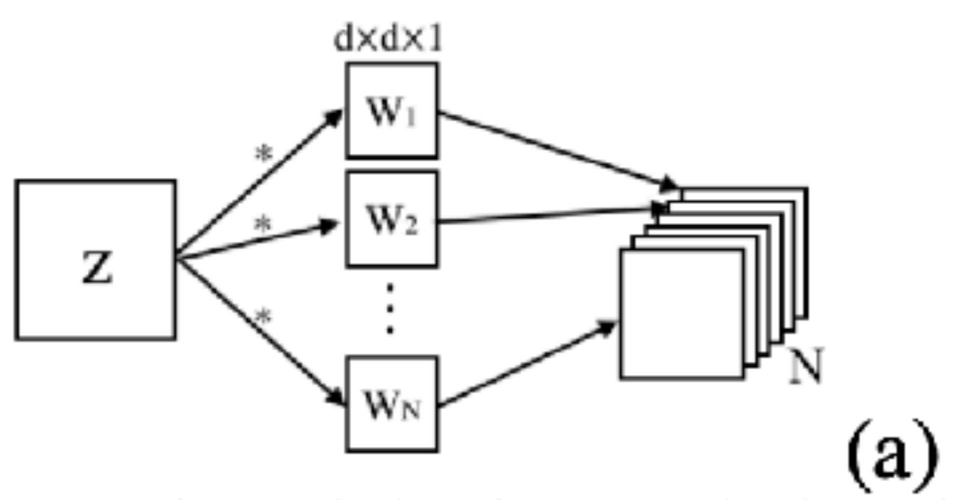
Theoretical Questions

- What is the sample complexity for various stylized tasks.
- What are the internal representation generate by a neural network, what is their implicit bias?
- How much learning is happening through Gaussian Processes like interpolations and how much through circuits/algorithmic-toolkits?
- How to predict the scaling behaviors of network performance? Does scaling imply universality?

Internal Representations in the Wild

- Network Compression/Pruning: Reducing weight to keep pre-activation PCA
- Sub-networks, Circuits
- Mechanistic Interpretability: Mono-semantic and Poly-semantic features

4 JADERBERG, VEDALDI, AND ZISSERMAN: SPEEDING UP CONVOLUTIONAL ...



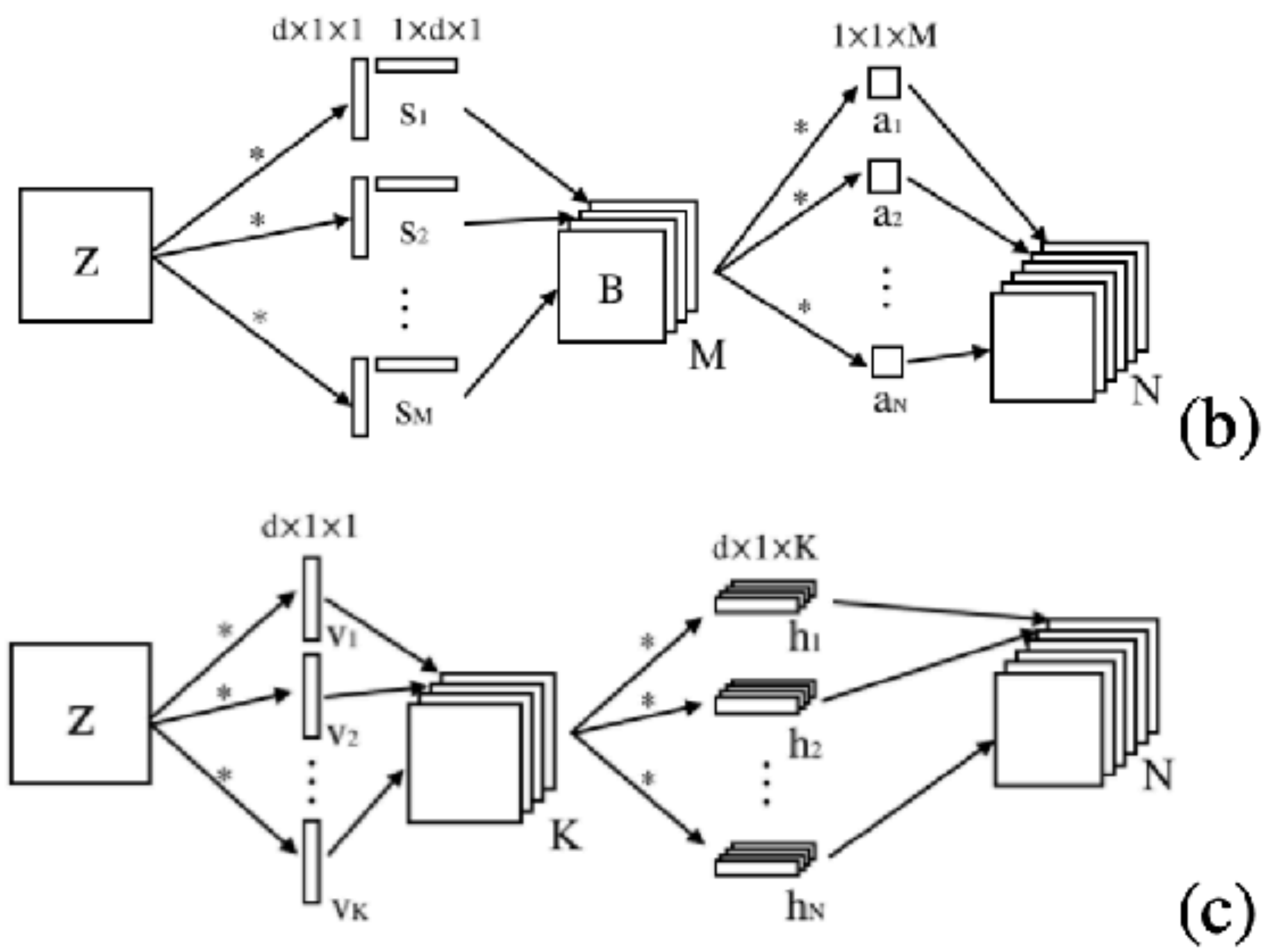


Figure 1: (a) The original convolutional layer acting on a single-channel input *i.e.* C=1. (b) The approximation to that layer using the method of Scheme 1. (c) The approximation to that layer using the method of Scheme 2. Individual filter dimensions are given above the filter layers.

Internal Representations in the Wild

- Network Compression/Pruning: Reducing weight to keep pre-activation PCA
- Sub-networks, Circuits
- Mechanistic Interpretability: Mono-semantic and Poly-semantic features

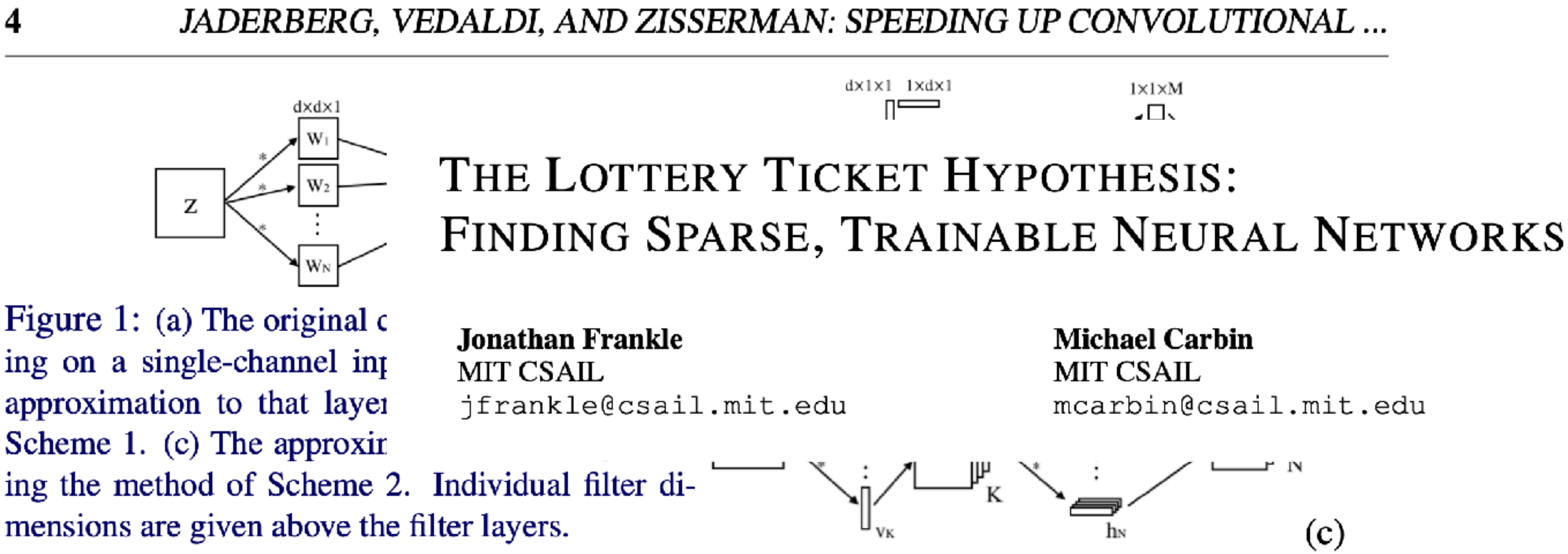

4 JADERBERG, VEDALDI, AND ZISSERMAN: SPEEDING UP CONVOLUTIONAL ...

Figure 1: (a) The original c
ing on a single-channel inp
approximation to that layer
Scheme 1. (c) The approxir
ing the method of Scheme 2. Individual filter dimensions are given above the filter layers.

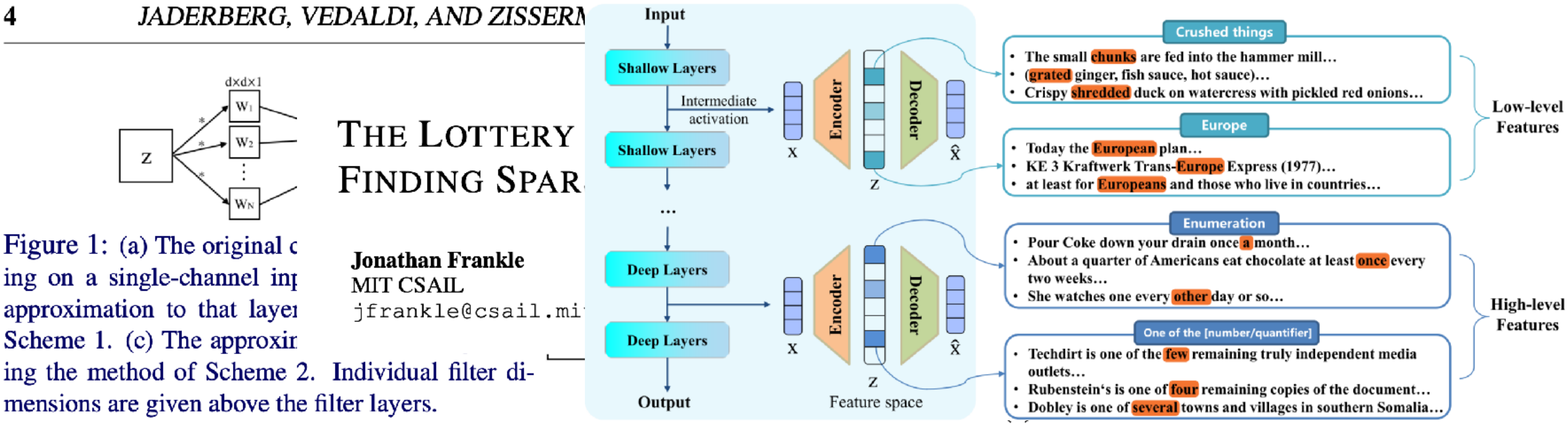
THE LOTTERY TICKET HYPOTHESIS: FINDING SPARSE, TRAINABLE NEURAL NETWORKS

Jonathan Frankle
MIT CSAIL
jfrankle@csail.mit.edu

Michael Carbin
MIT CSAIL
mcarbin@csail.mit.edu

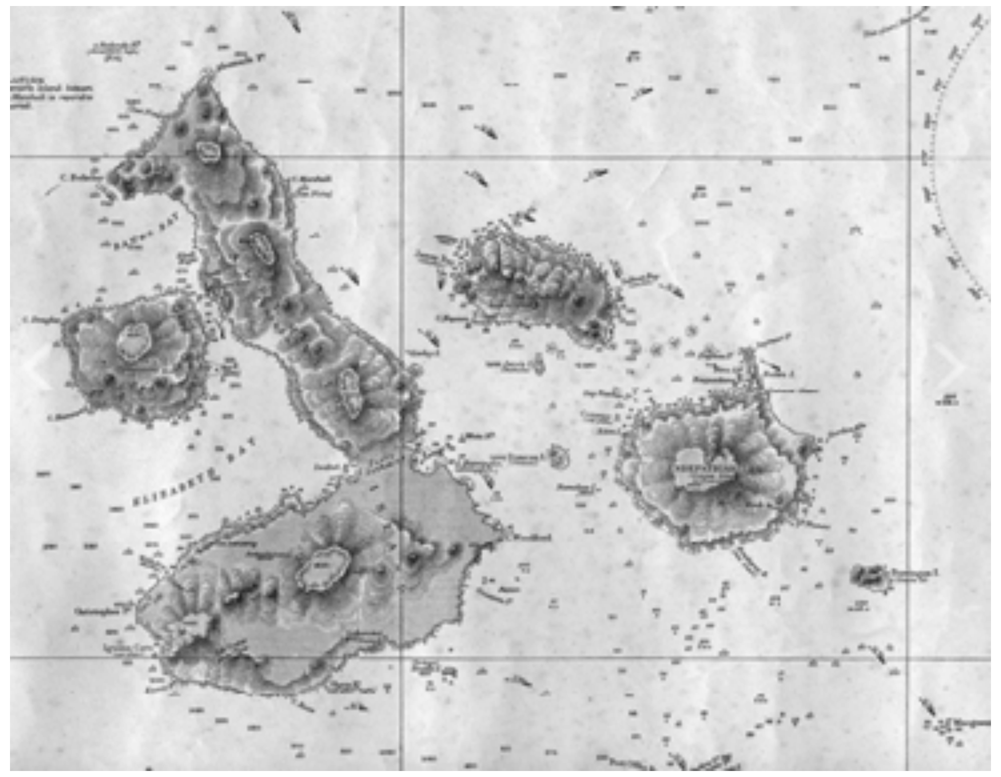
Internal Representations in the Wild

- Network Compression/Pruning: Reducing weight to keep pre-activation PCA
- Sub-networks, Circuits
- Mechanistic Interpretability: Mono-semantic and Poly-semantic features



Theoretical Approach for Rich Learning

- Saad and Sola like approaches
- Sequence multi-index model + ERM
- Deep Linear Networks
- Kernel-Scaling
- Kernel-Adaptation (Bayesian)
- DMFT for deep networks
- Rainbow networks



Kernel Adaptation and its Variants

Kernel Adaptation

- Statistical mechanics works by re-casting partition functions in terms of order-parameters and treating those using mean-field/saddle-point
- Kernel Adaptation uses pre-activation covariance matrix as order-parameters (as well as discrepancies in predictions).
- For a touch of the algebra, here is an exact Bayesian action with order-parameters marked

Seroussi, Naveh, Ringel (2021)

Kernel Adaptation

- Statistical mechanics works by re-casting partition functions in terms of order-parameters and treating those using mean-field/saddle-point
- Kernel Adaptation uses pre-activation covariance matrix as order-parameters (as well as discrepancies in predictions).
- For a touch of the algebra, here is an exact Bayesian action with order-parameters marked

$$S_{2layers} = \sum_c \frac{d|w_c|^2}{2\sigma_w^2} + \frac{\sigma_a^2}{N}\int d\mu_x d\mu_{x'} \tilde{f}(x) \sum_{c=1}^{N} \phi(w_c^T x)\phi(w_c^T x')\tilde{f}(x') + i\int d\mu_x \tilde{f}f - P\int d\mu_x e^{-\frac{[f(x)-y(x)]^2}{T}}$$

Seroussi, Naveh, Ringel (2021)

Kernel Adaptation

- Statistical mechanics works by re-casting partition functions in terms of order-parameters and treating those using mean-field/saddle-point
- Kernel Adaptation uses pre-activation covariance matrix as order-parameters (as well as discrepancies in predictions).
- For a touch of the algebra, here is an exact Bayesian action with order-parameters marked

$$S_{2layers} = \sum_c \frac{d|w_c|^2}{2\sigma_w^2} + \frac{\sigma_a^2}{N}\int d\mu_x d\mu_{x'} \tilde{f}(x) \overbrace{\sum_{c=1}^{N} \phi(w_c^T x)\phi(w_c^T x')}^{K_{\langle \tilde{f}\tilde{f}\rangle}(x,x')} \tilde{f}(x') + i\int d\mu_x \tilde{f}f - P\int d\mu_x e^{-\frac{[f(x)-y(x)]^2}{T}}$$

Seroussi, Naveh, Ringel (2021)

Kernel Adaptation

- Statistical mechanics works by re-casting partition functions in terms of order-parameters and treating those using mean-field/saddle-point
- Kernel Adaptation uses pre-activation covariance matrix as order-parameters (as well as discrepancies in predictions).
- For a touch of the algebra, here is an exact Bayesian action with order-parameters marked

$$S_{2layers} = \sum_c \frac{d|w_c|^2}{2\sigma_w^2} + \frac{\sigma_a^2}{N}\int d\mu_x d\mu_{x'} \tilde{f}(x) \overbrace{\sum_{c=1}^{N} \phi(w_c^T x)\phi(w_c^T x')}^{K_{\langle \tilde{f}\tilde{f}\rangle}(x,x')} \tilde{f}(x') + i\int d\mu_x \tilde{f}f - P\int d\mu_x e^{-\frac{[f(x)-y(x)]^2}{T}}$$

$$S_{3layers} = \frac{d|w|^2}{2} - i\int d\mu_x \left[\tilde{f}(x)f(x) + \sum_i \tilde{h}_i(x)h_i(x)\right] + \frac{1}{2N^{(1)}}\sum_i \left(\int d\mu_x \tilde{f}(x)\sigma(h_i(x))\right)^2 + \frac{1}{2N^{(0)}}\sum_{ij}\left(\int d\mu_x \tilde{h}_i(x)\sigma(w_j^{(0)}\cdot x)\right)^2 - P\int d\mu_x e^{-[f(x)-y(x)]^2/T}$$

Seroussi, Naveh, Ringel (2021)

Kernel Adaptation - Layer-wise actions

- Following the Mean-Field decoupling - one gets layer-wise actions which are coupled via pre-activation covariance matrices

Seroussi, Naveh, Ringel (2021)

Kernel Adaptation - Layer-wise actions

- Following the Mean-Field decoupling - one gets layer-wise actions which are coupled via pre-activation covariance matrices

$$\pi[w_c, h_c, f] \approx \pi[w_c]\pi[h_c]\pi[f]$$

Seroussi, Naveh, Ringel (2021)

Kernel Adaptation - Layer-wise actions

- Following the Mean-Field decoupling - one gets layer-wise actions which are coupled via pre-activation covariance matrices

$$\pi[w_c, h_c, f] \approx \pi[w_c]\pi[h_c]\pi[f]$$

$$\pi[w_c, h_c, f] \approx \pi_{<hh>}[w_c]\pi_{\langle \phi(wx)\phi(wx)\rangle, \langle ff \rangle}[h_c] \ldots \pi_{\langle \phi(h)\phi(h)\rangle}[f]$$

Seroussi, Naveh, Ringel (2021)

Kernel Adaptation - Layer-wise actions

- Following the Mean-Field decoupling - one gets layer-wise actions which are coupled via pre-activation covariance matrices

$$\pi[w_c, h_c, f] \approx \pi[w_c]\pi[h_c]\pi[f]$$

$$\pi[w_c, h_c, f] \approx \pi_{<hh>}[w_c]\pi_{\langle\phi(wx)\phi(wx)\rangle,\langle ff\rangle}[h_c]\ldots\pi_{\langle\phi(h)\phi(h)\rangle}[f]$$

$$\uparrow$$

$$\prod_{c=1}^{N_2} e^{-\int\int h_c(x)\langle\phi(w\cdot x')\phi(w\cdot x)\rangle^{-1}h_c(x')+\frac{P^2}{T^2N_2}[\int d\mu_x\langle f(x)-y(x)\rangle\phi(h_c(x)]^2}$$

Seroussi, Naveh, Ringel (2021)

Kernel Adaptation - GFL and Specialization

$$f(x) = \sum_{c=1}^{N} a_c Erf(w_c \cdot x) \quad y(x) = w_* \cdot x + 0.1 He_3(w_* \cdot x)$$

Rubin, Seroussi, Ringel (ICLR 2023)

Kernel Adaptation - GFL and Specialization

$$f(x) = \sum_{c=1}^{N} a_c Erf(w_c \cdot x) \quad y(x) = w_* \cdot x + 0.1 He_3(w_* \cdot x)$$

$$\pi[w, f] \approx \pi[w]\pi[f]$$

Rubin, Seroussi, Ringel (ICLR 2023)

Kernel Adaptation - GFL and Specialization

$$f(x) = \sum_{c=1}^{N} a_c Erf(w_c \cdot x) \quad y(x) = w_* \cdot x + 0.1 He_3(w_* \cdot x)$$

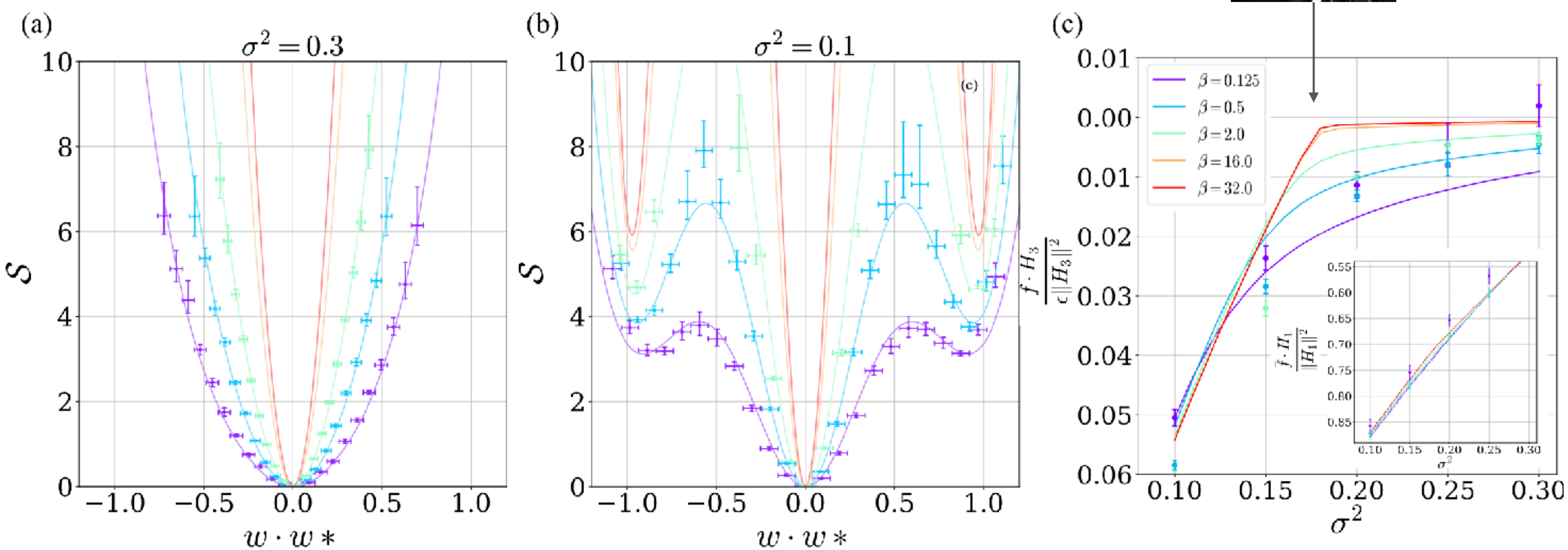
$$\pi[w, f] \approx \pi[w]\pi[f]$$

$$S = -\log\left(\int dw_\perp \pi[w \cdot w_*, w_\perp]\right)$$

Rubin, Seroussi, Ringel (ICLR 2023)

Kernel Adaptation - GFL and Specialization

GFL phase Specialization phase



$$\pi[w,f] \approx \pi[w]\pi[f]$$

$$S = -\log\left(\int dw_{\perp}\pi[w\cdot w_{*}, w_{\perp}]\right)$$

Similar transition for modular grokking

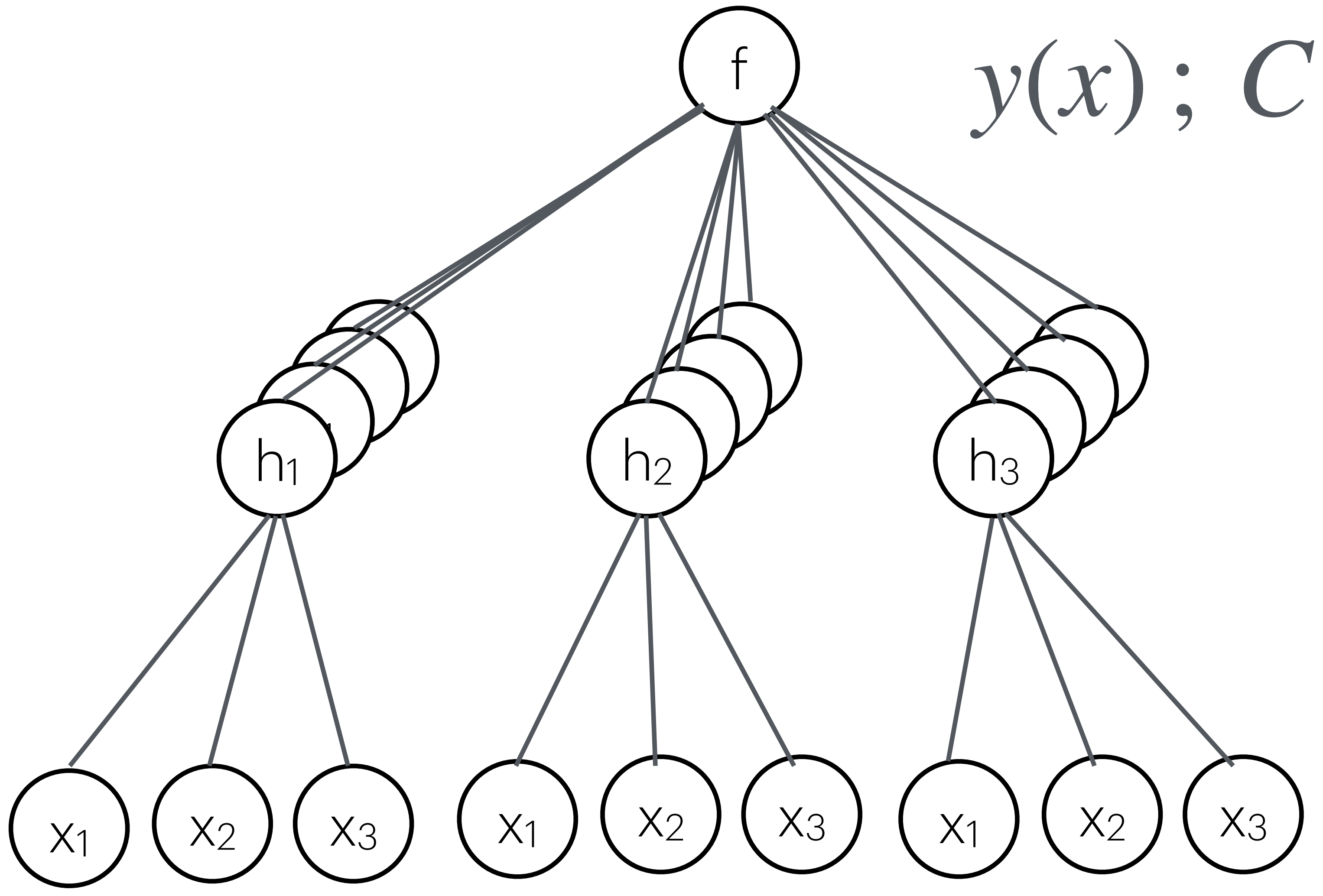
Rubin, Seroussi, Ringel (ICLR 2023)

Kernel Adaptation - Some New Results

- Small Ridge, Generalization, and Sample Complexity changes within GFL

$$f(x) = \sum_{i=1,c=1}^{\sqrt{d},C} a_{ic}\sigma\left(h_{i,c}(x)\right)$$

$$h_{i,c}(x) = w_c \cdot [x_{\sqrt{d}(i-1)}, \ldots, x_{\sqrt{d}i}]$$



$y(x)\,;\; C = 1$ student

Applications of Statistical Field Theory in Deep Learning

Zohar Ringel, Noa Rubin, Edo Mor, Moritz Helias, Inbar Seroussi

https://arxiv.org/abs/2502.18553

Kernel Adaptation - Some New Results

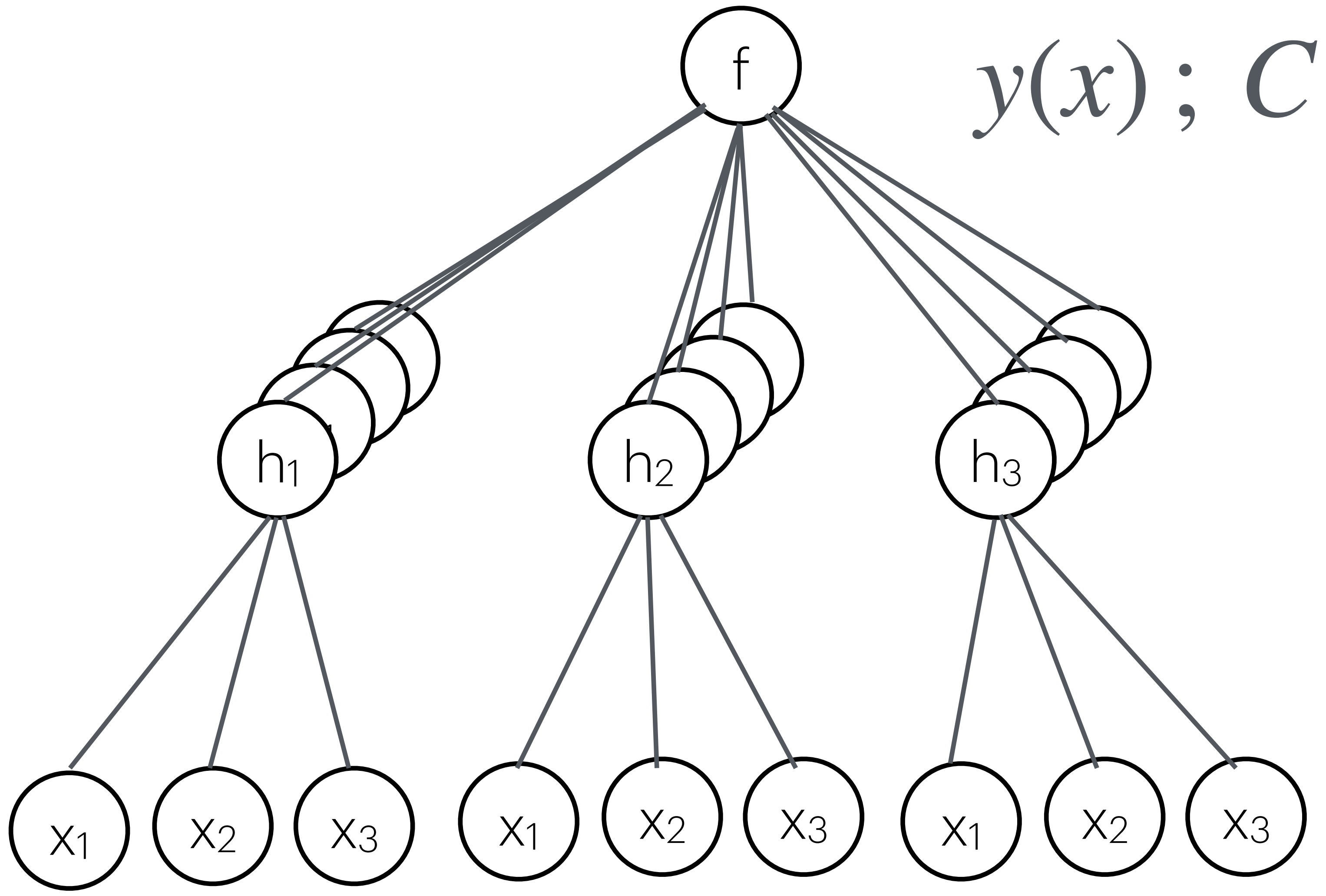
- Small Ridge, Generalization, and Sample Complexity changes within GFL

$$S_{2layers} = \sum_c \frac{d|w_c|^2}{2\sigma_w^2} + \frac{\sigma_a^2}{N}\int d\mu_x d\mu_{x'} \tilde{f}(x) \sum_{c=1}^{N} \phi(w_c^T x)\phi(w_c^T x')\tilde{f}(x') + i\int d\mu_x \tilde{f} f - P\int d\mu_x e^{-\frac{[f(x)-y(x)]^2}{T}}$$

$$f(x) = \sum_{i=1,c=1}^{\sqrt{d},C} a_{ic}\sigma\left(h_{i,c}(x)\right)$$

$$h_{i,c}(x) = w_c \cdot [x_{\sqrt{d}(i-1)}, \ldots, x_{\sqrt{d}i}]$$

$y(x)\,;\; C = 1$ student



Applications of Statistical Field Theory in Deep Learning

Zohar Ringel, Noa Rubin, Edo Mor, Moritz Helias, Inbar Seroussi

https://arxiv.org/abs/2502.18553

Kernel Adaptation - Some New Results

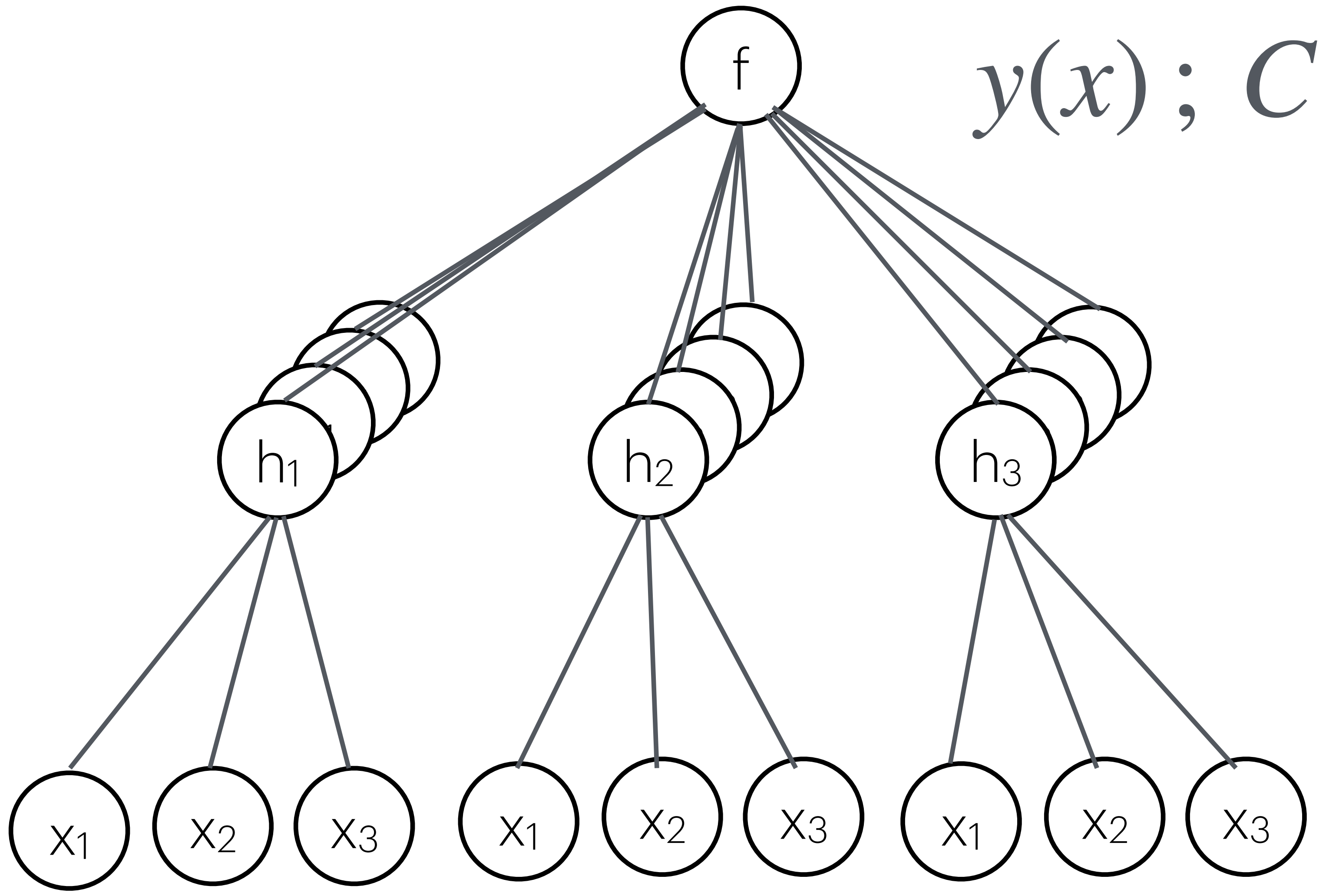
- Small Ridge, Generalization, and Sample Complexity changes within GFL

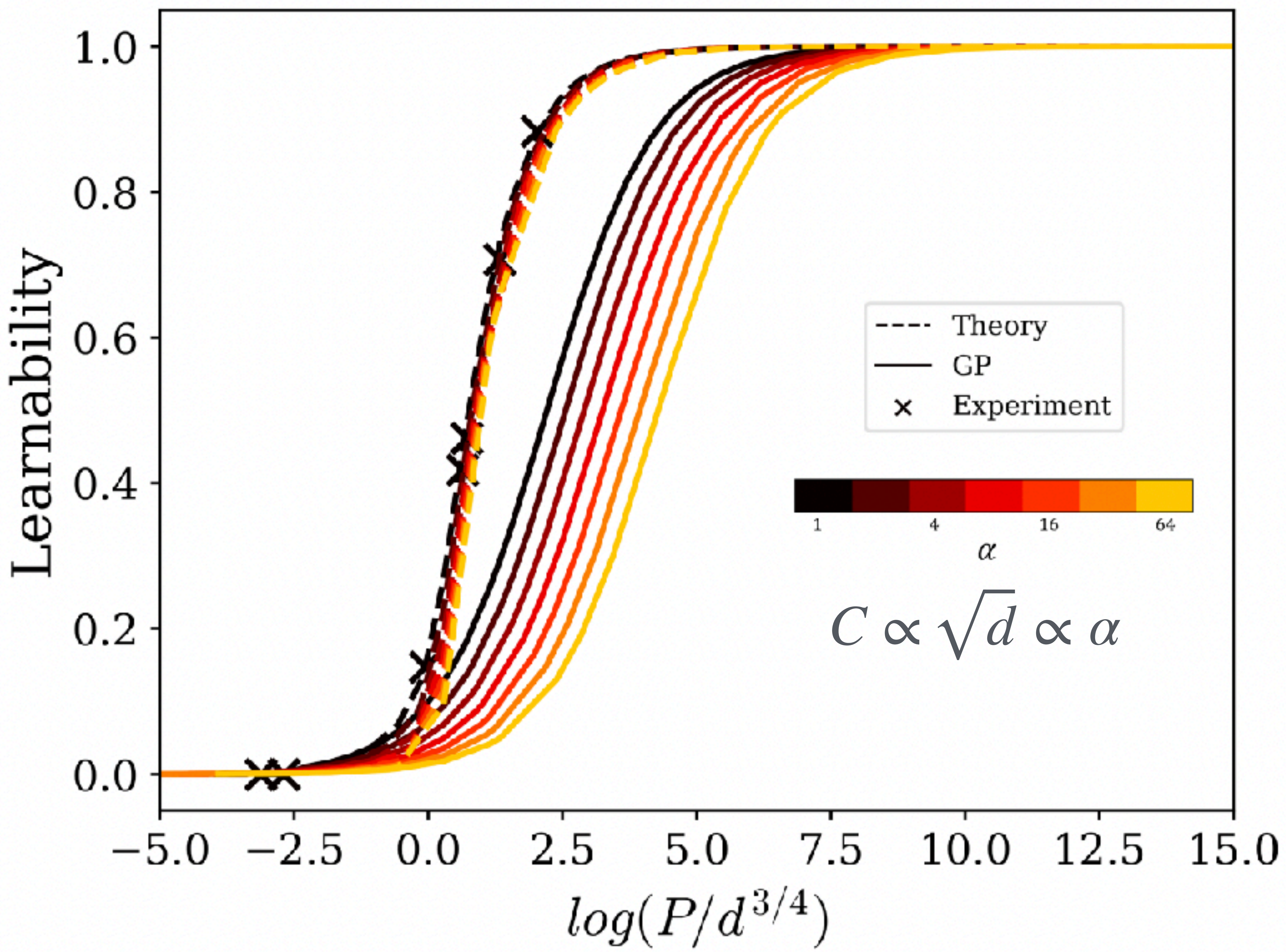
$$f(x) = \sum_{i=1,c=1}^{\sqrt{d},C} a_{ic}\sigma\left(h_{i,c}(x)\right)$$

$$S_{2layers} = \sum_c \frac{d|w_c|^2}{2\sigma_w^2} + \frac{\sigma_a^2}{N}\int d\mu_x d\mu_{x'}\tilde{f}(x)\sum_{c=1}^{N}\phi(w_c^T x)\phi(w_c^T x')\tilde{f}(x') + i\int d\mu_x \tilde{f}f - P\int d\mu_x e^{-\frac{[f(x)-y(x)]^2}{T}}$$

$$h_{i,c}(x) = w_c \cdot [x_{\sqrt{d}(i-1)},$$

$y(x)\,;\; C = 1$ stu





Applications of Statistical Field Theory in Deep Learn

Zohar Ringel, Noa Rubin, Edo Mor, Moritz Helias, Inbar Seroussi

https://arxiv.org/abs/2502.18553

Limitations

(Borges again)

- Dimension of non-linear equations grows as the number of kernel-eigenfunction components in the target
- Requires detailed knowledge of the input data-distribution.
- More than 3 trainable layers gets quite tedious (apart from linear or nearly linear activations)
- Non-quadratic losses require further approximations

Limitations

(Borges again)

- Dimension of non-linear equations grows as the number of kernel-eigenfunction components in the target
- Requires detailed knowledge of the input data-distribution.
- More than 3 trainable layers gets quite tedious (apart from linear or nearly linear activations)
- Non-quadratic losses require further approximations

Equations for GFL in 3-layer network

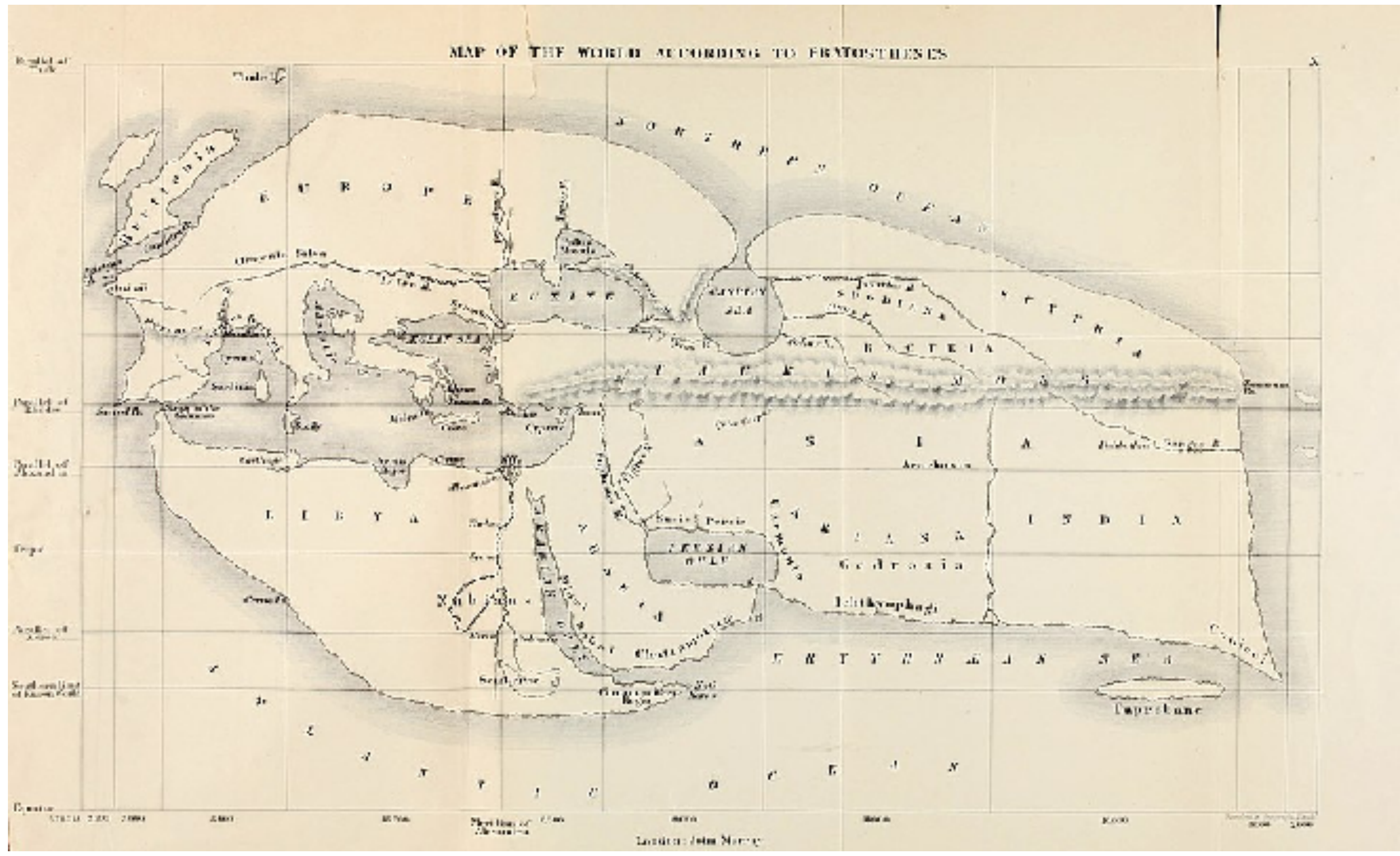
$$\bar{\mathbf{f}} = Q_f \left[Q_f + \sigma^2 I_n\right]^{-1} \mathbf{y}$$

$$\left[\left(K^{(L-1)}\right)^{-1}\right]_{\mu\nu} = \left[\left(Q^{(L-1)}\right)^{-1}\right]_{\mu\nu} - \frac{1}{N_{L-1}} \mathrm{Tr}\left\{A^{(L)} \frac{\partial Q_f}{\partial \left[K^{(L-1)}\right]_{\mu\nu}}\right\}$$

$$\left[\left[K^{(l-1)}\right]^{-1}\right]_{\mu\nu} = \left[\left[Q^{(l-1)}\right]^{-1}\right]_{\mu\nu} + \frac{2N_l}{N_{l-1}} \frac{\partial D_{\mathrm{KL}}(K^{(l)} || Q^{(l)})}{\partial \left[K^{(l-1)}\right]_{\mu\nu}} \text{ for all } l \in [2, L-1]$$

$$\left[\Sigma^{-1}\right]_{ss'} = \frac{d}{\sigma_1^2} \delta_{ss'} + \frac{2N_2}{N_1} \frac{\partial D_{\mathrm{KL}}(K^{(2)} || Q^{(2)})}{\partial \Sigma_{ss'}}$$

$$A^{(L)} = \sigma^{-4} (\mathbf{y} - \bar{\mathbf{f}})(\mathbf{y} - \bar{\mathbf{f}})^\top - \left[Q_f + \sigma^2 I_n\right]^{-1}$$



A Heuristic Approach to Sample Complexity and Feature Learning

N. Rubin, O. Davidovich, Z. Ringel ; Patterns in Feature Learning and Their Sample Complexity (TBP)

Alignment (A) as a control parameter instead of dataset size (P)

Sidelining overfitting effects which are often benign[1]— P and learnability can both be viewed as control parameters on feature learning.

Prior

Similar "posterior" for both.... $\pi_P[w^1,..,w^L] \propto P[w^1,..,w^L]\exp\left(\sum_{\mu=1}^{P}\frac{(f_\mu - y_\mu)^2}{2\kappa^2}\right)$

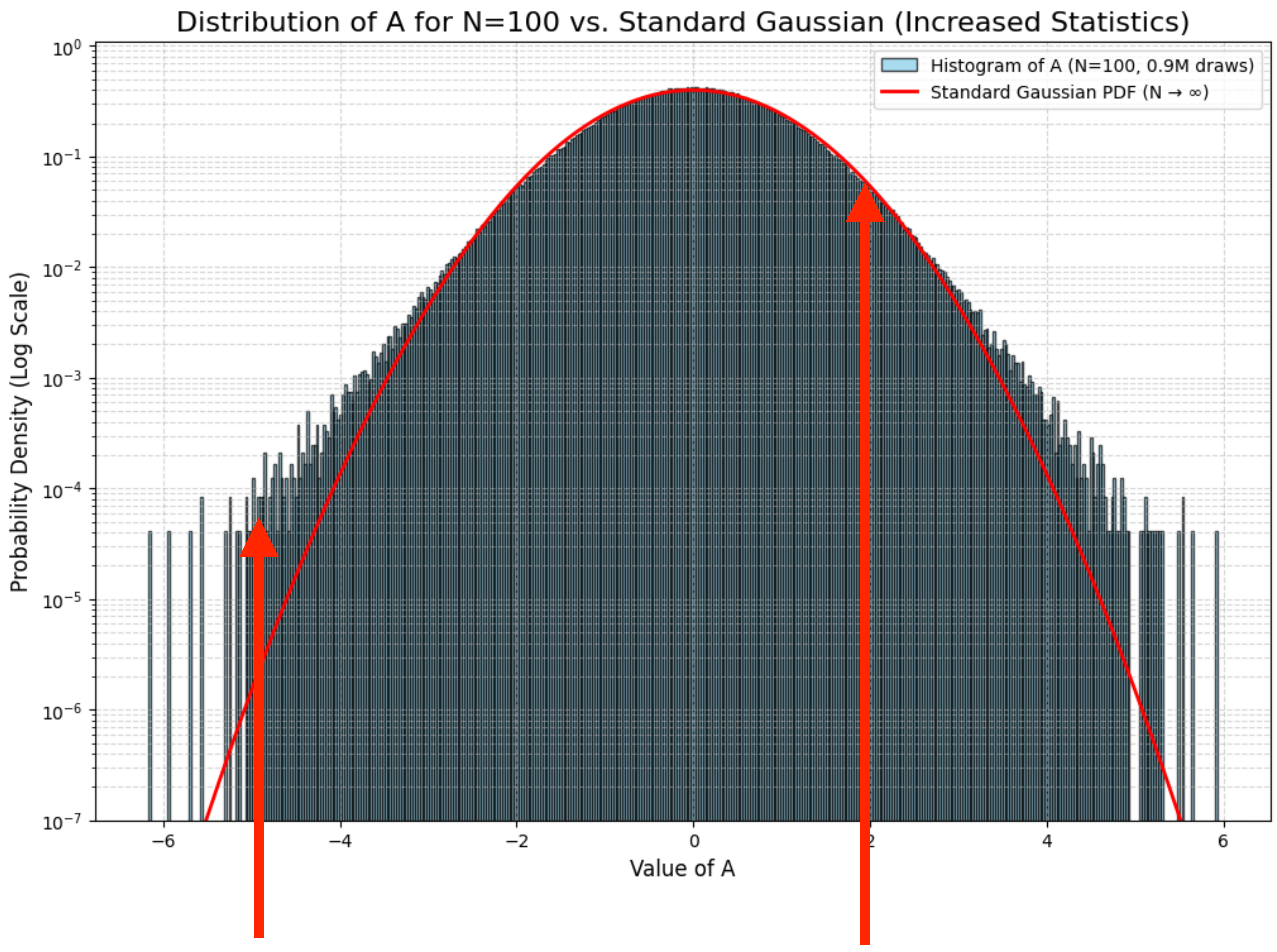
$$\pi_{A_P}[w^1,..,w^L] \propto P[w^1,..,w^L]\delta_\epsilon\left(\int d\mu_x f(x)y(x) - A_P\right)$$

Posteriors can be seen as skewing the prior towards rare events — **enter Large Deviation Theory**

1. Benign Overfitting in Linear Regression (2019); Canatar et. al. (2020);

Large Deviation Theory (LDT) 101

- A tool to analyze tails of a random variable typically written as sum of many (N) RV. Can also be seen as finite-N corrections to the Center Limit Theorem
- Consider $A = \frac{1}{\sqrt{N}} \sum_{c=1}^{N} \frac{a_c^3}{\sqrt{15}} \quad a_c \sim \mathcal{N}[0,1]$ which tends to $A \sim_{N\to\infty} \mathcal{N}[0,1]$
- However each (a_c^3) variable has an $\log(P(a_c^3 >> 1)) \propto a_c^{-2/3}$
- **LDT systemizes such computations via saddle-points and auxiliary tilt variables**



Feature learning as Large Deviation: a Toy Network Example

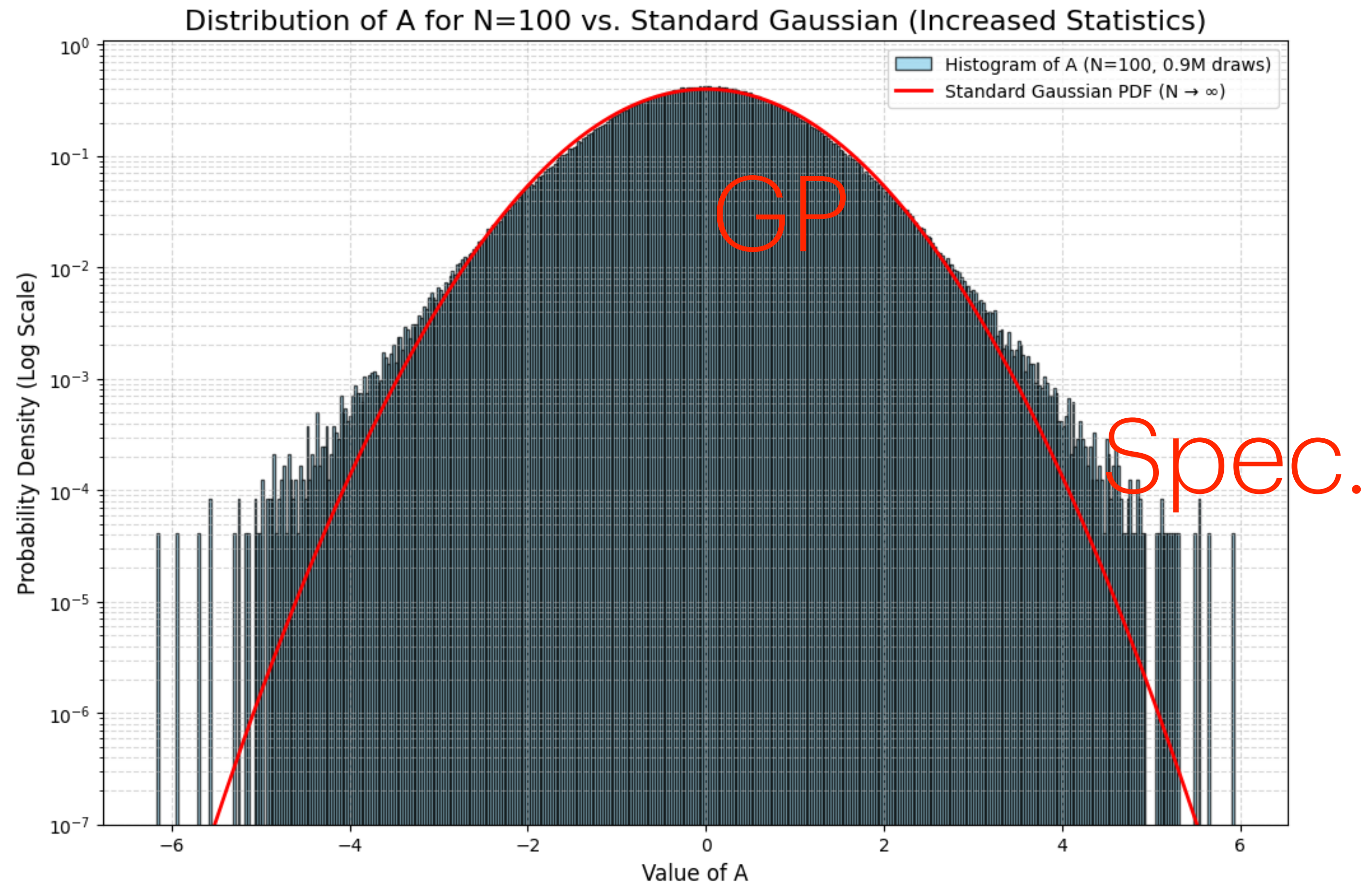
- Problem setup, the alignment integral

$$f(x) = \sum_{c=1}^{N} a_c Erf(w_c^T x) \quad x \in R^d, a_c \sim \mathcal{N}[0,N^{-1}], w_c \sim \mathcal{N}[0,d^{-1}] \quad y(x) = He_3(x_1)$$

$$A = \int d\mu_x f(x) He_3(x_1) = \sum_{i=1}^{N} a_i \frac{[w_i]_1^3}{(1 + 2[w_i]_1^2 + 2|\overrightarrow{w_i'}|^2)^{3/2}} \approx \sum_{i=1}^{N} a_i \frac{[w_i]_1^3}{(3 + 2[w_i]_1^2)^{3/2}}$$

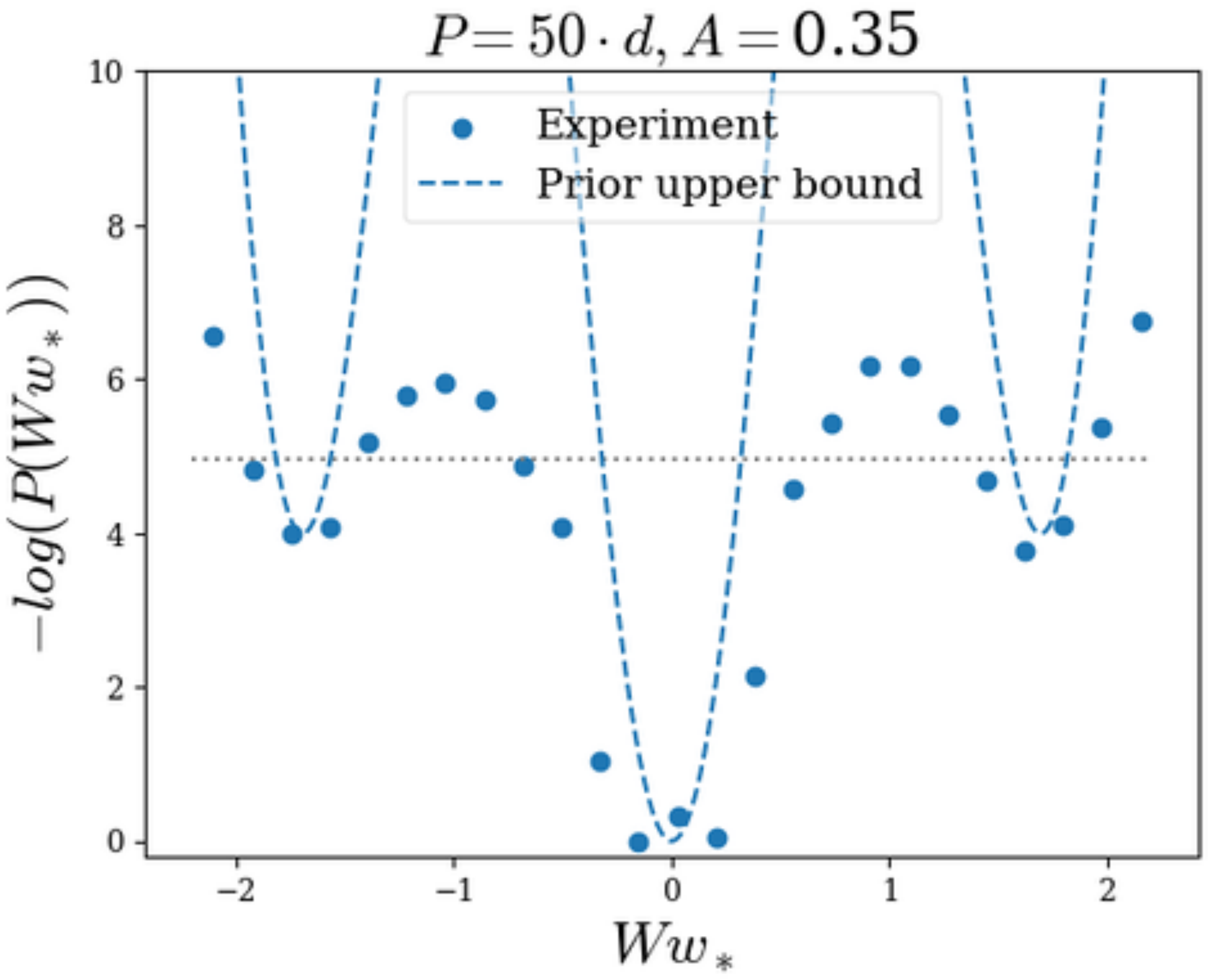
- We get a classical LDT problem: What rare-event/**Pattern** in a's and w_1's can generate an A=1 "event" ?

- The LDT equations turn out to be identical to Kernel Adaptation at large ridge



LDT weight configuration according for A compared to experiment

- Result from solving LDT/Kernel-Adaptation-at-large-ridge equations — an A=1 event is dominate by a **pattern** of O(1) specializing neurons



$$A \approx \sum_{i=1}^{N} a_i \frac{[w_i]_1^3}{(3 + 2[w_i]_1^2)^{3/2}}$$

Re-driving specialization result from heuristic

- Recall that $P(w,a) \propto e^{-d\sum_{i=1}^{N}|w_c|^2 - N\sum_{i=1}^{N}a_i^2}$ $\quad N \propto d \quad$ $A \approx \sum_{i=1}^{N} a_i \frac{[w_i]_1^3}{(3+2[w_i]_1^2)^{3/2}}$
- For $y(x) = He_3(x_1)$, what is the most likely weight configuration which gives $A \approx 1$?
  - **Pattern I** - O(1) w's specialize, O(1) a's specialize on the specialized w's

$$-\log(P(A \approx 1) \propto -\log\left(\frac{P(w_{SP}, a_{SP})}{P(w_{typ}, a_{typ})}\right) \propto d + N$$

Re-driving specialization result from heuristic

- Recall that $P(w,a) \propto e^{-d\sum_{i=1}^{N}|w_c|^2 - N\sum_{i=1}^{N} a_i^2}$ $\quad N \propto d \quad$ $A \approx \sum_{i=1}^{N} a_i \frac{[w_i]_1^3}{(3+2[w_i]_1^2)^{3/2}}$
- For $y(x) = He_3(x_1)$, what is the most likely weight configuration which gives $A \approx 1$?
  - **Pattern I** - O(1) w's specialize, O(1) a's specialize on the specialized w's

    $$-\log(P(A \approx 1) \propto -\log\left(\frac{P(w_{SP}, a_{SP})}{P(w_{typ}, a_{typ})}\right) \propto d + N$$

  - **Pattern II** - w's remain GP like, a's perform GPR on Random Features generate by w's $-\log\left(\frac{P(w_{typ}, a_{GP})}{P(w_{typ}, a_{typ})}\right) \propto d^3$

    (Alternatively $a_i \propto Sign(w_i)[O(w_i^3)]^{-1}/N \propto d^{1.5}/N$)

  - **Pattern III** - all w's inflate their variance along $\hat{x}_1$ by β, a's do a GP the random features generated by those w's

    $$-\log\left(\frac{P(w_{GFL}, a_{GP})}{P(w_{typ}, a_{typ})}\right) \propto N\beta + \left(\frac{d}{\beta}\right)^3 \Rightarrow_{optimize\ \ \beta} \propto (Nd)^{3/4}$$

Re-driving specialization result from heuristic

- Recall that $P(w,a) \propto e^{-d\sum_{i=1}^{N}|w_c|^2 - N\sum_{i=1}^{N} a_i^2}$ $\quad N \propto d \quad$ $A \approx \sum_{i=1}^{N} a_i \frac{[w_i]_1^3}{(3+2[w_i]_1^2)^{3/2}}$
- For $y(x) = He_3(x_1)$, what is the most likely weight configuration which gives $A \approx 1$?
  - **Pattern I** - O(1) w's specialize, O(1) a's specialize on the specialized w's

$$-\log(P(A \approx 1) \propto -\log\left(\frac{P(w_{SP}, a_{SP})}{P(w_{typ}, a_{typ})}\right) \propto d + N$$

Winner!

  - **Pattern II** - w's remain GP like, a's perform GPR on Random Features generate by w's $-\log\left(\frac{P(w_{typ}, a_{GP})}{P(w_{typ}, a_{typ})}\right) \propto d^3$

(Alternatively $a_i \propto Sign(w_i)[O(w_i^3)]^{-1}/N \propto d^{1.5}/N$)

  - **Pattern III** - all w's inflate their variance along $\hat{x}_1$ by β, a's do a GP the random features generated by those w's

$$-\log\left(\frac{P(w_{GFL}, a_{GP})}{P(w_{typ}, a_{typ})}\right) \propto N\beta + \left(\frac{d}{\beta}\right)^3 \Rightarrow_{optimize\ \beta} \propto (Nd)^{3/4}$$

From P(A) to sample complexity

Can we related the chance of a rare-$\boldsymbol{A}$-event in the prior to dataset-size?

$$-\log(P(A \approx 1) \propto -\log\left(\frac{P(w_{SP}, a_{SP})}{P(w_{typ}, a_{typ})}\right) \propto d + N$$

?

Number of samples ($\boldsymbol{P}$) required to reach $\boldsymbol{A}$

From P(A) to sample complexity

Can we related the chance of a rare-$\boldsymbol{A}$-event in the prior to dataset-size?

$$-\log(P(A \approx 1) \propto -\log\left(\frac{P(w_{SP}, a_{SP})}{P(w_{typ}, a_{typ})}\right) \propto d + N$$

? $\Longleftrightarrow$

Number of samples ($\boldsymbol{P}$) required to reach $\boldsymbol{A}$

- Unlearnability bound: $P > P_*$ is necessary to have learning where $P_* \propto -\log(P[A \approx 1])$
- One line derivation
- $$\pi(A \approx 1) = \frac{\int dwda\delta_\epsilon(\langle f|y\rangle_x - 1)P(w,a)e^{-\sum_{\mu=1}^{P} L(x_\mu)}}{\int dwdaP(w,a)e^{-\sum_{\mu=1}^{P} L(x_\mu)}} < \frac{\int dwda\delta_\epsilon(\langle f|y\rangle_x - 1)P(w,a)}{\int dwdaP(w,a)e^{-\sum_{\mu=1}^{P} L(x_\mu)}} <_{Jensen} = P(A \approx 1)e^{\sum_{\mu=1}^{P} \langle L(x_\mu)\rangle_{P(w,a)}}$$

Quick Ad: The Lazy Case - Data agnostic GP unlearnability bounds

$$\log(A_\lambda \approx 1) = \lambda^{-1} \quad A_\lambda = \int d\mu_x f(x)\phi_\lambda(x) \qquad \int d\mu_{x'} K(x,x')\phi_\lambda(x') = \phi_\lambda(x)$$

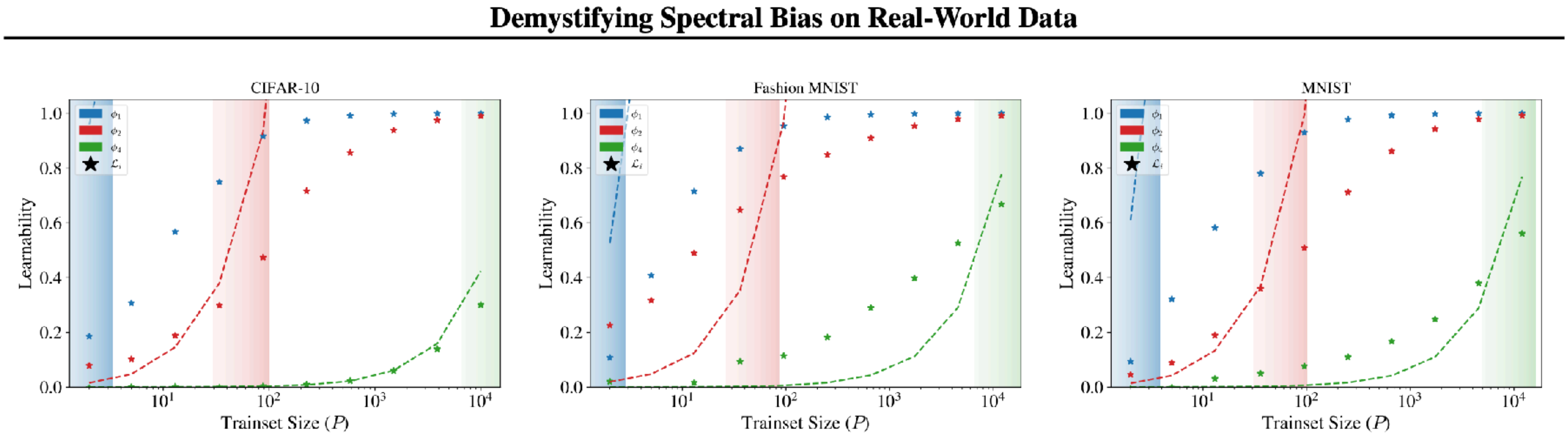


Figure 2: **(Theory predicts spectral bias on real-world datasets)** The (test) learnability (dots) together with the bound on the cross-dataset learnability bound in Eq. (10) (dashed). The shaded learning region indicated values of P given by the

- Colored Shaded Areas - Analytical predictions for around 65% learnability, matches where actual test learnability riches that regime.

Lavie, Ringel https://arxiv.org/abs/2406.02663

Summarizing Qualitative Lessons

Towards building a heuristic scaling argument

- $P_* \gtrsim -\log(P(A \approx 1)) \qquad A = \int d\mu_x f(x)\hat{y}(x) \qquad P(A) = \int dwdaP(w,a)\delta\left(\int d\mu_x f_{w,a}\hat{y} - A\right)$

- The networks weight arrangements according to the prior, conditioned on $A \approx 1$ are close to those in the posterior for P large enough to generate $A \approx 1$

- Feature Learning Pattern is given by the most likely weights which generate the unlikely $A \approx 1$;

- These often split layer-wise into few distinct patterns [GP,Specialization,GFL] which can be compared based on their log-prob.

- This most likely pattern can be translated into a bound/estimate on dataset size

- Almost agnostic to training set measure

Applying the Pattern Scaling Heuristic on several more examples

- Choose [GP,Specialization,GFL] for each layer
- Estimate layer-wise log prob. using excess-weight-decay/GP-on-random-features-of-previous-layer
- Sum those up to get tentative $\boldsymbol{P}_*$
- Optimize free-parameters
- Choose winning pattern
- Sample complexity scales as $\boldsymbol{P}_*$

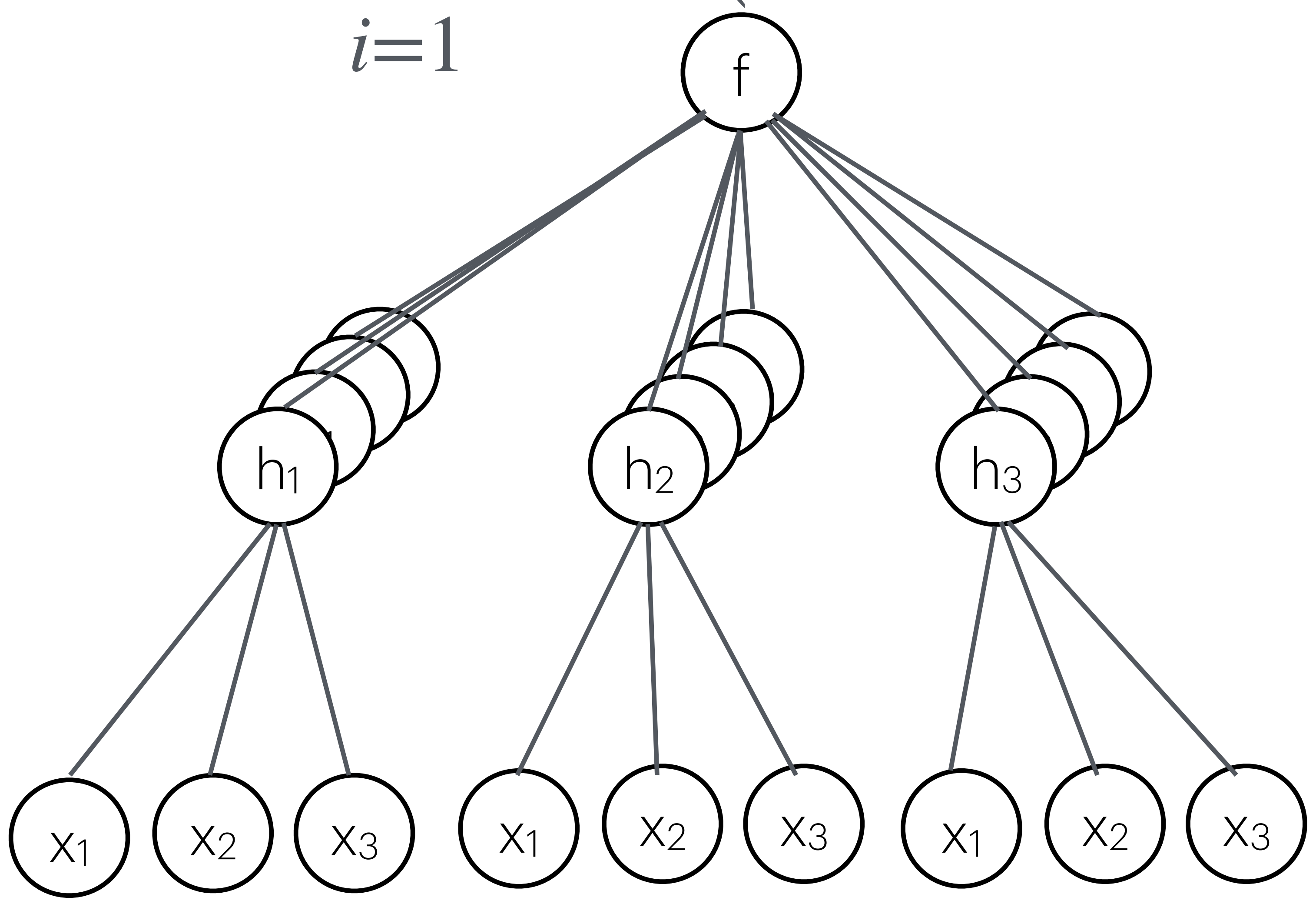
Recall again our CNN results

- Small Ridge, Generalization, and Sample Complexity changes within GFL

$$f(x) = \sum_{i=1,c=1}^{N,C} a_{ic}\sigma\left(h_{i,c}(x)\right) \quad N \propto S \propto C \propto \sqrt{d}$$

$$h_{i,c}(x) = w_c \cdot [x_{S(i-1)}, \ldots, x_{Si}]$$

$$y(x) = \sum_{i=1}^{N} a_i^* \sigma\left(w_* \cdot [x_{S(i-1)} \ldots X_{Si}]\right)$$



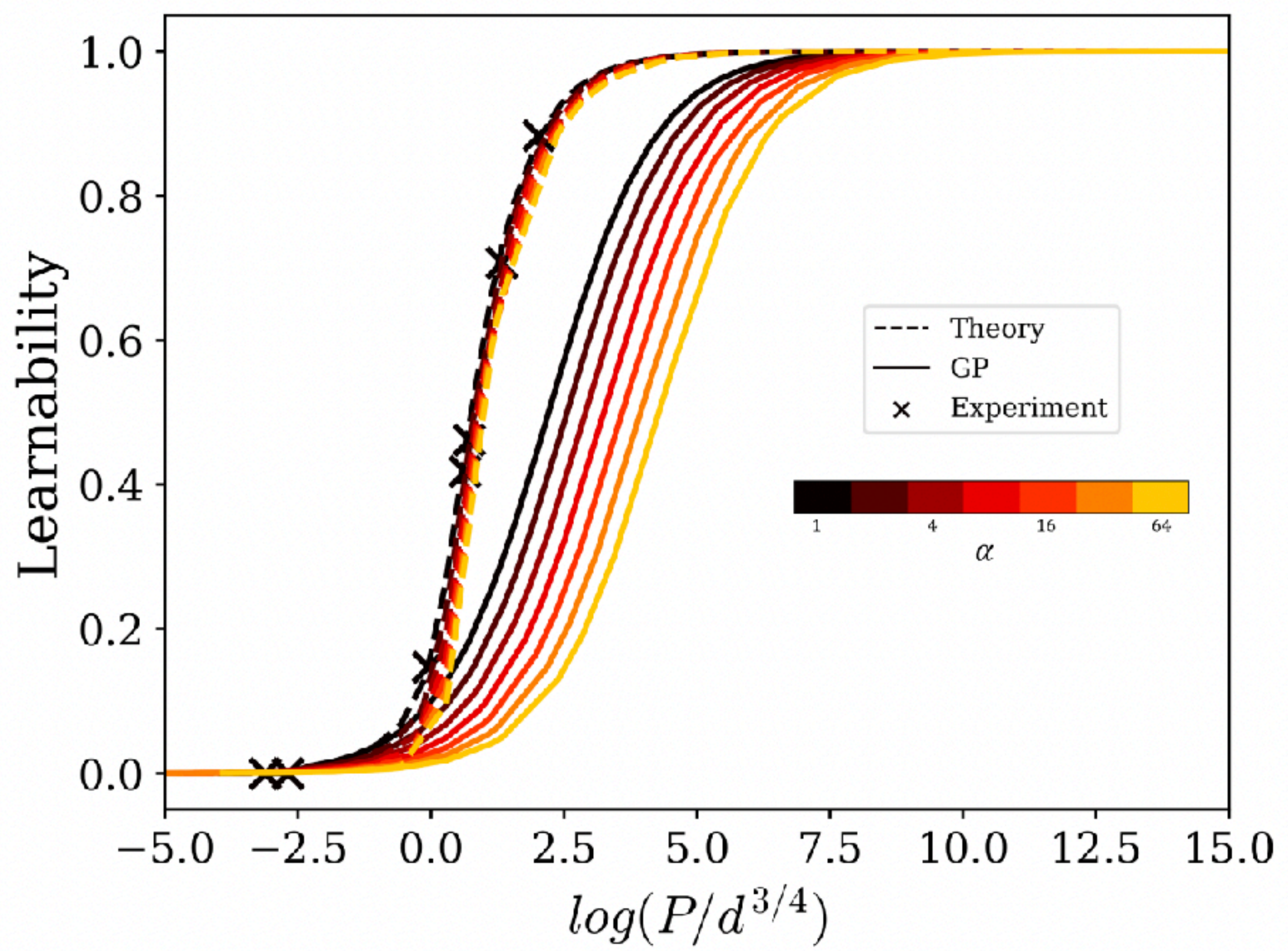


Figure 3.2: Learnability of linear CNNs as a function of P. We take $S, N, C \propto \alpha$, and consider different α scales of these parameters. Here the network is observed to learn the target at $P \propto d^{3/4}$, regardless of the parameter scale, as opposed to the GP predictions which predict learning at $P \propto d$. Parameters: $\chi = 100$, $N = 10\alpha, S = 50\alpha, C = 1000\alpha$.

Pattern scaling analysis

$$f(x) = \sum_{i=1,c=1}^{N,C} a_{ic}\sigma\left(h_{i,c}(x)\right) \quad N \propto S \propto C \propto \sqrt{d} \qquad y(x) = \sum_{i=1}^{N} a_i^* \sigma\left(w_* \cdot [x_{S(i-1)} \ldots X_{Si}]\right)$$

Pattern I - one a_{ic} and one w_c specialize to teacher

Pattern II - first layer increases its variance along w_* by β

Pattern scaling analysis

$$f(x) = \sum_{i=1,c=1}^{N,C} a_{ic}\sigma\left(h_{i,c}(x)\right) \quad N \propto S \propto C \propto \sqrt{d}$$

$$y(x) = \sum_{i=1}^{N} a_i^* \sigma\left(w_* \cdot [x_{S(i-1)} \ldots X_{Si}]\right)$$

$$P(w, a) \propto e^{-S|w|^2 - NC|a|^2}$$

Pattern I - one a_{ic} and one w_c specialize to teacher

Pattern II - first layer increases its variance along w_* by β

Pattern scaling analysis

$$f(x) = \sum_{i=1,c=1}^{N,C} a_{ic}\sigma\left(h_{i,c}(x)\right) \quad N \propto S \propto C \propto \sqrt{d} \qquad y(x) = \sum_{i=1}^{N} a_i^* \sigma\left(w_* \cdot [x_{S(i-1)} \dots X_{Si}]\right)$$

$$P(w, a) \propto e^{-S|w|^2 - NC|a|^2}$$

Pattern I - one a_{ic} and one w_c specialize to teacher

$$-\log\left(\frac{P(w_{SP}, a_{SP})}{P(w_{typ}, a_{typ})}\right) \propto S + NC \propto \sqrt{d}C \propto d$$

Pattern II - first layer increases its variance along w_* by β

Pattern scaling analysis

$$f(x) = \sum_{i=1,c=1}^{N,C} a_{ic}\sigma\left(h_{i,c}(x)\right) \quad N \propto S \propto C \propto \sqrt{d} \qquad y(x) = \sum_{i=1}^{N} a_i^* \sigma\left(w_* \cdot [x_{S(i-1)} \ldots X_{Si}]\right)$$

$$P(w, a) \propto e^{-S|w|^2 - NC|a|^2}$$

Pattern I - one a_{ic} and one w_c specialize to teacher

$$-\log\left(\frac{P(w_{SP}, a_{SP})}{P(w_{typ}, a_{typ})}\right) \propto S + NC \propto \sqrt{d}C \propto d$$

Pattern II - first layer increases its variance along w_* by β

$$P_* \propto -\log\left(\frac{P(w_{GFL}, a_{GP})}{P(w_{typ}, a_{typ})}\right) \propto C\beta + \frac{d}{\beta} \Rightarrow_{optimize\ \ \beta} \propto d^{3/4}$$

Pattern scaling analysis

$$f(x) = \sum_{i=1,c=1}^{N,C} a_{ic}\sigma\left(h_{i,c}(x)\right) \quad N \propto S \propto C \propto \sqrt{d} \qquad y(x) = \sum_{i=1}^{N} a_i^* \sigma\left(w_* \cdot [x_{S(i-1)} \dots X_{Si}]\right)$$

$$P(w,a) \propto e^{-S|w|^2 - NC|a|^2}$$

Pattern I - one a_{ic} and one w_c specialize to teacher

$$-\log\left(\frac{P(w_{SP}, a_{SP})}{P(w_{typ}, a_{typ})}\right) \propto S + NC \propto \sqrt{d}C \propto d$$

Pattern II - first layer increases its variance along w_* by β

$$P_* \propto -\log\left(\frac{P(w_{GFL}, a_{GP})}{P(w_{typ}, a_{typ})}\right) \propto C\beta + \frac{d}{\beta} \Rightarrow_{optimize\ \beta} \propto d^{3/4}$$

Winner!

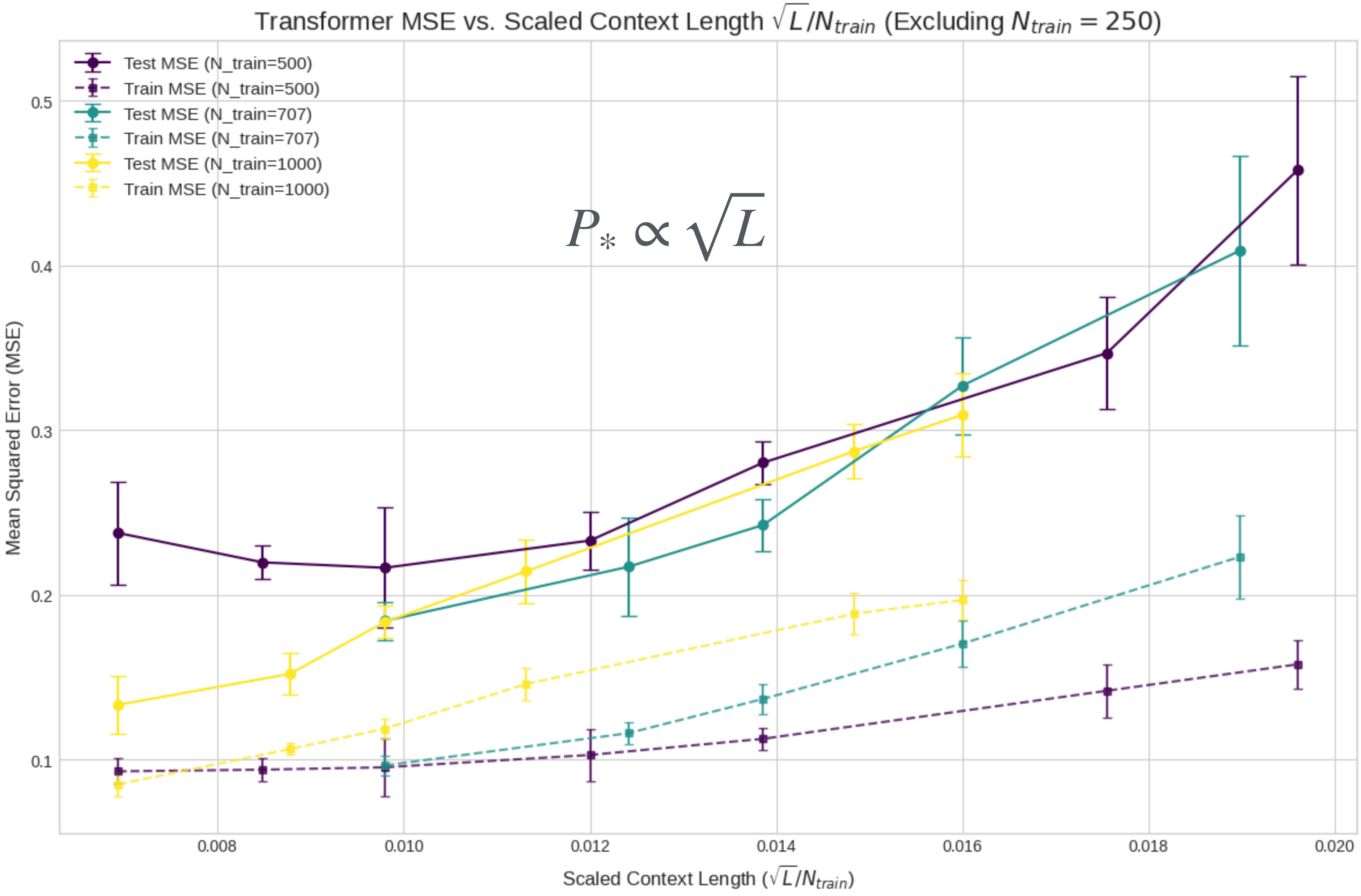
Scenarios for simplified transformer learning complex attention patterns

$$f(x) = \frac{1}{\sqrt{L}} \sum_{a,b=1}^{L} \frac{e^{-[x^a]^T A x^b}}{\sum_c e^{-[x^a]^T A x^c}} (w \cdot x^b)$$

$$y(x) = \frac{1}{L} \sum_{a,b} x_1^a x_2^a x_3^b \quad x \in R^{L\times d}$$

Winning Pattern

$$A_{12} = A_{21} = \epsilon; w_3 = \epsilon^{-1}$$



Patterns for a 3-layer FCN learning He3

$$f(x) = \sum_{c=1}^{N_2} a_c Erf(h_c(x)) \quad h_c(x) = \sum_{j=1}^{N_1} V_{cj} Erf(w_j \cdot x) \quad y(x) = He_3(x) \quad N_1 \propto N_2 \propto d$$

Patterns for a 3-layer FCN learning He3

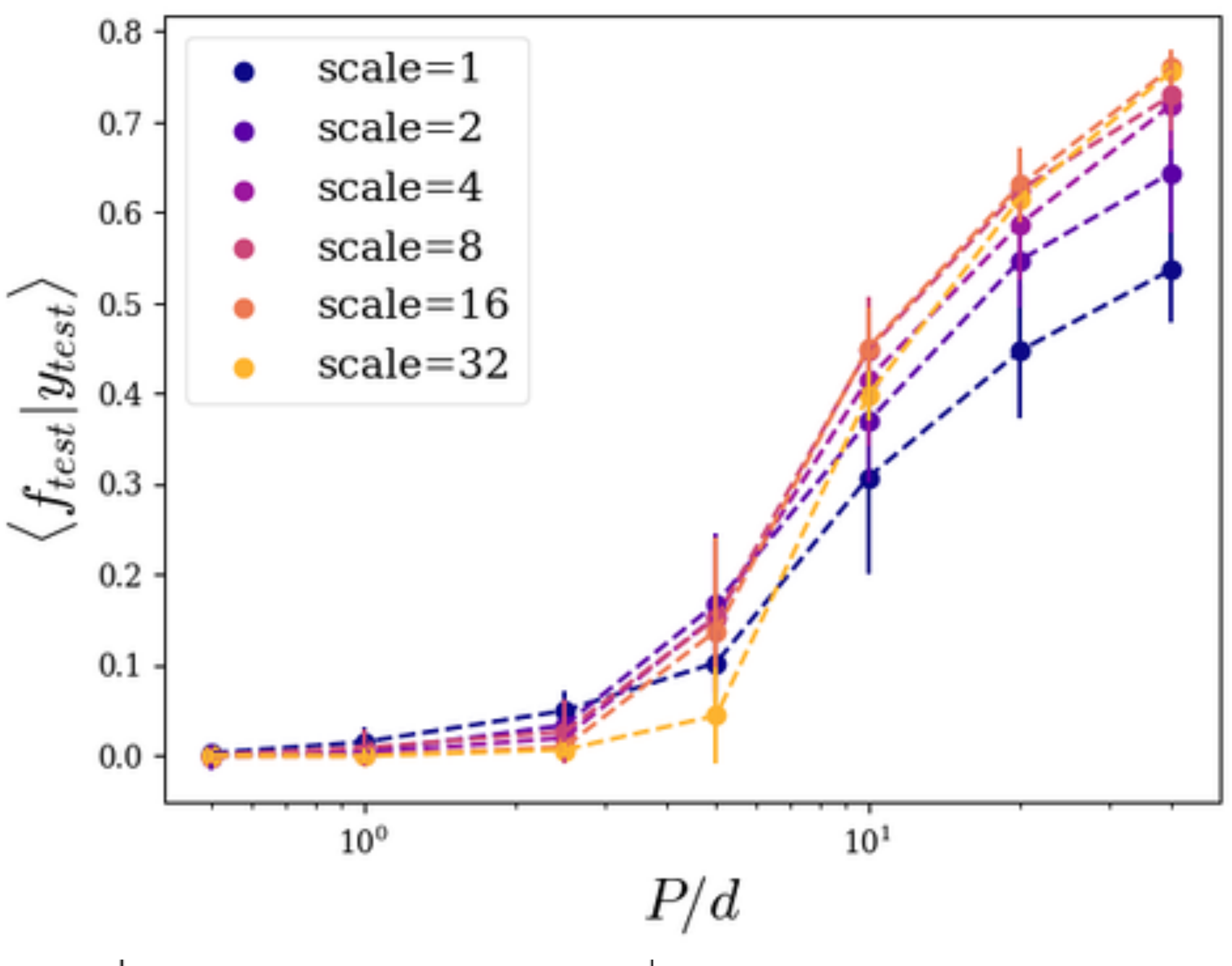
$$f(x) = \sum_{c=1}^{N_2} a_c Erf(h_c(x)) \quad h_c(x) = \sum_{j=1}^{N_1} V_{cj} Erf(w_j \cdot x) \quad y(x) = He_3(x) \quad N_1 \propto N_2 \propto d$$

Pattern Description	$-\log P_{Pattern}$	**Minimizing parameter values** ($M_i > 1$)	$-\log P_{Pattern}$ **at min params**
All layers have specialized neurons	$d + N_1 + N_2$	-	$d + N_1 + N_2$
GP in the input layer M_2 specialize in middle layer	$dM_2 + \frac{N_2}{M_2}$	$M_2 = \sqrt{\frac{N_2}{d}}$	$\sqrt{N_2 d}$
M_1 specialize in input layer GP in the rest	$dM_1 + \frac{N_1}{M_1}$	$M_1 = \sqrt{\frac{N_1}{d}}$	$\sqrt{N_1 d}$
M_1 specialize input Middle layer activations obtain $\pm\sqrt{\frac{\beta}{N_2}}y$ GP readout	$dM_1 + \frac{N_1}{M_1}\beta + \frac{N_2}{\beta}$	$\beta = \left(\frac{N_2^2}{N_1 d}\right)^{1/3}$ $M_1 = \left(\frac{N_2 N_1}{d^2}\right)^{1/3}$	$(N_2 N_1 d)^{1/3}$

Patterns for a 3-layer FCN learning He3

$$f(x) = \sum_{c=1}^{N_2} a_c Erf(h_c(x)) \quad h_c(x) = \sum_{j=1}^{N_1} V_{cj} Erf(w_j \cdot x) \quad y(x) = He_3(x) \quad N_1 \propto N_2 \propto d$$

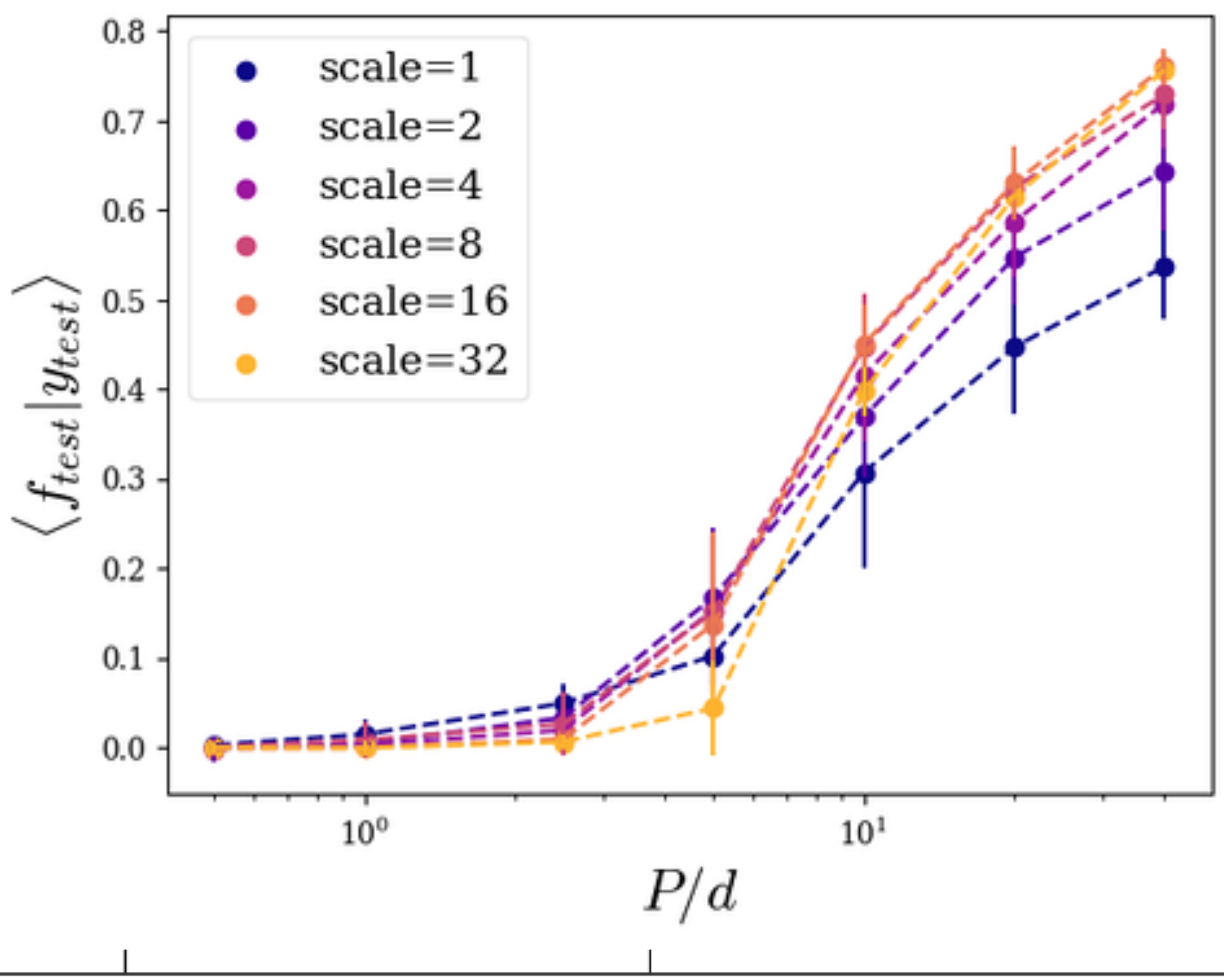
Pattern Description	$-\log P_{Pattern}$	**Minimizing parameter values** $(M_i > 1)$	$-\log P_{Pattern}$ **at min params**
All layers have specialized neurons			$d + N_1 + N_2$
GP in the input layer M_2 specialize in middle lay			$\sqrt{N_2 d}$
M_1 specialize in input layer GP in the rest			$\sqrt{N_1 d}$
M_1 specialize input Middle layer activations ob $\pm\sqrt{\frac{\beta}{N_2}}y$ GP readout			$(N_2 N_1 d)^{1/3}$



Patterns for a 3-layer FCN learning He3

$$f(x) = \sum_{c=1}^{N_2} a_c Erf(h_c(x)) \quad h_c(x) = \sum_{j=1}^{N_1} V_{cj} Erf(w_j \cdot x) \quad y(x) = He_3(x) \quad N_1 \propto N_2 \propto d$$

Pattern Description	$-\log P_{Pattern}$	**Minimizing parameter values** $(M_i > 1)$	$-\log P_{Pattern}$ **at min params**
All layers have specialized neurons			$d + N_1 + N_2$
GP in the input layer M_2 specialize in middle lay			$\sqrt{N_2 d}$
M_1 specialize in input layer GP in the rest			$\sqrt{N_1 d}$
M_1 specialize input Middle layer activations ob $\pm\sqrt{\frac{\beta}{N_2}}y$ GP readout			$(N_2 N_1 d)^{1/3}$



All scale as d
No clear winner

Predicting changes in feature learning pattern as N_1 grows

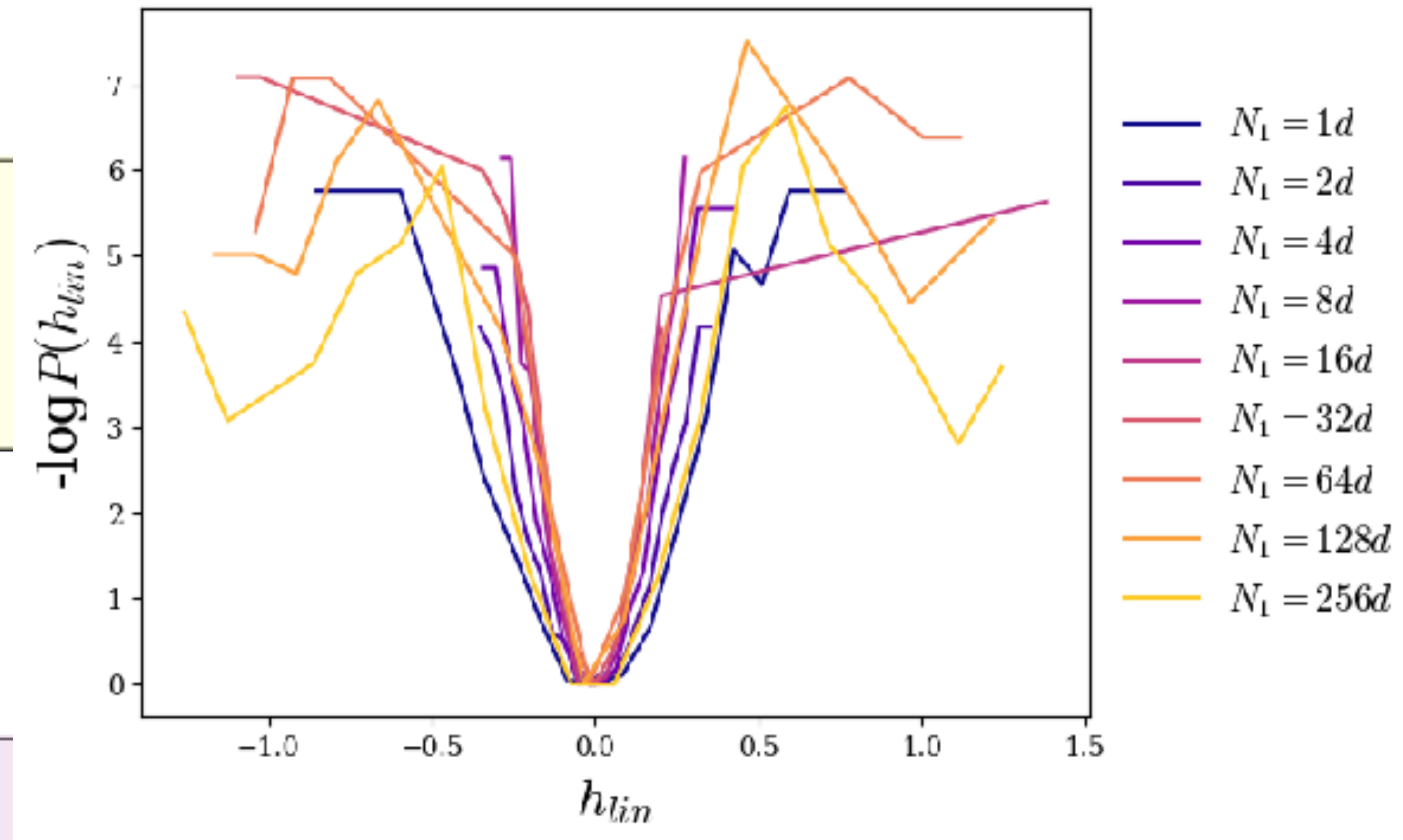
$$f(x) = \sum_{c=1}^{N_2} a_c Erf(h_c(x)) \quad h_c(x) = \sum_{j=1}^{N_1} V_{cj} Erf(w_j \cdot x) \quad y(x) = He_3(x) \quad N_1 \propto N_2 \propto d$$

Pattern Description	$-\log P_{Pattern}$	**Minimizing parameter values** $(M_i > 1)$	$-\log P_{Pattern}$ **at min params**
All layers have specialized neurons	$d + N_1 + N_2$	-	$d + N_1 + N_2$
GP in the input layer M_2 specialize in middle layer	$dM_2 + \frac{N_2}{M_2}$	$M_2 = \sqrt{\frac{N_2}{d}}$	$\sqrt{N_2 d}$
M_1 specialize in input layer GP in the rest	$dM_1 + \frac{N_1}{M_1}$	$M_1 = \sqrt{\frac{N_1}{d}}$	$\sqrt{N_1 d}$
M_1 specialize input Middle layer activations obtain $\pm\sqrt{\frac{\beta}{N_2}}y$ GP readout	$dM_1 + \frac{N_1}{M_1}\beta + \frac{N_2}{\beta}$	$\beta = \left(\frac{N_2^2}{N_1 d}\right)^{1/3}$ $M_1 = \left(\frac{N_2 N_1}{d^2}\right)^{1/3}$	$(N_2 N_1 d)^{1/3}$

Predicting changes in feature learning pattern as N_1 grows

$$f(x) = \sum_{c=1}^{N_2} a_c Erf(h_c(x)) \quad h_c(x) = \sum_{j=1}^{N_1} V_{cj} Erf(w_j \cdot x) \quad y(x) = He_3(x) \quad N_1 \propto N_2 \propto d$$

Pattern Description	$-\log P_{Pattern}$	**Minimizing parameter values** $(M_i > 1)$	$-\log P_{Pattern}$ **at min params**
All layers have specialized neurons	$d + N_1 + N_2$	-	
GP in the input layer M_2 specialize in middle layer	$dM_2 + \frac{N_2}{M_2}$	$M_2 = \sqrt{\frac{N_2}{d}}$	
M_1 specialize in input layer GP in the rest	$dM_1 + \frac{N_1}{M_1}$	$M_1 = \sqrt{\frac{N_1}{d}}$	
M_1 specialize input Middle layer activations obtain $\pm\sqrt{\frac{\beta}{N_2}}y$ GP readout	$dM_1 + \frac{N_1}{M_1}\beta + \frac{N_2}{\beta}$	$\beta = \left(\frac{N_2^2}{N_1 d}\right)^{1/3}$ $M_1 = \left(\frac{N_2 N_1}{d^2}\right)^{1/3}$	[illegible]



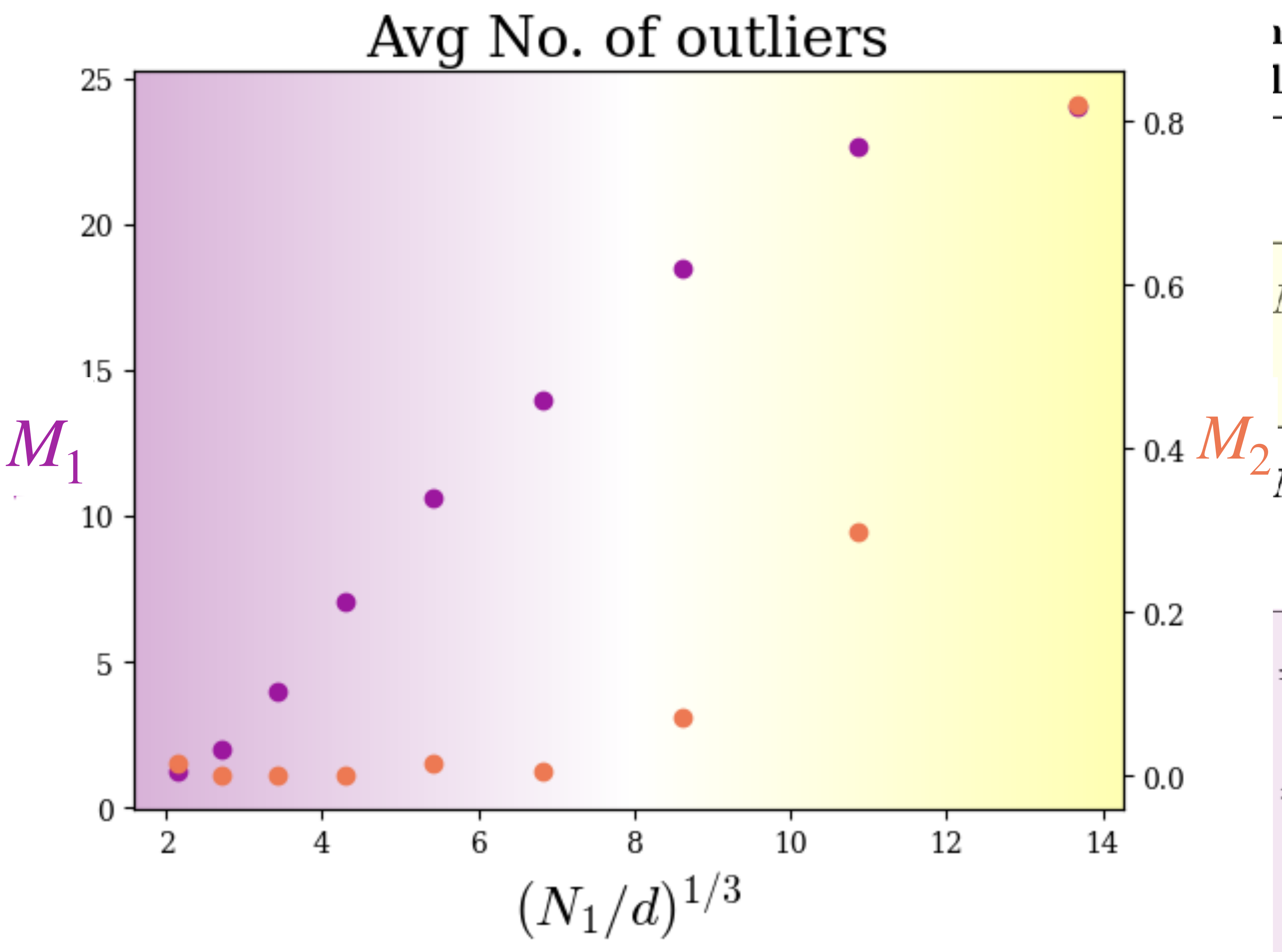
Predicting changes in feature learning pattern as N_1 grows

$$f(x) = \sum_{c=1}^{N_2} a_c Erf(h_c(x)) \quad h_c(x) = \sum_{j=1}^{N_1} V_{cj} Erf(w_j \cdot x) \quad y(x) = He_3(x) \quad N_1 \propto N_2 \propto d$$

Pattern Description	$-\log P_{Pattern}$	**Minimizing parameter values** $(M_i > 1)$	$-\log P_{Pattern}$ **at min params**
All layers have specialized neurons	$d + N_1 + N_2$	-	$d + N_1 + N_2$
GP in the input layer M_2 specialize in middle layer	$dM_2 + \frac{N_2}{M_2}$	$M_2 = \sqrt{\frac{N_2}{d}}$	$\sqrt{N_2 d}$
M_1 specialize in input layer GP in the rest	$dM_1 + \frac{N_1}{M_1}$	$M_1 = \sqrt{\frac{N_1}{d}}$	$\sqrt{N_1 d}$
M_1 specialize input Middle layer activations obtain $\pm\sqrt{\frac{\beta}{N_2}}y$ GP readout	$dM_1 + \frac{N_1}{M_1}\beta + \frac{N_2}{\beta}$	$\beta = \left(\frac{N_2^2}{N_1 d}\right)^{1/3}$ $M_1 = \left(\frac{N_2 N_1}{d^2}\right)^{1/3}$	$(N_2 N_1 d)^{1/3}$

Predicting changes in feature learning pattern as N_1 grows

$$f(x) = \sum_{c=1}^{N_2} a_c Erf(h_c(x)) \quad h_c(x) = \sum_{j=1}^{N_1} V_{cj} Erf(w_j \cdot x) \quad y(x) = He_3(x) \quad N_1 \propto N_2 \propto d$$

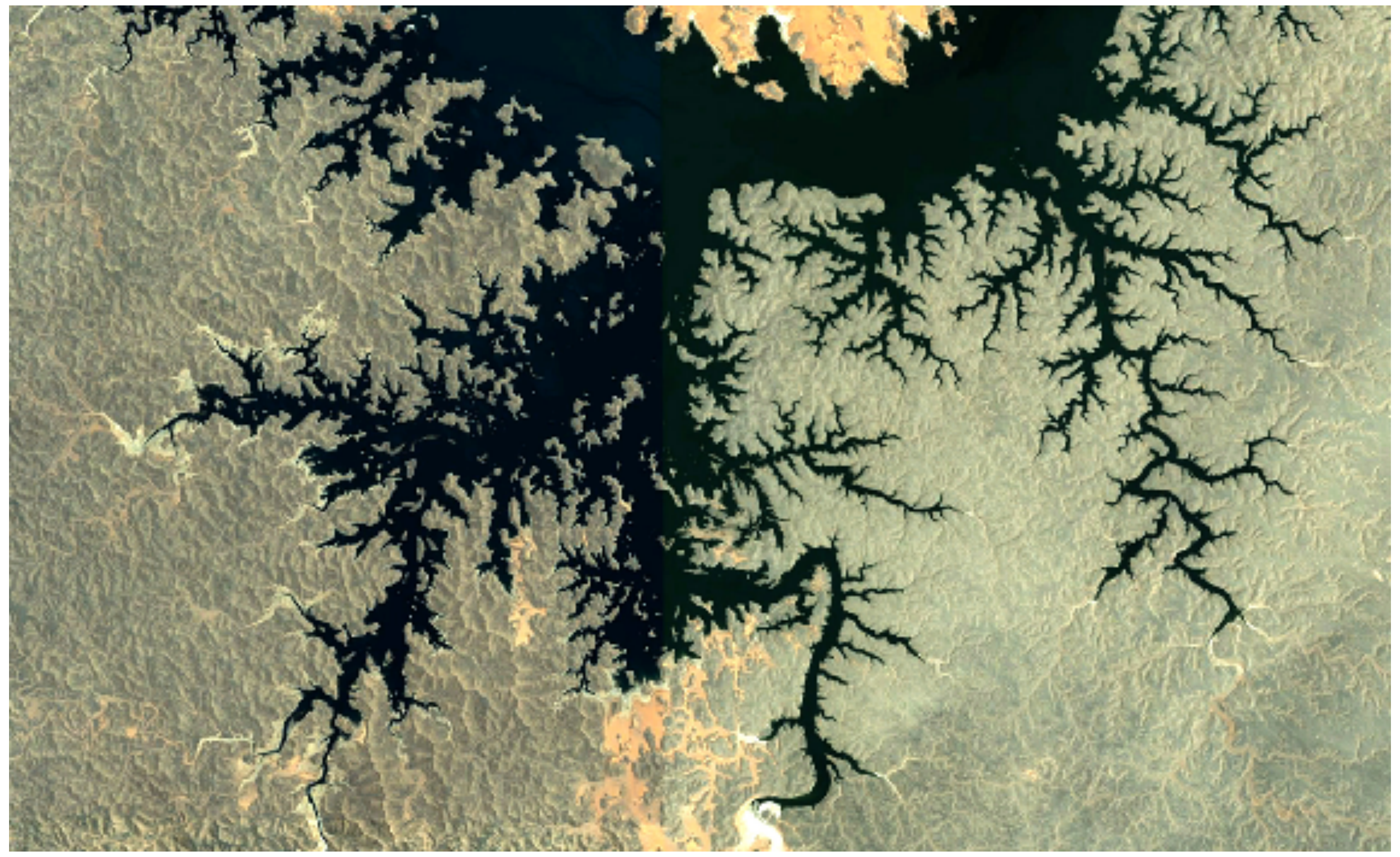


...izing parameter ...lues ($M_i > 1$)	$-\log P_{Pattern}$ at min params
-	$d + N_1 + N_2$
$M_2 = \sqrt{\frac{N_2}{d}}$	$\sqrt{N_2 d}$
$M_1 = \sqrt{\frac{N_1}{d}}$	$\sqrt{N_1 d}$
$= \left(\frac{N_2^2}{N_1 d}\right)^{1/3}$ $= \left(\frac{N_2 N_1}{d^2}\right)^{1/3}$	$(N_2 N_1 d)^{1/3}$

Summary of Verified Heuristic Results

Assign [GP,Specialization,GFL] to each layer => estimate layer-wise log prob. => sum those up

- P=d sample complexity for y=He3(x_1) and how specialization wins over GFL in 2-layer Erf FCN
- P=d^{3/4} sample complexity for “wide” CNN with single index teacher and how GFL wins over specialization
- P=d sample complexity for y=He3(x) in 3-layer Erf FCN, scaling of the specializing neurons with width’s, and transition between two feature learning patterns
- P=Context-length^{1/2} sample complexity for a soft-max attention model learning a two-sequence-index target.



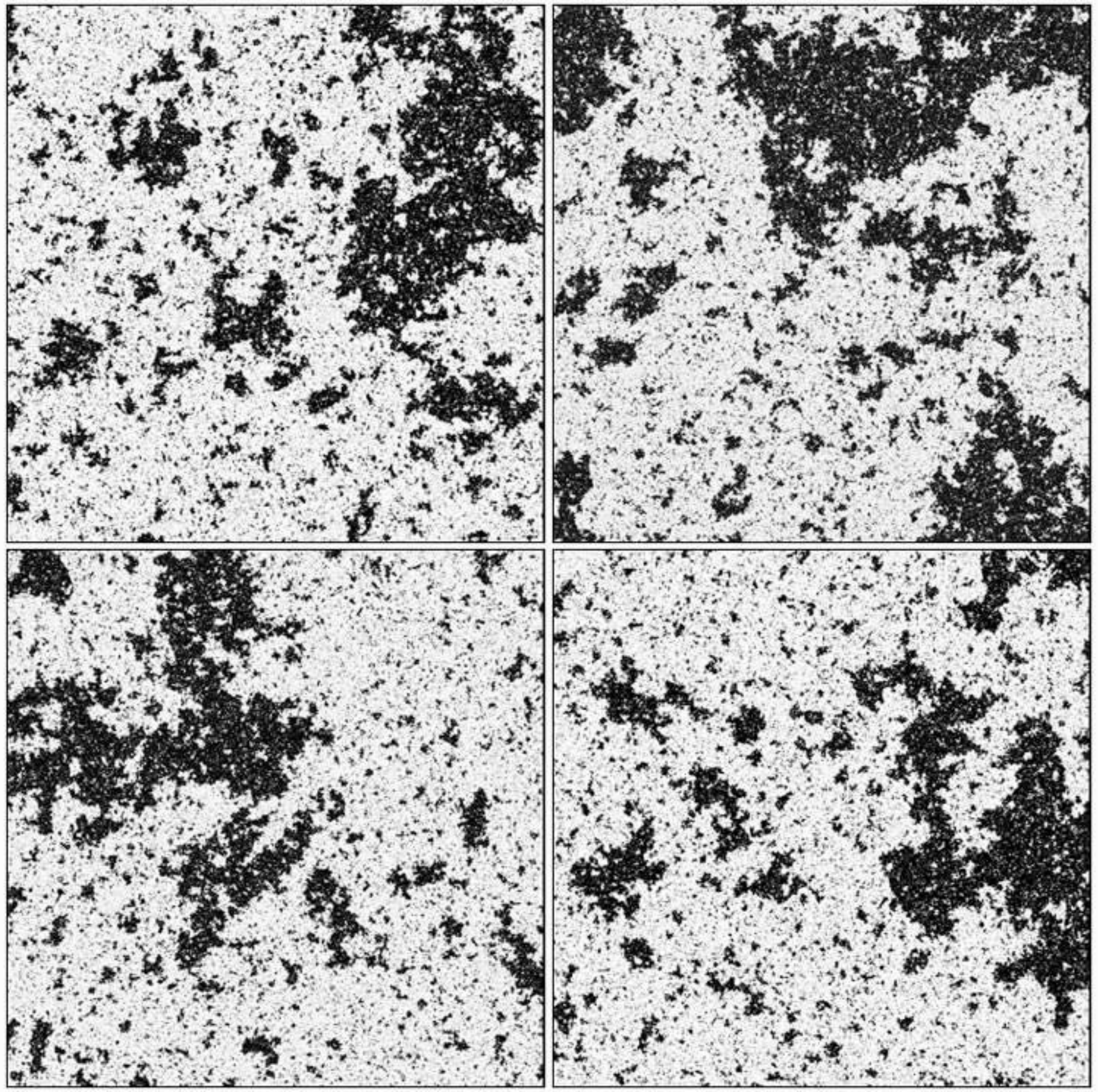
Renormalization group flows of neural networks
under gradual removal of high RKHS subspaces

Howard et. al. Wilsonian RG of NNGPs (2025)

Gorka et. al. RG flows, Universality and Irrelevance in Overparametrized Deep Neural Networks (TBP)

Self-similarity and power-law scaling

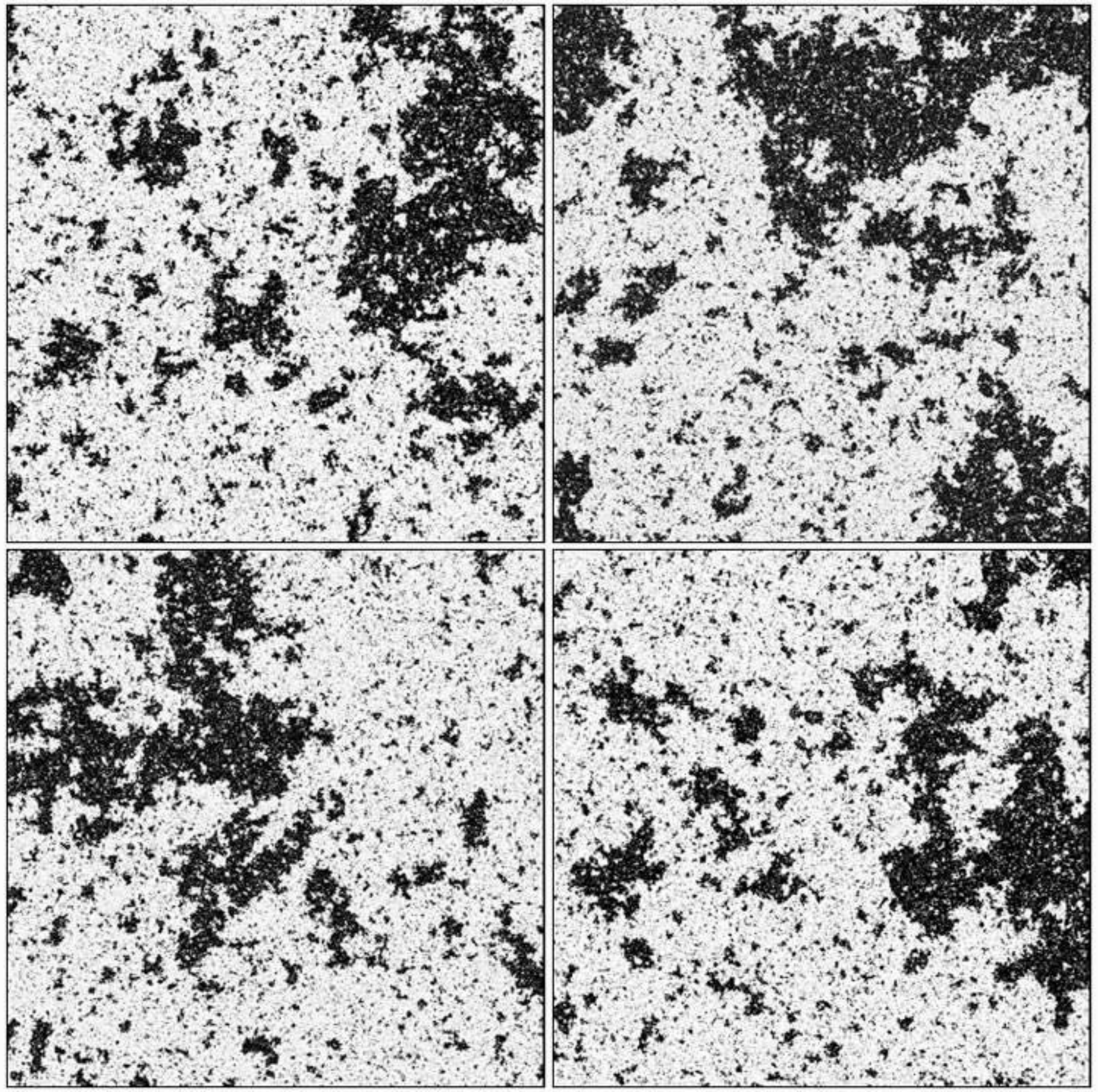
Critical 2d Ising Model



$$\langle s(0)s(x)\rangle \propto \frac{1}{x^{\gamma}}$$

Self-similarity and power-law scaling

Critical 2d Ising Model



$$\langle s(0)s(x)\rangle \propto \frac{1}{x^{\gamma}}$$

Self-similarity and universality

- **In a nut-shell:** Microscopic information is loss over so many length scales however, due to self-similarity, the macroscopic phenomena remains the same.

- **In detail**: Wilsonian RG, integration-out high wavelength physics, re-scaling, relevant and irrelevant operators.

Evidence for self-similarity and universality in deep learning

Robustness based evidence

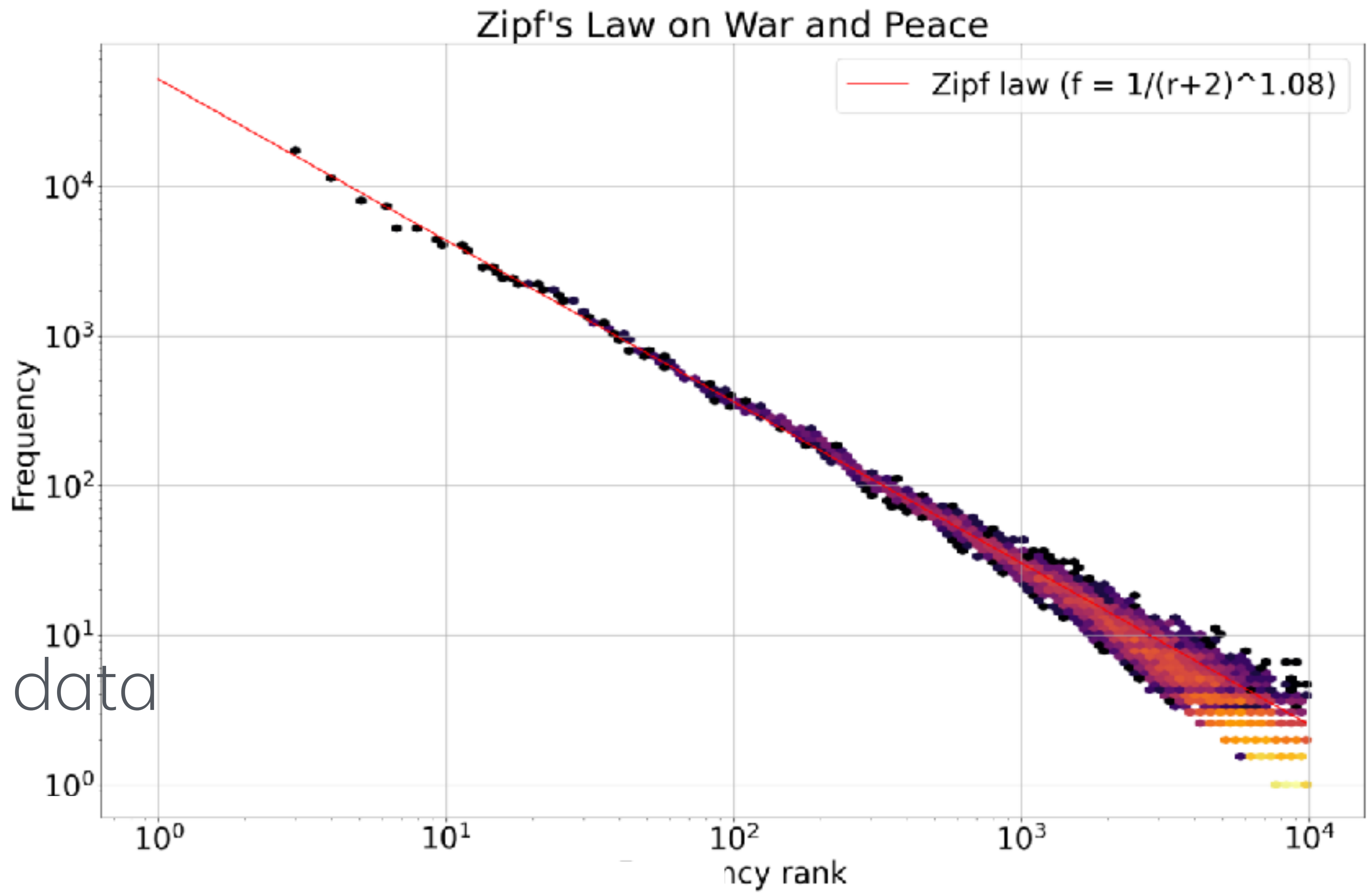
- Large models are quite robust (i.e. 2-3% changes) following:
  - Changes to loss functions between MSE loss, L1 loss, cross entropy loss
  - Changes to architecture within the same symmetry classes.
  - Changes to training algorithm (Large/small batch SGD, SGD with Momentum, Adam etc.. though training speed can be highly affected)
  - Changes to hyper-parameters such as weight decay, layer widths, learning rates.

Self-similarity based evidence - data shows power laws

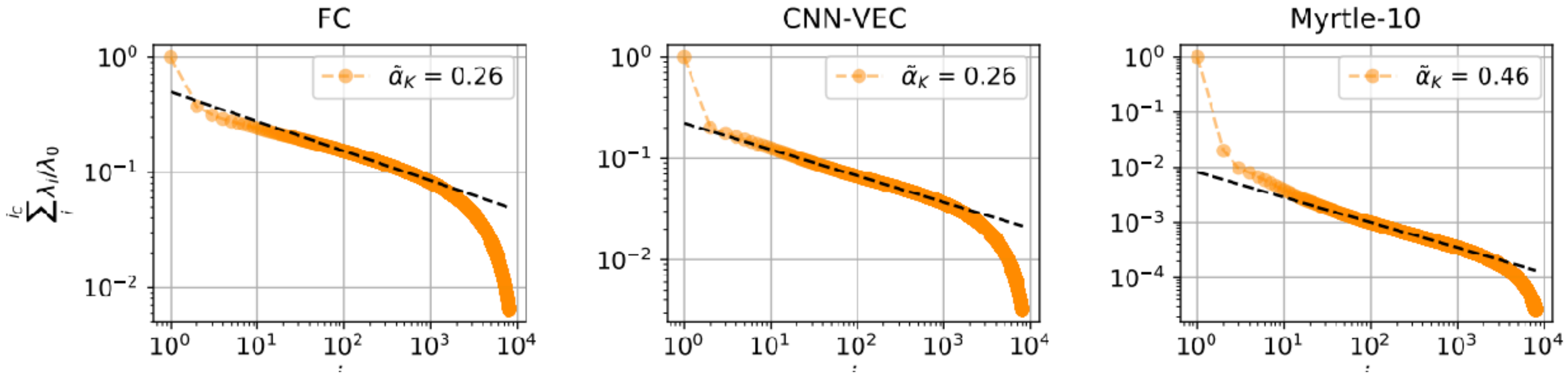
- Zipf's law

$$\text{frequency} \propto \frac{1}{(\text{rank} + b)^a}$$

where a, b are fitted parameters, with $a \approx 1$, and $b \approx 2.7$.[1]



- Kernel spectra (generalized PCA) of real world data



Self-similarity based evidence - performance shows power laws

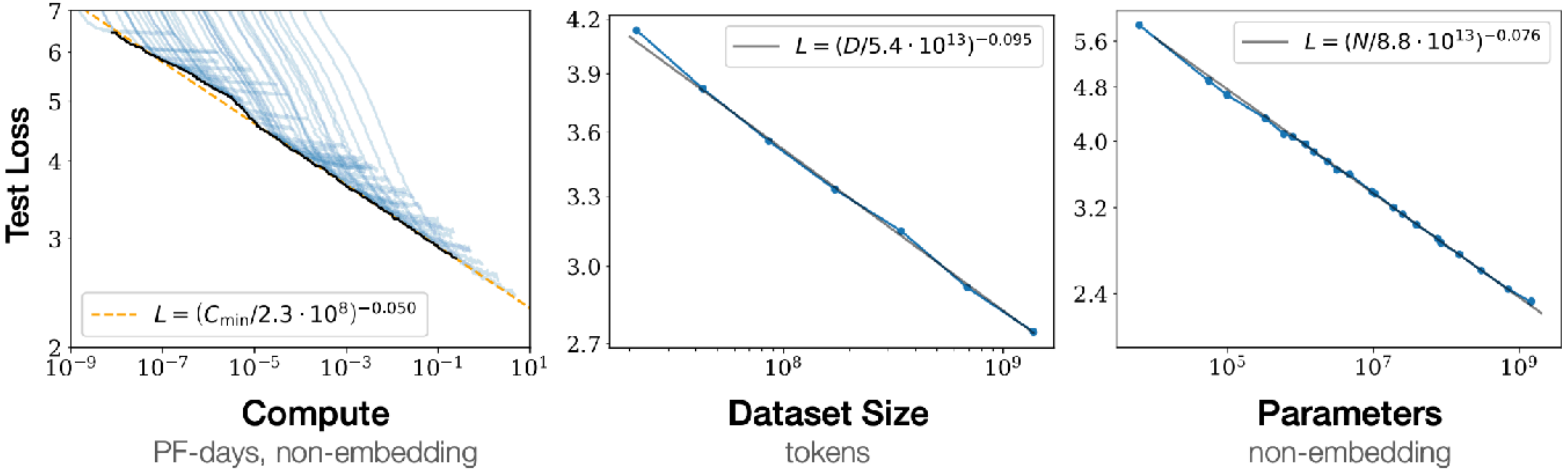


Figure 1 Language modeling performance improves smoothly as we increase the model size, datasetset size, and amount of compute[2] used for training. For optimal performance all three factors must be scaled up in tandem. Empirical performance has a power-law relationship with each individual factor when not bottlenecked by the other two.

Scaling Laws for Neural Language Models

Jared Kaplan *
Johns Hopkins University, OpenAI
jaredk@jhu.edu

Sam McCandlish*
OpenAI
sam@openai.com

Towards a scaling theory of DNNs

Self-similarity, Power-laws, RG, Universality, Complex data…

- Power-laws are very common in deep learning and in self-similar physical systems.
- However while self-similarity implies power-laws, the converse is less clear — in particular it requires a notion of scale and coarse graining, namely RG.
- Establishing a useful notion of RG on deep learning can relate power-laws to self-similarity and to the theoretical holy-grail of universality.

Towards a scaling theory of DNNs

Self-similarity, Power-laws, RG, Universality, Complex data...

- Power-laws are very common in deep learning and in self-similar physical systems.
- However while self-similarity implies power-laws, the converse is less clear — in particular it requires a notion of scale and coarse graining, namely RG.
- Establishing a useful notion of RG on deep learning can relate power-laws to self-similarity and to the theoretical holy-grail of universality.

Howard et. al. Wilsonian RG of NNGPs (2025)
Gorka et. al. RG flows, Universality and Irrelevance in Overparametrized Deep Neural Networks (TBP)

Towards a scaling theory of DNNs

Self-similarity, Power-laws, RG, Universality, Complex data…

- Power-laws are very common in deep learning and in self-similar physical systems.
- However while self-similarity implies power-laws, the converse is less clear — in particular it requires a notion of scale and coarse graining, namely RG.
- Establishing a useful notion of RG on deep learning can relate power-laws to self-similarity and to the theoretical holy-grail of universality.

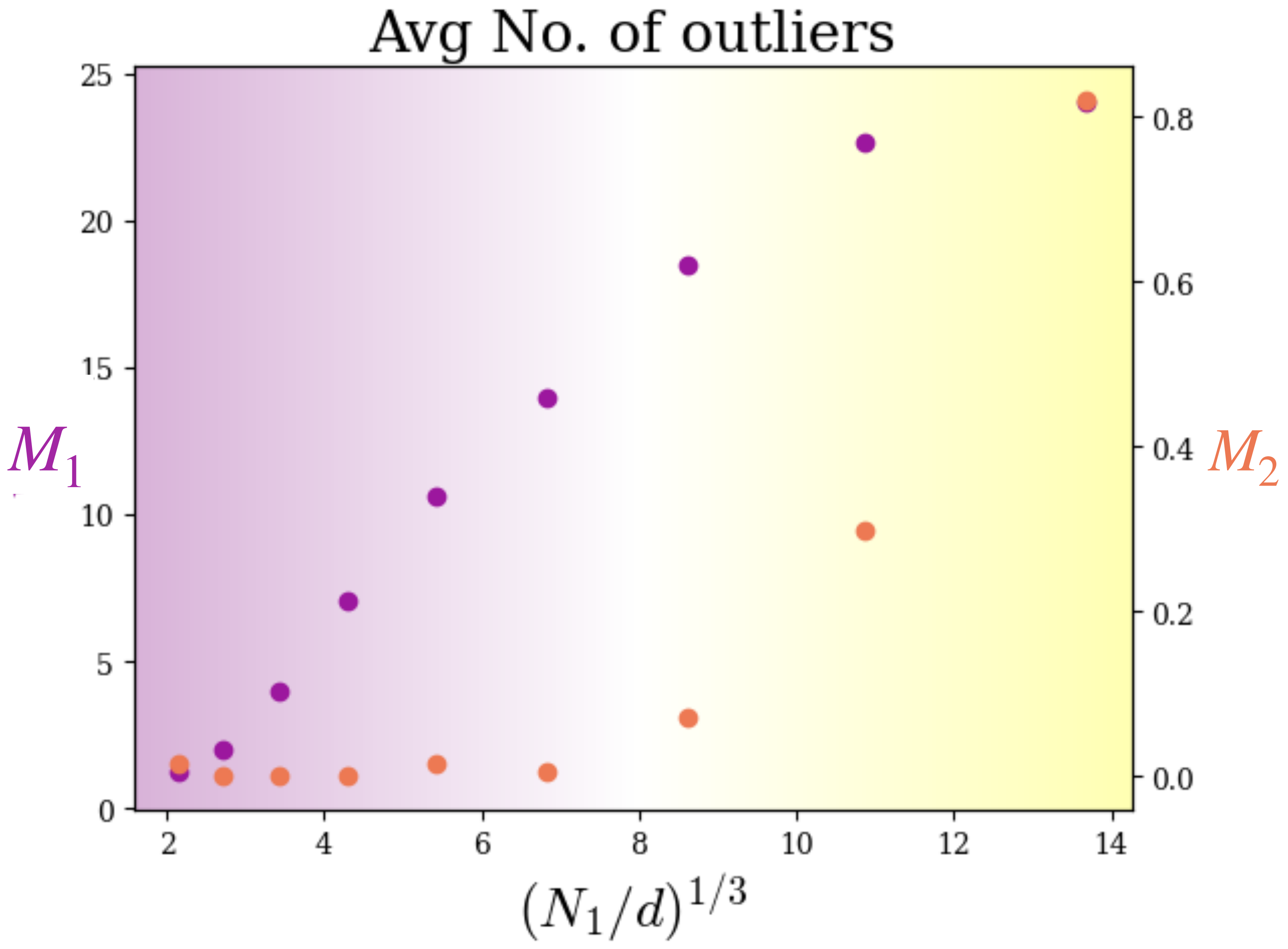
Howard et. al. Wilsonian RG of NNGPs (2025)

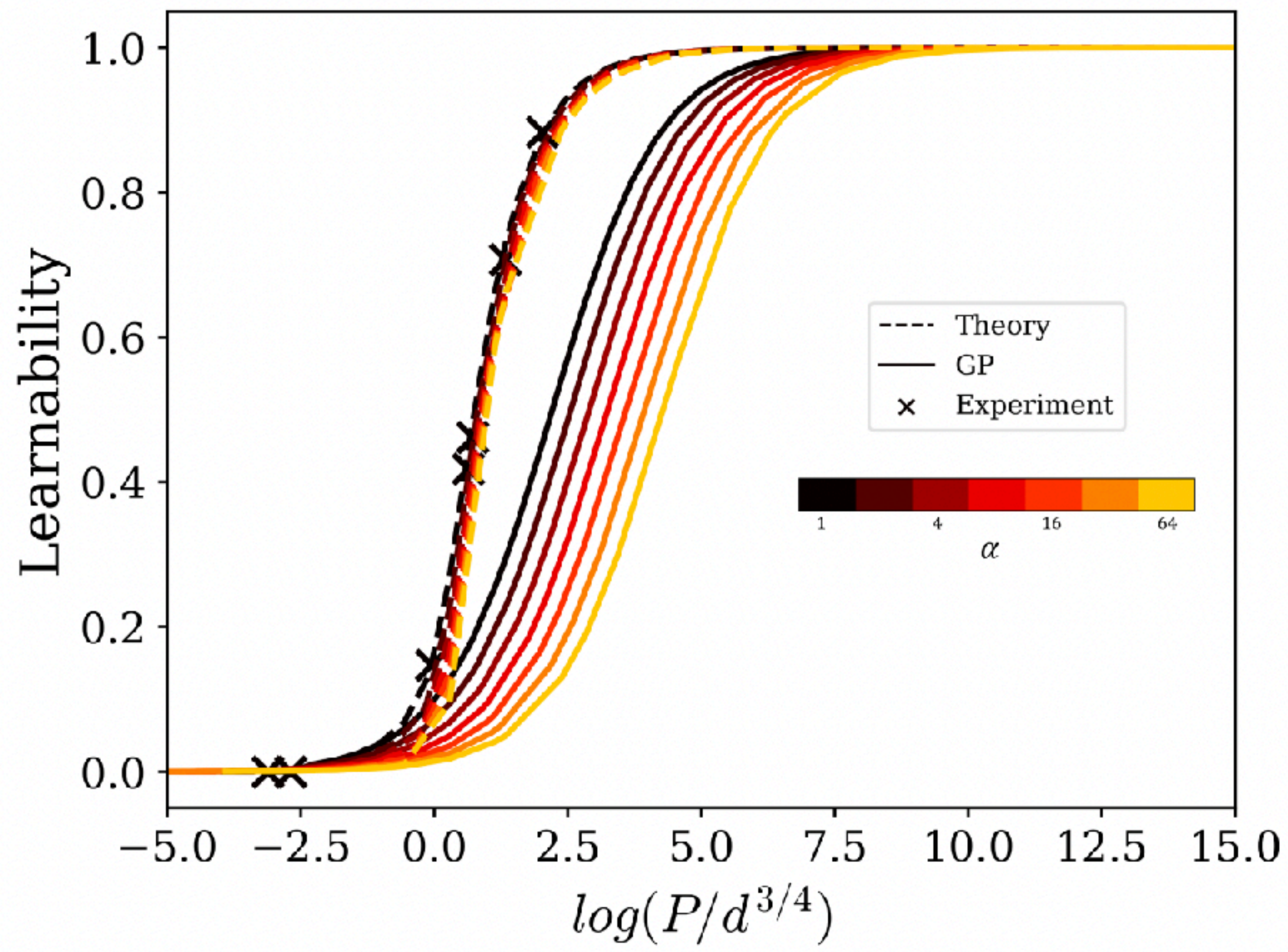
Gorka et. al. RG flows, Universality and Irrelevance in Overparametrized Deep Neural Networks (TBP)

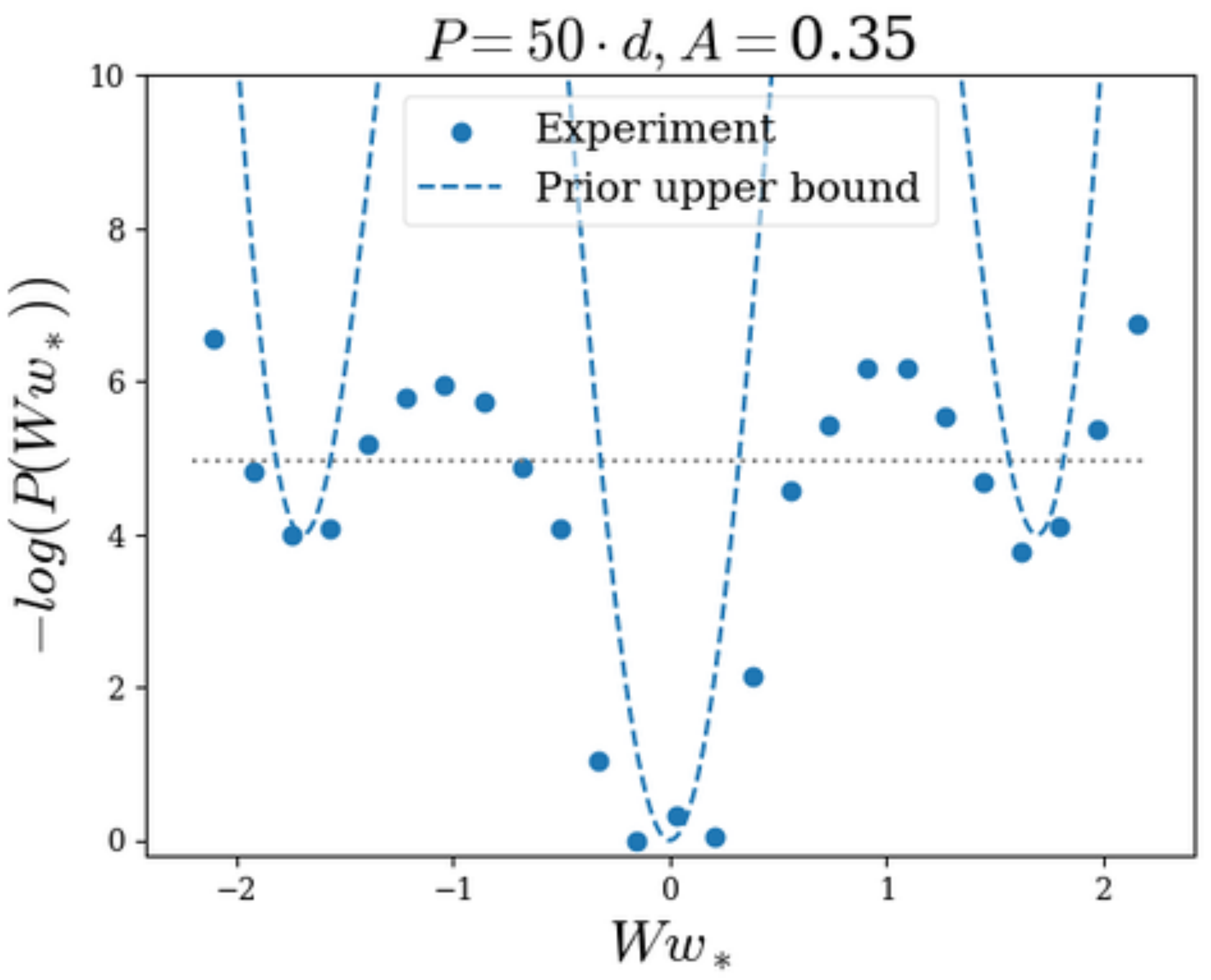
$$S_{GP}[f(x)] = \sum_{k=1}^{\Lambda} \lambda_k^{-1} f_k^2 + r \int d\mu_x (f(x) - y(x))^2 + u \int d\mu_x (f(x) - y(x))^4 \quad \lambda_k = k^{-1-\alpha}$$

Not a ϕ^4 flow! No WF fixed point

Summary







Pattern Description	$-\log P_{Pattern}$	**Minimizing parameter values** ($M_i > 1$)	$-\log P_{Pattern}$ **at min params**
All layers have specialized neurons	$d + N_1 + N_2$	-	$d + N_1 + N_2$
GP in the input layer M_2 specialize in middle layer	$dM_2 + \frac{N_2}{M_2}$	$M_2 = \sqrt{\frac{N_2}{d}}$	$\sqrt{N_2 d}$
M_1 specialize in input layer GP in the rest	$dM_1 + \frac{N_1}{M_1}$	$M_1 = \sqrt{\frac{N_1}{d}}$	$\sqrt{N_1 d}$
M_1 specialize input Middle layer activations obtain $\pm\sqrt{\frac{\beta}{N_2}}y$ GP readout	$dM_1 + \frac{N_1}{M_1}\beta + \frac{N_2}{\beta}$	$\beta = \left(\frac{N_2^2}{N_1 d}\right)^{1/3}$ $M_1 = \left(\frac{N_2 N_1}{d^2}\right)^{1/3}$	$(N_2 N_1 d)^{1/3}$

- Harder + real-world data + connection with Mech. Int.
- Implicit bias of feature learning in deeper networks
- Sparsity effects and interaction between features
- Overfitting patterns