

Practical neural network theory: from statistical mechanics basics to heavy-tailed self regularization to working with state of the art models

Michael W. Mahoney

ICSI, LBNL, and Dept of Statistics, UC Berkeley

<http://www.stat.berkeley.edu/~mmahoney/>

April 2023

(Joint work with Charles H. Martin, Yaoqing Yang, Ryan Theisen, Zhenyu Liao, Amir Gholami, Liam Hodgkinson, and many others)

Overview

- Weight Analysis and Heavy-Tailed Self-Regularization
- Phenomenological Approach to Statistical Mechanics of Generalization
- Using Heavy-Tailed Self-Regularization
- Random Matrix Theory for Modern ML
- Putting It All Together

Outline

- 1 Weight Analysis and Heavy-Tailed Self-Regularization
- 2 Phenomenological Approach to Statistical Mechanics of Generalization
- 3 Using Heavy-Tailed Self-Regularization
- 4 Random Matrix Theory for Modern ML
- 5 Putting It All Together
- 6 Conclusion

Statistical Mechanics Methods for Discovering Knowledge from Production-Scale Neural Networks

Charles H. Martin* and Michael W. Mahoney†

Tutorial at ACM-KDD, August 2019

*Calculation Consulting, charles@calculationconsulting.com

†ICSI and Dept of Statistics, UC Berkeley, <https://www.stat.berkeley.edu/~mmahoney/>

Outline

1 Prehistory and History

- Older Background
- A Very Simple Deep Learning Model
- More Immediate Background

2 Preliminary Results

- Regularization and the Energy Landscape
- Preliminary Empirical Results
- Gaussian and Heavy-tailed Random Matrix Theory

3 Developing a Theory for Deep Learning

- More Detailed Empirical Results
- An RMT-based Theory for Deep Learning
- Tikhonov Regularization versus Heavy-tailed Regularization

4 Validating and Using the Theory

- Varying the Batch Size: Explaining the Generalization Gap
- Using the Theory: `pip install weightwatcher`
- Diagnostics at Scale: Predicting Test Accuracies

5 More General Implications and Conclusions

Outline

1 Prehistory and History

- Older Background
- A Very Simple Deep Learning Model
- More Immediate Background

2 Preliminary Results

- Regularization and the Energy Landscape
- Preliminary Empirical Results
- Gaussian and Heavy-tailed Random Matrix Theory

3 Developing a Theory for Deep Learning

- More Detailed Empirical Results
- An RMT-based Theory for Deep Learning
- Tikhonov Regularization versus Heavy-tailed Regularization

4 Validating and Using the Theory

- Varying the Batch Size: Explaining the Generalization Gap
- Using the Theory: `pip install weightwatcher`
- Diagnostics at Scale: Predicting Test Accuracies

5 More General Implications and Conclusions

Statistical Physics & Neural Networks: A Long History

- 60s:
 - ▶ J. D. Cowan, *Statistical Mechanics of Neural Networks*, 1967.
- 70s:
 - ▶ W. A. Little, "The existence of persistent states in the brain," *Math. Biosci.*, 1974.
- 80s:
 - ▶ H. Sompolinsky, "Statistical mechanics of neural networks," *Physics Today*, 1988.
- 90s:
 - ▶ D. Haussler, M. Kearns, H. S. Seung, and N. Tishby, "Rigorous learning curve bounds from statistical mechanics," *Machine Learning*, 1996.
- 00s:
 - ▶ A. Engel and C. P. L. Van den Broeck, *Statistical mechanics of learning*, 2001.

Hopfield model

Hopfield, "Neural networks and physical systems with emergent collective computational abilities," PNAS 1982.

Hopfield model:

- Recurrent artificial neural network model
- Equivalence between behavior of NNs with symmetric connections and the equilibrium statistical mechanics behavior of certain magnetic systems.
- Can design NNs for associative memory and other computational tasks

Phase diagram with three kinds of phases (α is load parameter):

- Very low α regime: model has so so much capacity, it is a prototype method
- Intermediate α : spin glass phase, which is "pathologically non-convex"
- Higher α : generalization phase

But:

- Lots of subsequent work focusing on spin glasses, replica theory, etc.

Let's go back to the basics!

Restricted Boltzmann Machines and Variational Methods

RBM = Hopfield + temperature + backprop:

RBM and other more sophisticated variational free energy methods

- They have an intractable partition function.
- Goal: try to approximate partition function / free energy.
- Also, recent work on their phase diagram.

We do NOT do this.

Memorization, then and now.

- Three (then) versus two (now) phases.
- Modern “memorization” is probably more like spin glass phase.

Let's go back to the basics!

Some other signposts

- Cowen's introduction of sigmoid into neuroscience.
- Parisi's replica theory computations.
- Solla's statistical analysis.
- Gardner's analysis of annealed versus quenched entropy.
- Saad's analysis of dynamics of SGD.
- More recent work on dynamics, energy landscapes, etc.

Lots more ...

Important: Our Methodological Approach

Most people like training and validating ideas by training.

We will use **pre-trained models**.

- Many state-of-the-art models are publicly available.
- They are “machine learning models that work” ... so analyze them.
- Selection bias: you can't talk with deceased patients.

Of course, one could use these methods to improve training ... we won't.

Benefits of this methodological approach.

- Can develop a practical theory.
(Current theory isn't ... loose bounds and convergence rates.)
- Can evaluate theory on state-of-the-art models.
(Big models are different than small ... easily-trainable models.)
- Can be more reproducible.
(Training isn't reproducible ... too many knobs.)

You can **“pip install weightwatcher”**

Outline

1 Prehistory and History

- Older Background
- A Very Simple Deep Learning Model
- More Immediate Background

2 Preliminary Results

- Regularization and the Energy Landscape
- Preliminary Empirical Results
- Gaussian and Heavy-tailed Random Matrix Theory

3 Developing a Theory for Deep Learning

- More Detailed Empirical Results
- An RMT-based Theory for Deep Learning
- Tikhonov Regularization versus Heavy-tailed Regularization

4 Validating and Using the Theory

- Varying the Batch Size: Explaining the Generalization Gap
- Using the Theory: `pip install weightwatcher`
- Diagnostics at Scale: Predicting Test Accuracies

5 More General Implications and Conclusions

PAC/VC versus Statistical Mechanics Approaches (1 of 2)

Basic Student-Teacher Learning Setup:

- Classify elements of *input space* X into $\{0, 1\}$
- Target rule / *teacher* T ; and *hypothesis space* \mathcal{F} of possible mappings
- Given T for $\mathcal{X} \subset X$, the *training set*, select a *student* $f^* \in \mathcal{F}$, and evaluate how well f^* approximates T on X
- *Generalization error* (ϵ): probability of disagreement bw student and teacher on X
- *Training error* (ϵ_t): fraction of disagreement bw student and teacher on \mathcal{X}
- *Learning curve*: behavior of $|\epsilon_t - \epsilon|$ as a function of control parameters

PAC/VC Approach:

- Related to statistical problem of convergence of frequencies to probabilities

Statistical Mechanics Approach:

- Exploit the thermodynamic limit from statistical mechanics

PAC/VC versus Statistical Mechanics Approaches (2 of 2)

PAC/VC: get *bounds* on *worst-case* results

- View $m = |\mathcal{X}|$ as the main control parameter; fix the function class \mathcal{F} ; and ask how $|\epsilon_t - \epsilon|$ varies
- Natural to consider $\gamma = P[|\epsilon_t - \epsilon| > \delta]$
 - ▶ Related to problem of convergence of frequencies to probabilities
 - ▶ Hoeffding-type approach not appropriate (f^* depends on training data)
- Fix \mathcal{F} and construct uniform bound $P[\max_{h \in \mathcal{F}} |\epsilon_t(h) - \epsilon(h)| > \delta] \leq 2|\mathcal{F}| e^{-2m\delta^2}$
 - ▶ Straightforward if $|\mathcal{F}| < \infty$; use VC dimension (etc.) otherwise

Statistical Mechanics: get *precise* results for *typical* configurations

- Function class $\mathcal{F} = \mathcal{F}_N$ varies with m ; and let m and (size of \mathcal{F}) vary in well-defined manner
- *Thermodynamic limit*: $m, N \rightarrow \infty$ s.t. $\alpha = \frac{m}{N}$ (like load in associative memory models).
 - ▶ Limit s.t. (when it exists) certain quantities get sharply peaked around their most probable value.
 - ▶ Describe learning curve as competition between error (energy) and log of number of functions with that energy (entropy)
 - ▶ Get precise results for typical (most probably in that limit) quantities

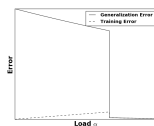
Rethinking generalization requires revisiting old ideas

Martin and Mahoney <https://arxiv.org/abs/1710.09553>

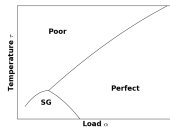
Very Simple Deep Learning (VSDL) model:

- DNN is a black box, load-like parameters α , & temperature-like parameters τ
- Adding noise to training data decreases α
- Early stopping increases τ

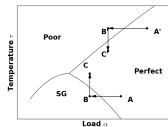
Nearly any non-trivial model[‡] exhibits “phase diagrams,” with *qualitatively* different generalization properties, for different parameter values.



(a) Training/general-ization error.



(b) Learning phases in τ - α plane.



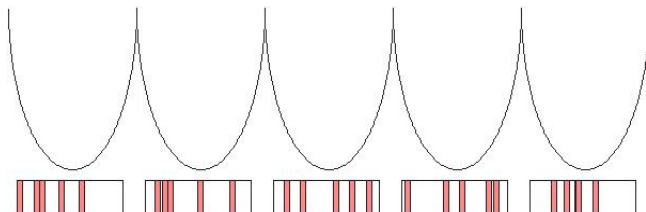
(c) Noisifying data and adjusting knobs.

[‡]when analyzed via the *Statistical Mechanics Theory of Generalization (SMToG)* ▶

Remembering Regularization

Martin and Mahoney <https://arxiv.org/abs/1710.09553>

Statistical Mechanics (1990s): (this) Overtraining \rightarrow Spin Glass Phase



Binary Classifier with N Random Labelings:

2^N over-trained solutions: locally (ruggedly) convex, very high barriers, all unable to generalize
implication: solutions inside basins should be more similar than solutions in different basins

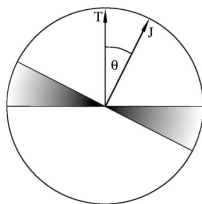
Stat Mech Setup: Student Teacher Model

Martin and Mahoney <https://arxiv.org/abs/1710.09553>

Given N labeled data points

- Imagine a Teacher Network \mathbf{T} that maps data to labels
- Learning finds a Student \mathbf{J} similar to the Teacher \mathbf{T}
- Consider all possible Student Networks \mathbf{J} for all possible teachers \mathbf{T}

The *Generalization error* ϵ is related to the phase space volume Ω_ϵ of all possible Student-Teacher overlaps for all possible \mathbf{J}, \mathbf{T}



$$\epsilon = \arccos R, \quad R = \frac{1}{N} \mathbf{J}^\dagger \mathbf{T}$$

Stat Mech Setup: Student Teacher Model

Martin and Mahoney <https://arxiv.org/abs/1710.09553>

Statistical Mechanics Foundations:

- Spherical (Energy) Constraints: $\delta(\text{Tr}[\mathbf{J}^2] - N)$
- Teacher Overlap (Potential): $\delta(\frac{1}{N} \text{Tr}[\mathbf{J}^\dagger \mathbf{T}] - \cos(\pi\epsilon))$
- Teacher Phase Space Volume (Density of States):

$$\Omega_T(\epsilon) = \int d\mathbf{J} \delta(\text{Tr}[\mathbf{J}^2] - N) \delta(\frac{1}{N} \text{Tr}[\mathbf{J}^\dagger \mathbf{T}] - \cos(\pi\epsilon))$$

Comparison to traditional Statistical Mechanics:

- Phase Space Volume, free particles:

$$\Omega_E = \int d^N \mathbf{r} \int d^N \mathbf{p} \delta\left(\sum_i \frac{\mathbf{p}_i^2}{2m_i} - E\right) \sim V^N$$

- Canonical Ensemble: Legendre Transform in $R = \cos(\pi\epsilon)$:

actually more technical, and must choose sign convention on $\text{Tr}[\mathbf{J}^\dagger \mathbf{T}]$, \mathcal{H}

$$\Omega_\beta(R) \sim \int d\mu(\mathbf{J}) e^{-\lambda \text{Tr}[\mathbf{J}^\dagger \mathbf{T}]} \sim \int d\mathbf{q}^N d\mathbf{p}^N e^{-\beta \mathcal{H}(\mathbf{p}, \mathbf{q})}$$

Stat Mech Setup: Student Teacher Model

Martin and Mahoney <https://arxiv.org/abs/1710.09553>

Early Models: Perception: \mathbf{J}, \mathbf{T} N -dim vectors

- Continuous Perception $J_i \in \mathbb{R}$ (not so interesting)
- Ising Perception $J_i = \pm 1$ (sharp transitions, requires Replica theory)

Our Proposal: \mathbf{J}, \mathbf{T} ($N \times M$) Real (possibly Heavy Tailed) matrices

- Practical Applications: Hinton, Bengio, etc.
- Related to complexity of (Levy) spin glasses (Bouchaud)

Our Expectation:

- Heavy-tailed structure means there is less capacity/entropy available for integrals, which will affect generalization properties non-trivially
- Multi-class classification is very different than binary classification

Student Teacher: Recent Practical Application

“Similarity of Neural Network Representations Revisited”

Kornblith, Norouzi, Lee, Hinton; <https://arxiv.org/abs/1905.00414>

- Examined different Weight matrix similarity metrics
- Best method: Canonical Correlation Analysis (CCA): $\|\mathbf{Y}^\dagger \mathbf{X}\|_F^2$

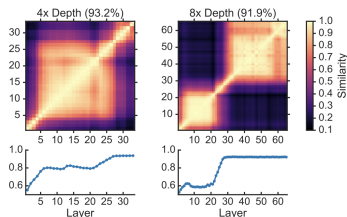


Figure: Diagnostic Tool for both individual and comparative DNNs

Student Teacher: Recent Generalization vs. Memorization

“Insights on representational similarity in neural networks with canonical correlation”

Morcos, Raghu, Bengio; <https://arxiv.org/pdf/1806.05759.pdf>

- Compare NN representations and how they evolve during training
- Projection weighted Canonical Correlation Analysis (PWCCA)

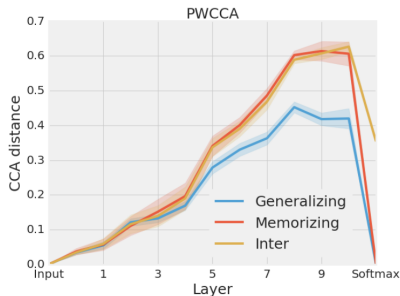


Figure: Generalizing networks converge to more similar solutions than memorizing networks.

Outline

1 Prehistory and History

- Older Background
- A Very Simple Deep Learning Model
- **More Immediate Background**

2 Preliminary Results

- Regularization and the Energy Landscape
- Preliminary Empirical Results
- Gaussian and Heavy-tailed Random Matrix Theory

3 Developing a Theory for Deep Learning

- More Detailed Empirical Results
- An RMT-based Theory for Deep Learning
- Tikhonov Regularization versus Heavy-tailed Regularization

4 Validating and Using the Theory

- Varying the Batch Size: Explaining the Generalization Gap
- Using the Theory: `pip install weightwatcher`
- Diagnostics at Scale: Predicting Test Accuracies

5 More General Implications and Conclusions

Motivations: Theoretical AND Practical

Theoretical: deeper insight into *Why Deep Learning Works?*

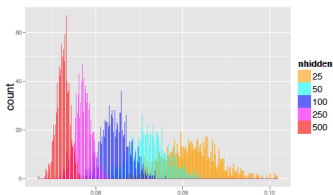
- convex versus non-convex optimization?
- explicit/implicit regularization?
- is / why is / when is deep better?
- VC theory versus Statistical Mechanics theory?
- ...

Practical: use insights to improve engineering of DNNs?

- when is a network fully optimized?
- can we use labels and/or domain knowledge more efficiently?
- large batch versus small batch in optimization?
- designing better ensembles?
- ...

Motivations: towards a Theory of Deep Learning

DNNs as spin glasses,
Choromanska et al. 2015



Looks exactly like old protein folding results (late 90s)

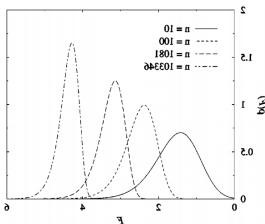
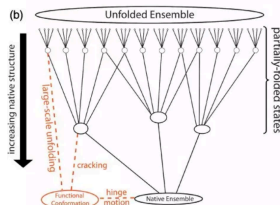
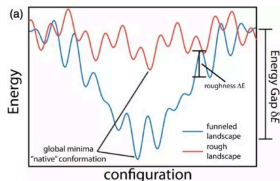


FIG. 1. Plot of the probability distribution $P(F)$ for different numbers of components n ($n = 10, 100, 1000, 10000$) calculated using the random matrix model.

Energy Landscape Theory



Energy Landscape of MultiScale Spin Glass Model

Completely different picture of DNNs

Raises broad questions about Why Deep Learning Works

Motivations: regularization in DNNs?

ICLR 2017 Best paper

- Large neural network models can easily overtrain/overfit on randomly labeled data
- Popular ways to regularize (basically $\min_x f(x) + \lambda g(x)$, with “control parameter” λ) may or may not help.

Understanding deep learning requires rethinking generalization??

<https://arxiv.org/abs/1611.03530>

Rethinking generalization requires revisiting old ideas: statistical mechanics approaches and complex learning behavior!!

<https://arxiv.org/abs/1710.09553> (Martin & Mahoney)

Motivations: stochastic optimization DNNs?

Theory (from convex problems):

- First order (SGD, e.g., Bottou 2010)
larger “batches” are better (at least up to statistical noise)
- Second order (SSN, e.g., Roosta and Mahoney 2016)
larger “batches” are better (at least up to statistical noise)
- *Large batch sizes have better computational properties!*

So, people just increase batch size (and compensate with other parameters)

Practice (from non-convex problems):

- SGD-like methods “saturate”
(<https://arxiv.org/abs/1811.12941>)
- SSN-like methods “saturate”
(<https://arxiv.org/abs/1903.06237>)
- *Small batch sizes have better statistical properties!*

Is batch size a computational parameter, or a statistical parameter, or what?

How should batch size be chosen?

Set up: the Energy Landscape

Energy/Optimization function:

$$E_{DNN} = h_L(\mathbf{W}_L \times h_{L-1}(\mathbf{W}_{L-1} \times h_{L-2}(\dots) + \mathbf{b}_{L-1}) + \mathbf{b}_L)$$

Train this on labeled data $\{d_i, y_i\} \in \mathcal{D}$, using Backprop, by minimizing loss \mathcal{L} :

$$\min_{W_i, b_i} \mathcal{L} \left(\sum_i E_{DNN}(d_i) - y_i \right)$$

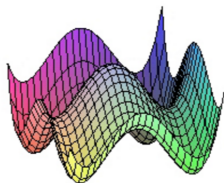
E_{DNN} is “the” *Energy Landscape*:

- The part of the optimization problem parameterized by the heretofore unknown elements of the weight matrices and bias vectors, and as defined by the data $\{d_i, y_i\} \in \mathcal{D}$
- Pass the data through the Energy function E_{DNN} multiple times, as we run Backprop training
- The Energy Landscape[§] is *changing* at each epoch

[§]i.e., the optimization function that is *nominally* being optimized

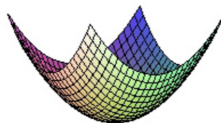
Problem: How can this possibly work?

Expected



Highly non-convex?

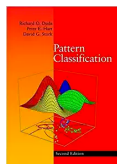
Observed



Apparently not!

It has been known for a long time that local minima are not the issue.

Problem: Local Minima?

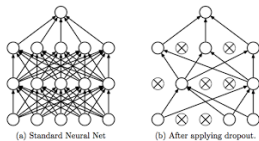


Duda, Hart and Stork, 2000

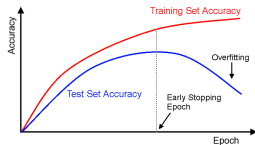
Whereas in low-dimensional spaces, **local minima** can be plentiful, in high dimension, the problem of **local minima** is different: The high-dimensional space may afford more ways (dimensions) for the system to “get around” a barrier or **local maximum** during learning. The more superfluous the weights, the less likely it is a network will get trapped in **local minima**. However, networks with an unnecessarily large number of weights are undesirable because of the dangers of overfitting, as we shall see in Section 6.11.

Solution: add more capacity and regularize, i.e., over-parameterization

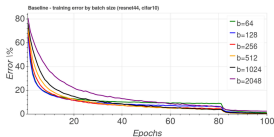
Motivations: what is regularization?



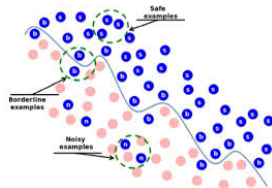
(a) Dropout.



(b) Early Stopping.



(c) Batch Size.



(d) Noisify Data.

Every adjustable *knob* and *switch*—and there are *many*[¶]—is regularization.

[¶]<https://arxiv.org/pdf/1710.10686.pdf>

Outline

1 Prehistory and History

- Older Background
- A Very Simple Deep Learning Model
- More Immediate Background

2 Preliminary Results

- Regularization and the Energy Landscape
- Preliminary Empirical Results
- Gaussian and Heavy-tailed Random Matrix Theory

3 Developing a Theory for Deep Learning

- More Detailed Empirical Results
- An RMT-based Theory for Deep Learning
- Tikhonov Regularization versus Heavy-tailed Regularization

4 Validating and Using the Theory

- Varying the Batch Size: Explaining the Generalization Gap
- Using the Theory: `pip install weightwatcher`
- Diagnostics at Scale: Predicting Test Accuracies

5 More General Implications and Conclusions

Outline

1 Prehistory and History

- Older Background
- A Very Simple Deep Learning Model
- More Immediate Background

2 Preliminary Results

- Regularization and the Energy Landscape
- Preliminary Empirical Results
- Gaussian and Heavy-tailed Random Matrix Theory

3 Developing a Theory for Deep Learning

- More Detailed Empirical Results
- An RMT-based Theory for Deep Learning
- Tikhonov Regularization versus Heavy-tailed Regularization

4 Validating and Using the Theory

- Varying the Batch Size: Explaining the Generalization Gap
- Using the Theory: `pip install weightwatcher`
- Diagnostics at Scale: Predicting Test Accuracies

5 More General Implications and Conclusions

Basics of Regularization

Ridge Regression / Tikhonov-Phillips Regularization

$$\hat{\mathbf{W}}\mathbf{x} = \mathbf{y}$$

$$\mathbf{x} = \left(\hat{\mathbf{W}}^T \hat{\mathbf{W}} + \alpha I \right)^{-1} \hat{\mathbf{W}}^T \mathbf{y} \quad \left\{ \begin{array}{l} \text{Moore-Penrose pseudoinverse (1955)} \\ \text{Ridge regularization (Phillips, 1962)} \end{array} \right.$$

$$\min_{\mathbf{x}} \|\hat{\mathbf{W}}\mathbf{x} - \mathbf{y}\|_2^2 + \alpha \|\hat{\mathbf{x}}\|_2^2$$

familiar optimization problem

Softens the rank of $\hat{\mathbf{W}}$ to focus on large eigenvalues.

Related to Truncated SVD, which does hard truncation on rank of $\hat{\mathbf{W}}$

Early stopping, truncated random walks, etc. often implicitly solve regularized optimization problems.

How we will study regularization

The Energy Landscape is *determined* by layer weight matrices \mathbf{W}_L :

$$E_{DNN} = h_L(\mathbf{W}_L \times h_{L-1}(\mathbf{W}_{L-1} \times h_{L-2}(\cdots) + \mathbf{b}_{L-1}) + \mathbf{b}_L)$$

Traditional regularization is applied to \mathbf{W}_L :

$$\min_{W_l, b_l} \mathcal{L} \left(\sum_i E_{DNN}(d_i) - y_i \right) + \alpha \sum_l \|\mathbf{W}_l\|$$

Different types of regularization, e.g., different norms $\|\cdot\|$, leave different empirical signatures on \mathbf{W}_L .

What we do:

- Turn off “all” regularization.
- Systematically turn it back on, explicitly with α or implicitly with knobs/switches.
- **Study empirical properties of \mathbf{W}_L .**

Lots of DNNs Analyzed

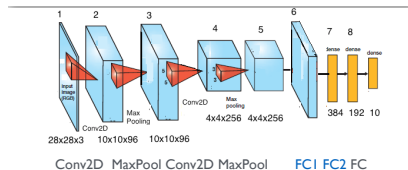
Question: What happens to the layer weight matrices \mathbf{W}_L ?

(Don't evaluate your method on one/two/three NN, evaluate it on a dozen/hundred.)

Retrained LeNet5 on MNIST using Keras.

Two other small models:

- 3-Layer MLP
- Mini AlexNet



Wide range of state-of-the-art pre-trained models:

- AlexNet, Inception, etc.

Outline

1 Prehistory and History

- Older Background
- A Very Simple Deep Learning Model
- More Immediate Background

2 Preliminary Results

- Regularization and the Energy Landscape
- **Preliminary Empirical Results**
- Gaussian and Heavy-tailed Random Matrix Theory

3 Developing a Theory for Deep Learning

- More Detailed Empirical Results
- An RMT-based Theory for Deep Learning
- Tikhonov Regularization versus Heavy-tailed Regularization

4 Validating and Using the Theory

- Varying the Batch Size: Explaining the Generalization Gap
- Using the Theory: `pip install weightwatcher`
- Diagnostics at Scale: Predicting Test Accuracies

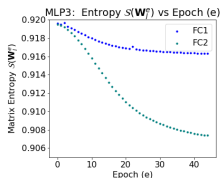
5 More General Implications and Conclusions

Matrix complexity: Matrix Entropy and Stable Rank

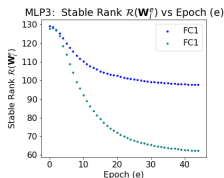
$$\mathbf{W} = \mathbf{U}\mathbf{\Sigma}\mathbf{V}^T \quad \nu_i = \Sigma_{ii} \quad p_i = \nu_i^2 / \sum_i \nu_i^2$$

$$\mathcal{S}(\mathbf{W}) = \frac{-1}{\log(R(\mathbf{W}))} \sum_i p_i \log p_i \quad \mathcal{R}_s(\mathbf{W}) = \frac{\|\mathbf{W}\|_F^2}{\|\mathbf{W}\|_2^2} = \frac{\sum_i \nu_i^2}{\nu_{\max}^2}$$

A warm-up: train a 3-Layer MLP:



(e) MLP3 Entropies.



(f) MLP3 Stable Ranks.

Figure: Matrix Entropy & Stable Rank show transition during Backprop training.

Matrix complexity: Scree Plots

$$\mathbf{W} = \mathbf{U}\Sigma\mathbf{V}^T$$

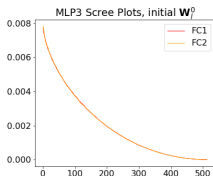
$$\nu_i = \Sigma_{ii}$$

$$p_i = \nu_i^2 / \sum_i \nu_i^2$$

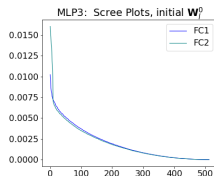
$$\mathcal{S}(\mathbf{W}) = \frac{-1}{\log(R(\mathbf{W}))} \sum_i p_i \log p_i$$

$$\mathcal{R}_s(\mathbf{W}) = \frac{\|\mathbf{W}\|_F^2}{\|\mathbf{W}\|_2^2} = \frac{\sum_i \nu_i^2}{\nu_{\max}^2}$$

A warm-up: train a 3-Layer MLP:



(a) Initial Scree Plot.



(b) Final Scree Plot.

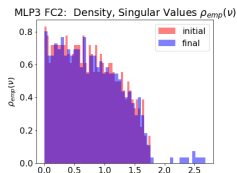
Figure: Scree plots for initial and final configurations for MLP3.

Matrix complexity: Singular/Eigen Value Densities

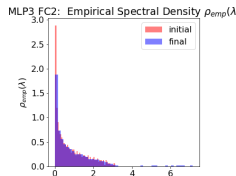
$$\mathbf{W} = \mathbf{U}\mathbf{\Sigma}\mathbf{V}^T \quad \nu_i = \Sigma_{ii} \quad \rho_i = \nu_i^2 / \sum_i \nu_i^2$$

$$\mathcal{S}(\mathbf{W}) = \frac{-1}{\log(R(\mathbf{W}))} \sum_i p_i \log p_i \quad \mathcal{R}_s(\mathbf{W}) = \frac{\|\mathbf{W}\|_F^2}{\|\mathbf{W}\|_2^2} = \frac{\sum_i \nu_i^2}{\nu_{\max}^2}$$

A warm-up: train a 3-Layer MLP:



(a) Singular val. density



(b) Eigenvalue density

Figure: Histograms of the Singular Values ν_i and associated Eigenvalues $\lambda_i = \nu_i^2$.

ESD: detailed insight into W_L

Empirical Spectral Density (ESD: eigenvalues of $X = \mathbf{W}_L^T \mathbf{W}_L$)

```
import keras

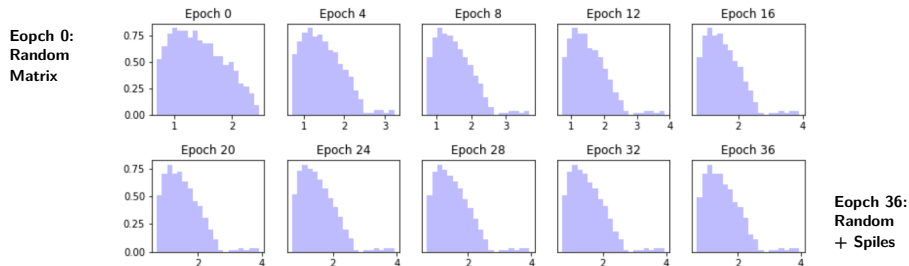
import numpy as np
import matplotlib.pyplot as plt

...
W = model.layers[i].get_weights()[0]
...
X = np.dot(W, W.T)
evals, evecs = np.linalg.eig(W, W.T)

plt.hist(X, bin=100, density=True)
```


ESD: detailed insight into W_L

Empirical Spectral Density (ESD: eigenvalues of $X = \mathbf{W}_L^T \mathbf{W}_L$)



Entropy decrease corresponds to:

- modification (later, breakdown) of random structure and
- onset of a new kind of self-regularization.

Outline

1 Prehistory and History

- Older Background
- A Very Simple Deep Learning Model
- More Immediate Background

2 Preliminary Results

- Regularization and the Energy Landscape
- Preliminary Empirical Results
- Gaussian and Heavy-tailed Random Matrix Theory

3 Developing a Theory for Deep Learning

- More Detailed Empirical Results
- An RMT-based Theory for Deep Learning
- Tikhonov Regularization versus Heavy-tailed Regularization

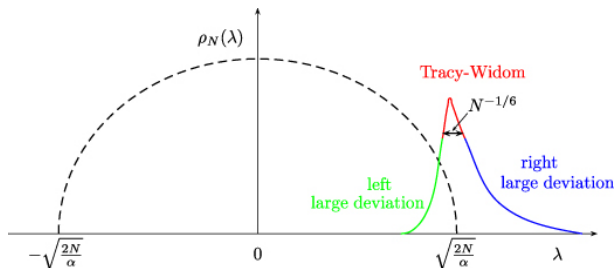
4 Validating and Using the Theory

- Varying the Batch Size: Explaining the Generalization Gap
- Using the Theory: `pip install weightwatcher`
- Diagnostics at Scale: Predicting Test Accuracies

5 More General Implications and Conclusions

Random Matrix Theory 101: Wigner and Tracy-Widom

- Wigner: *global bulk statistics* approach universal semi-circular form
- Tracy-Widom: *local edge statistics* fluctuate in universal way



Problems with Wigner and Tracy-Widom:

- Weight matrices usually not square
- Typically do only a single training run

Random Matrix Theory 102: Marchenko-Pastur

Let \mathbf{W} be an $N \times M$ random matrix, with elements $W_{ij} \sim N(0, \sigma_{mp}^2)$.

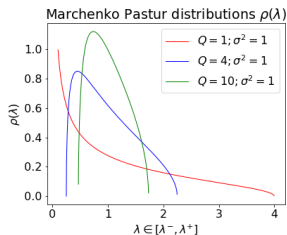
Then, the ESD of $\mathbf{X} = \mathbf{W}^T \mathbf{W}$, converges to a deterministic function:

$$\rho_N(\lambda) \quad := \quad \frac{1}{N} \sum_{i=1}^M \delta(\lambda - \lambda_i)$$
$$\xrightarrow[\substack{N \rightarrow \infty \\ Q \text{ fixed}}]{} \begin{cases} \frac{Q}{2\pi\sigma_{mp}^2} \frac{\sqrt{(\lambda^+ - \lambda)(\lambda - \lambda^-)}}{\lambda} & \text{if } \lambda \in [\lambda^-, \lambda^+] \\ 0 & \text{otherwise.} \end{cases}$$

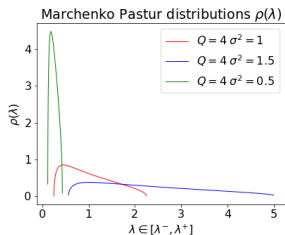
with well-defined edges (which depend on Q , the aspect ratio):

$$\lambda^\pm = \sigma_{mp}^2 \left(1 \pm \frac{1}{\sqrt{Q}} \right)^2 \quad Q = N/M \geq 1.$$

Random Matrix Theory 102': Marchenko-Pastur



(a) Vary aspect ratios



(b) Vary variance parameters

Figure: Marchenko-Pastur (MP) distributions.

Important points:

- *Global bulk stats*: The overall shape is deterministic, fixed by Q and σ .
- *Local edge stats*: The edge λ^+ is very crisp, i.e., $\Delta\lambda_M = |\lambda_{max} - \lambda^+| \sim O(M^{-2/3})$, plus Tracy-Widom fluctuations.

We use both *global bulk statistics* as well as *local edge statistics* in our theory.

Random Matrix Theory 103: Heavy-tailed RMT

Go beyond the (relatively easy) Gaussian Universality class:

- *model* strongly-correlated systems (“signal”) with heavy-tailed random matrices.

	Generative Model w/ elements from Universality class	Finite- N Global shape $\rho_N(\lambda)$	Limiting Global shape $\rho(\lambda), N \rightarrow \infty$	Bulk edge Local stats $\lambda \approx \lambda^+$	(far) Tail Local stats $\lambda \approx \lambda_{max}$
Basic MP	Gaussian	MP distribution	MP	TW	No tail.
Spiked- Covariance	Gaussian, + low-rank perturbations	MP + Gaussian spikes	MP	TW	Gaussian
Heavy tail, $4 < \mu$	(Weakly) Heavy-Tailed	MP + PL tail	MP	Heavy-Tailed*	Heavy-Tailed*
Heavy tail, $2 < \mu < 4$	(Moderately) Heavy-Tailed (or “fat tailed”)	PL** $\sim \lambda^{-(a\mu+b)}$	PL $\sim \lambda^{-(\frac{1}{2}\mu+1)}$	No edge.	Frechet
Heavy tail, $0 < \mu < 2$	(Very) Heavy-Tailed	PL** $\sim \lambda^{-(\frac{1}{2}\mu+1)}$	PL $\sim \lambda^{-(\frac{1}{2}\mu+1)}$	No edge.	Frechet

Basic MP theory, and the spiked and Heavy-Tailed extensions we use, including known, empirically-observed, and conjectured relations between them. Boxes marked “*” are best described as following “TW with large finite size corrections” that are likely Heavy-Tailed, leading to bulk edge statistics and far tail statistics that are indistinguishable. Boxes marked “**” are phenomenological fits, describing large ($2 < \mu < 4$) or small ($0 < \mu < 2$) finite-size corrections on $N \rightarrow \infty$ behavior.

Fitting Heavy-tailed Distributions

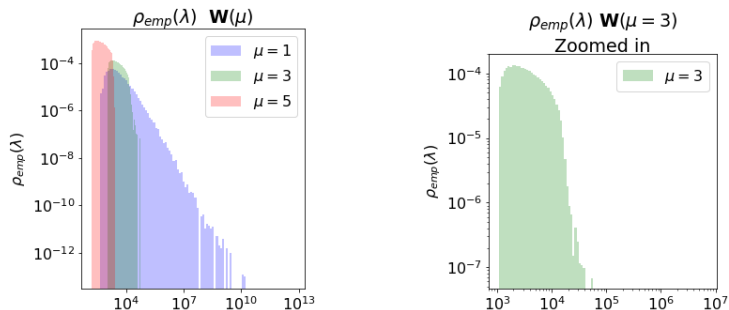
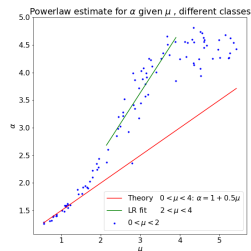
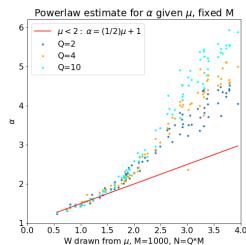


Figure: The log-log histogram plots of the ESD for three Heavy-Tailed random matrices \mathbf{M} with same aspect ratio $Q = 3$, with $\mu = 1.0, 3.0, 5.0$, corresponding to the three Heavy-Tailed Universality classes ($0 < \mu < 2$ vs $2 < \mu < 4$ and $4 < \mu$).

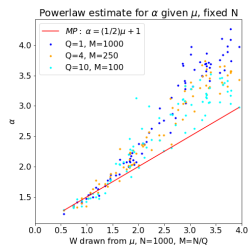
Non-negligible finite size effects



(a) $M = 1000, N = 2000$.



(b) Fixed M .



(c) Fixed N .

Figure: Dependence of α (the fitted PL parameter) on μ (the hypothesized limiting PL parameter).

Heavy Tails (!) and Heavy-Tailed Universality (?)

Universality: large-scale properties are independent of small-scale details

- Mathematicians: justify proving theorems in random matrix theory
- Physicists: derive new phenomenological relations and predict things
- Gaussian Universality is most common, but there are many other types.

Heavy-Tailed Phenomenon

- Rare events are not extraordinarily rare, i.e., are heavier than Gaussian tails
- Modeled with power law and related functions
- Seen in finance, structural glass theory, etc.

Heavy-Tailed Random Matrix Theory

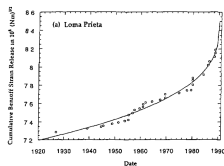
- Phenomenological work by physicists (Bouchard, Potters, Sornette, 90s)
- Theorem proving by mathematicians (Auffinger, Ben Arous, Burda, Peche, 00s)
- Universality of Power Laws, Levy-based dynamics, finite-size attractors, etc.

Heavy-Tailed Universality: Earthquake prediction

"Complex Critical Exponents from Renormalization Group Theory of Earthquakes ..." Sornette et al. (1985)

Power law fit^{||} of the regional strain ϵ (a measure of seismic release) before the critical time t_c (of the earthquake)

$$\frac{d\epsilon}{dt} = A + B(t - t_c)^m$$



(a)

Table I. — Parameters found by fitting time-to-failure equations to the cumulative Benioff strain.

Parameters	Loma Prieta	Kommandorski Island
Power fit		
Equation (2)		
A	8.50 ± 0.73	6.23 ± 26.9
B	-0.29 ± 0.44	-2.49 ± 19.3
m	0.35 ± 0.23	0.26 ± 1.0
t_c	1990.3 ± 4.1	1998.8 ± 19.7

(b)

Figure: (a) Cumulative Benioff strain released by magnitude 5 and greater earthquakes in the San Francisco Bay area prior to the 1989 Loma Prieta earthquake. (b) Fit of Power Law exponent (m).

^{||} More sophisticated Renormalization Group (RG) analysis uses complex critical exponents, giving log-perpodic corrections.

Heavy-Tailed Universality: Market Crashes

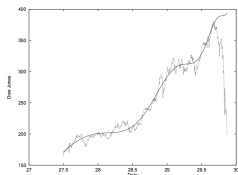
"Why Stock Markets Crash: Critical Events in Complex Financial Systems" by D. Sornette (book, 2003)

Simple Power Law

$$\log p(t) = A + B(t - t_c)^\beta$$

Complex Power Law (RG Log Periodic corrections)

$$\log p(t) = A + B(t - t_c)^\beta + C(t - t_c)^\beta (\cos(\omega \log(t - t_c) - \phi))$$



(a) Dow Jones 1929 crash

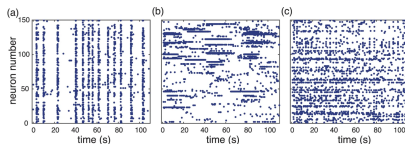
crash	t_c	t_{max}	t_{min}	drop	β	ω_1	λ	A	B	C	Var
1929 (WS)	30.22	29.65	29.87	47%	0.45	7.9	2.2	571	-267	14.3	56
1985 (DEM)	85.20	85.15	85.30	14%	0.28	6.0	2.8	3.88	1.16	-0.77	0.0028
1985 (CHF)	85.19	85.18	85.30	15%	0.36	5.2	3.4	3.10	-0.86	-0.055	0.0012
1987 (WS)	87.74	87.65	87.80	30%	0.33	7.4	2.3	411	-165	12.2	36
1997 (H-K)	97.74	97.60	97.82	46%	0.34	7.5	2.3	20077	-8241	-397	190360
1998 (WS)	98.72	98.55	98.67	19%	0.60	6.4	2.7	1321	-402	19.7	375
1998 (YEN)	98.78	98.61	98.77	21%	0.19	7.2	2.4	207	-84.5	2.78	17
1998 (CAN\$)	98.66	98.66	98.71	5.1%	0.26	8.2	2.2	1.62	-0.23	-0.011	0.00024

(b) Universal parameters, fit to RG model

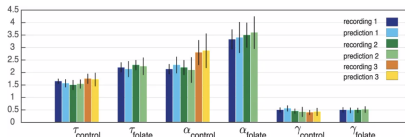
Heavy-Tailed Universality: Neuronal Avalanches

Neuronal avalanche dynamics indicates different universality classes in neuronal cultures; Scientific Reports 3417 (2018)

From: Neuronal avalanche dynamics indicates different universality classes in neuronal cultures



(c) Spiking activity of cultured neurons



(d) Critical exponents, fit to scaling model

Outline

1 Prehistory and History

- Older Background
- A Very Simple Deep Learning Model
- More Immediate Background

2 Preliminary Results

- Regularization and the Energy Landscape
- Preliminary Empirical Results
- Gaussian and Heavy-tailed Random Matrix Theory

3 Developing a Theory for Deep Learning

- More Detailed Empirical Results
- An RMT-based Theory for Deep Learning
- Tikhonov Regularization versus Heavy-tailed Regularization

4 Validating and Using the Theory

- Varying the Batch Size: Explaining the Generalization Gap
- Using the Theory: `pip install weightwatcher`
- Diagnostics at Scale: Predicting Test Accuracies

5 More General Implications and Conclusions

Outline

1 Prehistory and History

- Older Background
- A Very Simple Deep Learning Model
- More Immediate Background

2 Preliminary Results

- Regularization and the Energy Landscape
- Preliminary Empirical Results
- Gaussian and Heavy-tailed Random Matrix Theory

3 Developing a Theory for Deep Learning

- **More Detailed Empirical Results**
- An RMT-based Theory for Deep Learning
- Tikhonov Regularization versus Heavy-tailed Regularization

4 Validating and Using the Theory

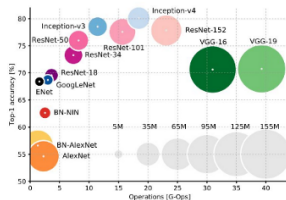
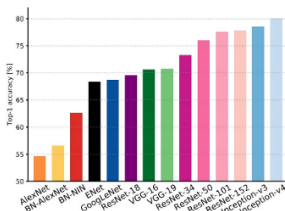
- Varying the Batch Size: Explaining the Generalization Gap
- Using the Theory: `pip install weightwatcher`
- Diagnostics at Scale: Predicting Test Accuracies

5 More General Implications and Conclusions

Experiments: just apply this to pre-trained models

https://medium.com/@siddharthdas_32104/cnns-architectures-lexnet-alexnet-vgg-googlenet-resnet-and-more-...

Year	CNN	Developed by	Place	Top-5 error rate	No. of parameters
1998	LeNet(8)	Yann LeCun et al			60 thousand
2012	AlexNet(7)	Alex Krizhevsky, Geoffrey Hinton, Ilya Sutskever	1st	15.3%	60 million
2013	ZFNet()	Matthew Zeiler and Rob Fergus	1st	14.8%	
2014	GoogLeNet(19)	Google	1st	6.67%	4 million
2014	VGG Net(16)	Simonyan, Zisserman	2nd	7.3%	138 million
2015	ResNet(152)	Kaiming He	1st	3.6%	



An Analysis of Deep Neural Network Models for Practical Applications, 2017.

Experiments: just apply this to pre-trained models

Model	Layer	Q	$(M \times N)$	α	D	Best Fit
alexnet	17/FC1	2.25	(4096 × 9216)	2.29	0.0527	PL
	20/FC2	1	(4096 × 4096)	2.25	0.0372	PL
	22/FC3	4.1	(1000 × 4096)	3.02	0.0186	PL
densenet121	432	1.02	(1000 × 1024)	3.32	0.0383	PL
densenet121	432	1.02	(1000 × 1024)	3.32	0.0383	PL
densenet161	572	2.21	(1000 × 2208)	3.45	0.0322	PL
densenet169	600	1.66	(1000 × 1664)	3.38	0.0396	PL
densenet201	712	1.92	(1000 × 1920)	3.41	0.0332	PL
inception v3	L226	1.3	(768 × 1000)	5.26	0.0421	PL
	L302	2.05	(1000 × 2048)	4.48	0.0275	PL
resnet101	286	2.05	(1000 × 2048)	3.57	0.0278	PL
resnet152	422	2.05	(1000 × 2048)	3.52	0.0298	PL
resnet18	67	1.95	(512 × 1000)	3.34	0.0342	PL
resnet34	115	1.95	(512 × 1000)	3.39	0.0257	PL
resnet50	150	2.05	(1000 × 2048)	3.54	0.027	PL
vgg11	24	6.12	(4096 × 25088)	2.32	0.0327	PL
	27	1	(4096 × 4096)	2.17	0.0309	TPL
	30	4.1	(1000 × 4096)	2.83	0.0398	PL
vgg11 bn	32	6.12	(4096 × 25088)	2.07	0.0311	TPL
	35	1	(4096 × 4096)	1.95	0.0336	TPL
	38	4.1	(1000 × 4096)	2.99	0.0339	PL
vgg16	34	6.12	(4096 × 25088)	2.3	0.0277	PL
	37	1	(4096 × 4096)	2.18	0.0321	TPL
	40	4.1	(1000 × 4096)	2.09	0.0403	TPL
vgg16 bn	47	6.12	(4096 × 25088)	2.05	0.0285	TPL
	50	1	(4096 × 4096)	1.97	0.0363	TPL
	53	4.1	(1000 × 4096)	3.03	0.0358	PL
vgg19	40	6.12	(4096 × 25088)	2.27	0.0247	PL
	43	1	(4096 × 4096)	2.19	0.0313	PL
	46	4.1	(1000 × 4096)	2.07	0.0368	TPL
vgg19 bn	56	6.12	(4096 × 25088)	2.04	0.0295	TPL
	59	1	(4096 × 4096)	1.98	0.0373	TPL
	62	4.1	(1000 × 4096)	3.03	0.035	PL

RMT: LeNet5 (an old/small NN example)

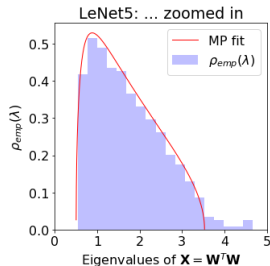
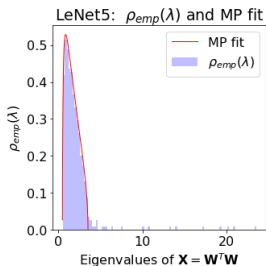
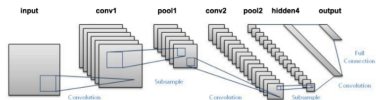


Figure: Full and zoomed-in ESD for LeNet5, Layer FC1.

Marchenko-Pastur Bulk + Spikes

RMT: AlexNet (a typical modern/large DNN example)

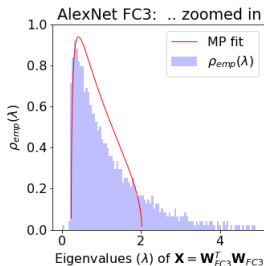
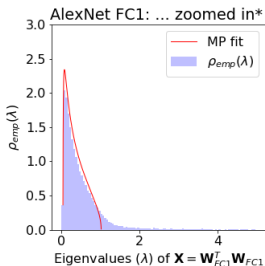
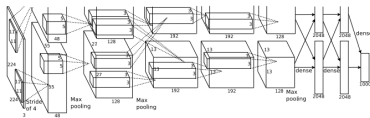


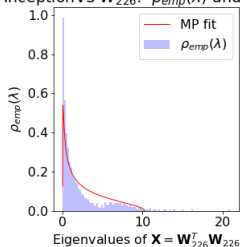
Figure: Zoomed-in ESD for Layer FC1 and FC3 of AlexNet.

Marchenko-Pastur Bulk-decay + Heavy-tailed

RMT: InceptionV3 (a particularly unusual example)



InceptionV3 W_{226} : $\rho_{emp}(\lambda)$ and MP fi



InceptionV3 W_{302} : $\rho_{emp}(\lambda)$ and MP fi

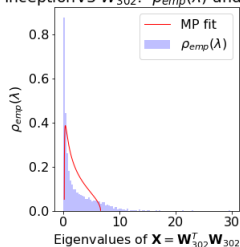
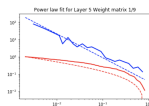


Figure: ESD for Layers L226 and L302 in InceptionV3, as distributed w/ pyTorch.

Marchenko-Pastur bulk decay, onset of Heavy Tails

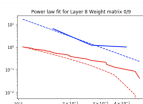
Convolutional 2D Feature Maps

We analyze Conv2D layers by extracting the feature maps individually, i.e., A $(3 \times 3 \times 64 \times 64)$ Conv2D layer yields 9 (64×64) Feature Maps



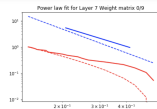
ESD (Empirical Spectral Density) $p(x)$ for Layer 5 Weight matrix 1/9, $\alpha=1.38$

(a) $\alpha = 1.38$



ESD (Empirical Spectral Density) $p(x)$ for Layer 8 Weight matrix 0/9, $\alpha=2.74$

(b) $\alpha = 2.74$



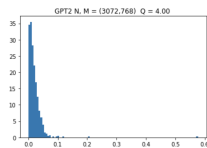
ESD (Empirical Spectral Density) $p(x)$ for Layer 7 Weight matrix 0/9, $\alpha=3.02$

(c) $\alpha = 3.02$

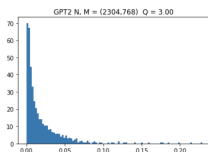
Figure: Select Feature Maps from different Conv2D layers of VGG16. Fits shown with PDF (blue) and CDF (red)

Open AI GPT2 Attention Matrices

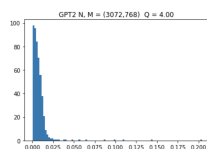
NLP Embedding and Attention Matrices are Dense/Linear, but generally have large aspect ratios



(a) $\alpha =$



(b) $\alpha =$



(c) $\alpha =$

Figure: Selected ESDs from Open AI GPT2 (Huggingface implementation)

Outline

1 Prehistory and History

- Older Background
- A Very Simple Deep Learning Model
- More Immediate Background

2 Preliminary Results

- Regularization and the Energy Landscape
- Preliminary Empirical Results
- Gaussian and Heavy-tailed Random Matrix Theory

3 Developing a Theory for Deep Learning

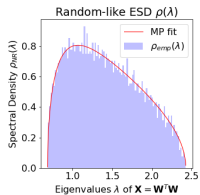
- More Detailed Empirical Results
- **An RMT-based Theory for Deep Learning**
- Tikhonov Regularization versus Heavy-tailed Regularization

4 Validating and Using the Theory

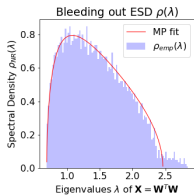
- Varying the Batch Size: Explaining the Generalization Gap
- Using the Theory: `pip install weightwatcher`
- Diagnostics at Scale: Predicting Test Accuracies

5 More General Implications and Conclusions

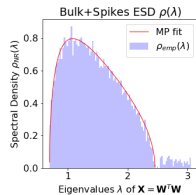
RMT-based 5+1 Phases of Training



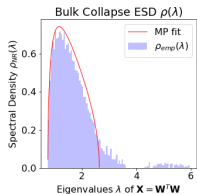
(a) RANDOM-LIKE.



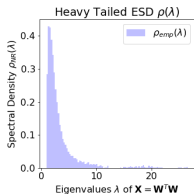
(b) BLEEDING-OUT.



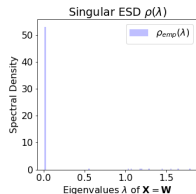
(c) BULK+SPIKES.



(d) BULK-DECAY.



(e) HEAVY-TAILED.



(f) RANK-COLLAPSE.

Figure: The 5+1 phases of learning we identified in DNN training.

RMT-based 5+1 Phases of Training

We *model* “noise” and also “signal” with random matrices:

$$\mathbf{W} \simeq \mathbf{W}^{rand} + \Delta^{sig}. \quad (1)$$

	Operational Definition	Informal Description via Eqn. (1)	Edge/tail Fluctuation Comments	Illustration and Description
RANDOM-LIKE	ESD well-fit by MP with appropriate λ^+	\mathbf{W}^{rand} random; $\ \Delta^{sig}\ $ zero or small	$\lambda_{max} \approx \lambda^+$ is sharp, with TW statistics	Fig. 15(a)
BLEEDING-OUT	ESD RANDOM-LIKE, excluding eigenmass just above λ^+	\mathbf{W} has eigenmass at bulk edge as spikes “pull out”; $\ \Delta^{sig}\ $ medium	BPP transition, λ_{max} and λ^+ separate	Fig. 15(b)
BULK+SPIKES	ESD RANDOM-LIKE plus ≥ 1 spikes well above λ^+	\mathbf{W}^{rand} well-separated from low-rank Δ^{sig} ; $\ \Delta^{sig}\ $ larger	λ^+ is TW, λ_{max} is Gaussian	Fig. 15(c)
BULK-DECAY	ESD less RANDOM-LIKE; Heavy-Tailed eigenmass above λ^+ ; some spikes	Complex Δ^{sig} with correlations that don't fully enter spike	Edge above λ^+ is not concave	Fig. 15(d)
HEAVY-TAILED	ESD better-described by Heavy-Tailed RMT than Gaussian RMT	\mathbf{W}^{rand} is small; Δ^{sig} is large and strongly-correlated	No good λ^+ ; $\lambda_{max} \gg \lambda^+$	Fig. 15(e)
RANK-COLLAPSE	ESD has large-mass spike at $\lambda = 0$	\mathbf{W} very rank-deficient; over-regularization	—	Fig. 15(f)

The 5+1 phases of learning we identified in DNN training.

RMT-based 5+1 Phases of Training

Lots of technical issues ...

Outline

1 Prehistory and History

- Older Background
- A Very Simple Deep Learning Model
- More Immediate Background

2 Preliminary Results

- Regularization and the Energy Landscape
- Preliminary Empirical Results
- Gaussian and Heavy-tailed Random Matrix Theory

3 Developing a Theory for Deep Learning

- More Detailed Empirical Results
- An RMT-based Theory for Deep Learning
- **Tikhonov Regularization versus Heavy-tailed Regularization**

4 Validating and Using the Theory

- Varying the Batch Size: Explaining the Generalization Gap
- Using the Theory: `pip install weightwatcher`
- Diagnostics at Scale: Predicting Test Accuracies

5 More General Implications and Conclusions

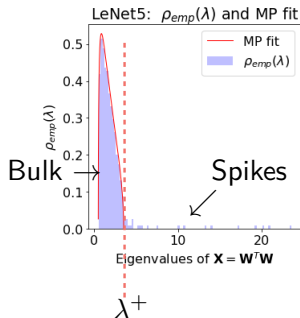
Bulk+Spikes: Small Models \sim Tikhonov regularization

Low-rank perturbation

$$\mathbf{W}_l \simeq \mathbf{W}_l^{rand} + \Delta^{large}$$

Perturbative correction

$$\lambda_{max} = \sigma^2 \left(\frac{1}{Q} + \frac{|\Delta|^2}{N} \right) \left(1 + \frac{N}{|\Delta|^2} \right)$$
$$|\Delta| > (Q)^{-\frac{1}{4}}$$



simple scale threshold

$$\mathbf{x} = \left(\hat{\mathbf{X}} + \alpha \mathbf{I} \right)^{-1} \hat{\mathbf{W}}^T \mathbf{y}$$

eigenvalues $> \alpha$ (Spikes)
carry most of the
signal/information

Smaller, older models like LeNet5 exhibit traditional regularization and can be described perturbatively with Gaussian RMT

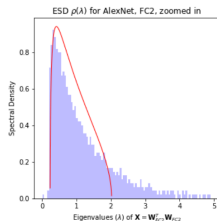
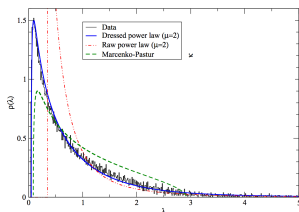
Heavy-tailed Self-regularization

\mathbf{W} is *strongly-correlated* and highly non-random

- We *model* strongly-correlated systems by heavy-tailed random matrices
- I.e., we *model* signal (not noise) by heavy-tailed random matrices

Then RMT/MP ESD will also have heavy tails

Known results from RMT / polymer theory (Bouchaud, Potters, etc)



AlexNet
ReseNet50
Inception V3
DenseNet201

...

“All” larger, modern DNNs exhibit novel Heavy-tailed self-regularization

Heavy-tailed Self-regularization

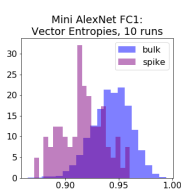
Summary of what we “suspect” today

- No single scale threshold.
- No simple low rank approximation for \mathbf{W}_L .
- Contributions from correlations at all scales.
- Can *not* be treated perturbatively.

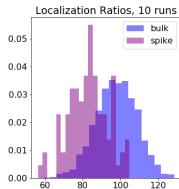
“All” larger, modern DNNs exhibit novel Heavy-tailed self-regularization

Spikes: carry more “information” than the Bulk

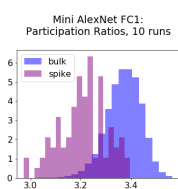
Spikes have less entropy, are more localized than bulk.



(a) Vector Entropies.



(b) Localization Ratios.



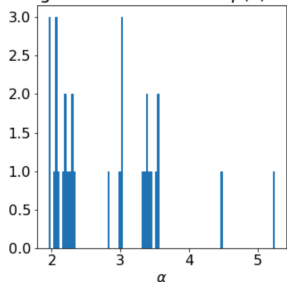
(c) Participation Ratios.

Figure: Eigenvector localization metrics for the FC1 layer of MiniAlexNet.

Information begins to concentrate in the spikes.

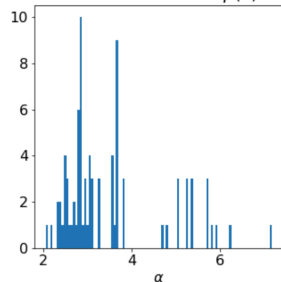
Power Law Universality: ImageNet and AllenNLP

ImageNet Power Law fits: $p(\lambda) \sim \lambda^{-\alpha}$



(a) ImageNet pyTorch models

AllenNLP Power Law fits: $p(\lambda) \sim \lambda^{-\alpha}$

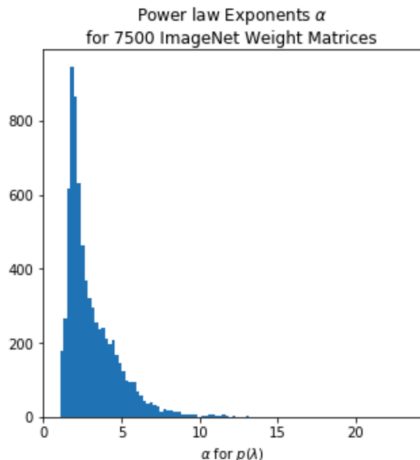


(b) AllenNLP models

Figure 12: Distribution of power law exponents α for linear layers in pre-trained models trained on ImageNet, available in pyTorch, and for those NLP models, available in AllenNLP.

All these models display remarkable Heavy Tailed Universality

Power Law Universality: ImageNet

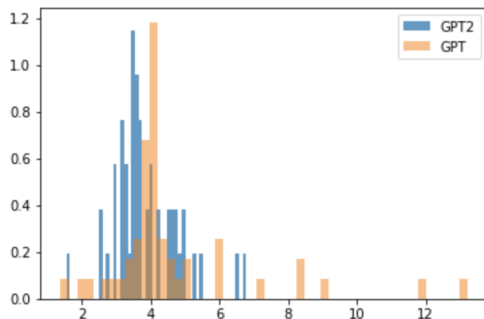


- 7500 matrices (and Conv2D feature maps)
- over 50 architectures
- Linear layers and Conv2D feature maps
- 80 – 90% < 5

All these models display remarkable Heavy Tailed Universality

Power Law Universality: Open AI GPT versus GPT2

GPT and GPT2 Layer Weight Matrix
Power Law Exponents α , $\rho(\lambda) \sim \lambda^{-\alpha}$



GPT versus GPT2: (Huggingface implementation)
example of a class of models that “improves” over time.

Mechanisms?

Spiked-Covariance Model

- Statistical model: Johnstone, “On the distribution . . .” 2001.
- Simple self-organization model: Malevergne and Sornette, “Collective Origin of the Coexistence of Apparent RMT Noise and Factors in Large Sample Correlation Matrices,” 2002.
- Low-rank perturbative variant of Gaussian randomness modeling noise

Heavy-tailed Models: Self-organized criticality (and others ...)

- Johansen, Sornette, and Ledoit, “Predicting financial crashes using discrete scale invariance,” 1998.
- Markovic and Gros, “Power laws and self-organized criticality in theory and nature,” 2013.
- Non-perturbative model where heavy-tails and power laws are used to model strongly correlated systems

Outline

1 Prehistory and History

- Older Background
- A Very Simple Deep Learning Model
- More Immediate Background

2 Preliminary Results

- Regularization and the Energy Landscape
- Preliminary Empirical Results
- Gaussian and Heavy-tailed Random Matrix Theory

3 Developing a Theory for Deep Learning

- More Detailed Empirical Results
- An RMT-based Theory for Deep Learning
- Tikhonov Regularization versus Heavy-tailed Regularization

4 Validating and Using the Theory

- Varying the Batch Size: Explaining the Generalization Gap
- Using the Theory: `pip install weightwatcher`
- Diagnostics at Scale: Predicting Test Accuracies

5 More General Implications and Conclusions

Outline

1 Prehistory and History

- Older Background
- A Very Simple Deep Learning Model
- More Immediate Background

2 Preliminary Results

- Regularization and the Energy Landscape
- Preliminary Empirical Results
- Gaussian and Heavy-tailed Random Matrix Theory

3 Developing a Theory for Deep Learning

- More Detailed Empirical Results
- An RMT-based Theory for Deep Learning
- Tikhonov Regularization versus Heavy-tailed Regularization

4 Validating and Using the Theory

- **Varying the Batch Size: Explaining the Generalization Gap**
- Using the Theory: `pip install weightwatcher`
- Diagnostics at Scale: Predicting Test Accuracies

5 More General Implications and Conclusions

Self-regularization: Batch size experiments

A theory should make predictions:

- We predict the existence of 5+1 phases of increasing implicit self-regularization
- We characterize their properties in terms of HT-RMT

Do these phases exist? Can we find them?

There are *many* knobs. Let's vary one—batch size.

- Tune the batch size from very large to very small
- A small (i.e., retrainable) model exhibits all 5+1 phases
- Large batch sizes \Rightarrow decrease generalization accuracy
- Large batch sizes \Rightarrow decrease implicit self-regularization

Generalization Gap Phenomena: all else being equal, small batch sizes lead to more implicitly self-regularized models.

Batch size and the Generalization Gap

Large versus small batches?

- Larger is better:
 - ▶ Convex theory: SGD is closer to gradient descent
 - ▶ Implementations: Better parallelism, etc.
(But see Golmant et al. (arxiv:1811.12941) and Ma et al. (arxiv:1903.06237) for “inefficiency” of SGD and KFAC.)
- Smaller is better:
 - ▶ Empirical: Hoffer et al. (arXiv:1705.08741) and Keskar et al. (arXiv:1609.04836)
 - ▶ Information: Schwartz-Ziv and Tishby (arxiv:1703.00810)
(This is like a “supervised” version of our approach.)

Connection with weight norms?

- Older: Bartlett, 1997; Mahoney and Narayanan, 2009.
- Newer: Liao et al., 2018; Soudry et al., 2017; Poggio et al., 2018; Neyshabur et al., 2014; 2015; 2017a; Bartlett et al., 2017; Yoshida and Miyato, 2017; Kawaguchi et al., 2017; Neyshabur et al., 2017b; Arora et al., 2018b;a; Zhou and Feng, 2018.

Batch Size Tuning: Generalization Gap

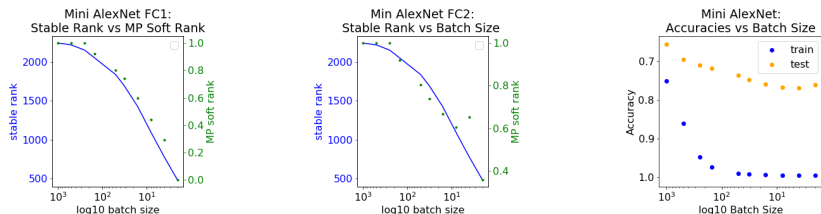


Figure: Varying Batch Size: Stable Rank and MP Softrank for FC1 and FC2 Training and Test Accuracies versus Batch Size for MiniAlexNet.

- *Decreasing* batch size leads to *better* results—it *induces* strong correlations in \mathbf{W} .
- *Increasing* batch size leads to *worse* results—it *washes out* strong correlations in \mathbf{W} .

Batch Size Tuning: Generalization Gap

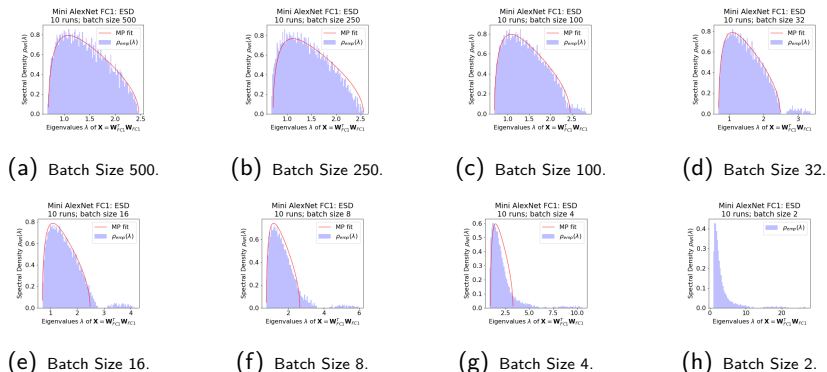


Figure: Varying Batch Size. ESD for Layer FC1 of MiniAlexNet. We exhibit all 5 of the main phases of training by varying only the batch size.

- **Decreasing** batch size **induces** strong correlations in \mathbf{W} , leading to a **more** implicitly-regularized model.
- **Increasing** batch size **washes out** strong correlations in \mathbf{W} , leading to a **less** implicitly-regularized model.

Summary so far

applied Random Matrix Theory (RMT)

self-regularization \sim entropy / information decrease

5+1 phases of learning

small models \sim Tinkhonov-like regularization

modern DNNs \sim heavy-tailed self-regularization

Remarkably ubiquitous

How can this be used?

Why does deep learning work?

Outline

1 Prehistory and History

- Older Background
- A Very Simple Deep Learning Model
- More Immediate Background

2 Preliminary Results

- Regularization and the Energy Landscape
- Preliminary Empirical Results
- Gaussian and Heavy-tailed Random Matrix Theory

3 Developing a Theory for Deep Learning

- More Detailed Empirical Results
- An RMT-based Theory for Deep Learning
- Tikhonov Regularization versus Heavy-tailed Regularization

4 Validating and Using the Theory

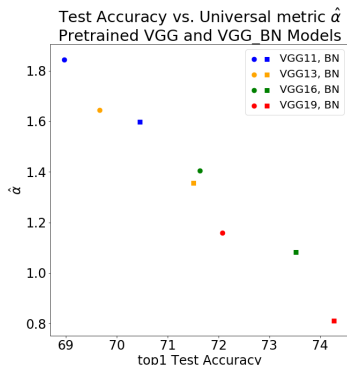
- Varying the Batch Size: Explaining the Generalization Gap
- **Using the Theory: `pip install weightwatcher`**
- Diagnostics at Scale: Predicting Test Accuracies

5 More General Implications and Conclusions

Open source tool: weightwatcher

<https://github.com/CalculatedContent/WeightWatcher>

A python tool to analyze weight matrices in Deep Neural Networks.



All our results can be reproduced by anyone on a basic laptop using widely available, open source, pretrained models: (keras, pytorch, osmr/imgclsmob, huggingface, allennlp, distiller, modelzoo, etc.) and without even needing the test or training data!

WeightWatcher

WeightWatcher is an open source tool to analyze DNNs with our heavy tailed α and weighted $\hat{\alpha}$ metric (and other useful theories)

- goal: to develop a useful, open source tool
- supports: Keras, PyTorch, some custom libs (i.e. huggingface)
- implements: various norm and rank metrics

`pip install weightwatcher`

- current version: 0.1.2
- latest from source: 0.1.3
- looking for: users and contributors

<https://github.com/CalculatedContent/WeightWatcher>

WeightWatcher: Usage

Usage

```
import weightwatcher as ww
watcher = ww.WeightWatcher(model=model)
results = watcher.analyze()
```

```
watcher.get_summary()
watcher.print_results()
```

Advanced Usage

```
def analyze(self, model=None, layers= [],
            min_size= 50, max_size= 0,
            compute_alphas=True,
            compute_lognorms=True,
            compute_spectralnorms=True,
            ...
            plot=True):
```

WeightWatcher: example VGG19_BN

```
import weightwatcher as ww
import torchvision.models as models

model = models.vgg19_bn(pretrained=True)
watcher = ww.WeightWatcher(model=model)
results = watcher.analyze(compute_alphas=True)
data.append("name": "vgg19bntorch", "summary": watcher.get_summary())
```

```
'name': 'vgg19bntorch',
  'summary': 'lognorm': 0.8185,
  'lognorm_compound': 0.9365,
  'alpha': 2.9646,
  'alpha_compound': 2.8479
  'alpha_weighted': 1.1588
  'alpha_weighted_compound': 1.5002
```

We also provide a pandas dataframe with detailed results for each layer

Outline

1 Prehistory and History

- Older Background
- A Very Simple Deep Learning Model
- More Immediate Background

2 Preliminary Results

- Regularization and the Energy Landscape
- Preliminary Empirical Results
- Gaussian and Heavy-tailed Random Matrix Theory

3 Developing a Theory for Deep Learning

- More Detailed Empirical Results
- An RMT-based Theory for Deep Learning
- Tikhonov Regularization versus Heavy-tailed Regularization

4 Validating and Using the Theory

- Varying the Batch Size: Explaining the Generalization Gap
- Using the Theory: `pip install weightwatcher`
- **Diagnostics at Scale: Predicting Test Accuracies**

5 More General Implications and Conclusions

Bounding Generalization Error

BTW, *Bounding Generalization Error* \neq *Predicting Test Accuracies*

A lot of recent interest, e.g.:

- Bartlett et al: (arxiv:1706.08498): bounds based on ratio of output margin distribution and spectral complexity measure
- Neyshabur et al. (arxiv:1707.09564,arxiv:1706.08947): bounds based on the product norms of the weights across layers
- Arora et al. (arxiv:1802.05296): bounds based on compression and noise stability properties
- Liao et al. (arxiv:1807.09659): normalized cross-entropy measure that correlates well with test loss
- Jiang et al. (arxiv:1810.00113): measure based on distribution of margins at multiple layers that correlates well with test loss**

These use/develop *learning theory bounds* and then *apply to training of MNIST/CIFAR10/etc.*

Question: How do these norm-based metrics perform on state-of-the-art pre-trained models?

** and released DEMOGEN pretrained models (after our 1901.08278 paper).

Predicting test accuracies (at scale): Product norms

M&M: "Heavy-Tailed Universality Predicts Trends in Test Accuracies ... Pre-Trained ..." <https://arxiv.org/abs/1901.08278>

The **product norm** is a VC-like data-dependent capacity metric for DNNs.

People prove theorems and then may use it to guide training.

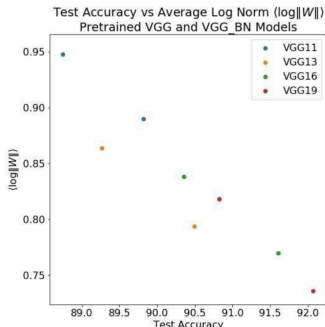
But how does it perform on state-of-the-art production-quality models?

$$\mathcal{C} \sim \|\mathbf{W}_1\| \times \|\mathbf{W}_2\| \cdots \|\mathbf{W}_L\|$$

$$\log \mathcal{C} \sim \log \left[\|\mathbf{W}_1\| \times \|\mathbf{W}_2\| \cdots \|\mathbf{W}_L\| \right]$$

$$\log \mathcal{C} \sim \left[\log \|\mathbf{W}_1\| + \log \|\mathbf{W}_2\| \cdots \log \|\mathbf{W}_L\| \right]$$

$$\langle \log \|\mathbf{W}\|_F \rangle = \frac{1}{N_L} \sum_L \log \|\mathbf{W}_L\|$$



We can predict trends in the test accuracy in state-of-the-art production-quality models—*without peeking at the test data!*

“pip install weightwatcher”

Universality, capacity control, and norm-powerlaw relations

M&M: “Heavy-Tailed Universality Predicts Trends in Test Accuracies ... Pre-Trained ...” <https://arxiv.org/abs/1901.08278>

- “Universality” *suggests* the power law exponent α would make a good, Universal, DNN capacity control metric.
- For multi-layer NN, consider a weighted average

$$\hat{\alpha} = \frac{1}{N} \sum_{l,i} b_{l,i} \alpha_{l,i}$$

- To get weights $b_{l,i}$, relate Frobenius norm and Power Law exponent.
- Create a random Heavy-Tailed (Pareto) matrix:

$$\Pr(W_{i,j}^{rand}) \sim \frac{1}{x^{1+\mu}}$$

- Examine norm-powerlaw relations:

$$\frac{\log \|\mathbf{W}\|_F^2}{\log \lambda_{max}} \quad \text{versus} \quad \alpha$$

- Argue^{††} that:

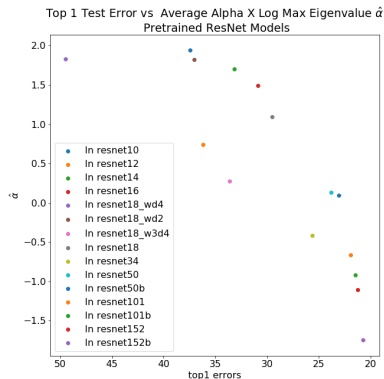
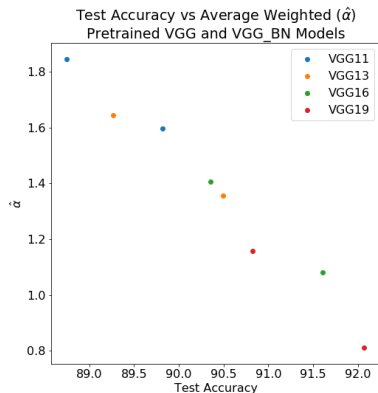
$$\text{PL-Norm Relation: } \alpha \log \lambda^{max} \approx \log \|\mathbf{W}\|_F^2.$$

^{††}Open problem: make the “heuristic” argument more “rigorous.”

Predicting test accuracies better: Weighted Power Laws

M&M: "Heavy-Tailed Universality Predicts Trends in Test Accuracies ... Pre-Trained ..." <https://arxiv.org/abs/1901.08278>

Use the **weighted PL metric**: $\hat{\alpha} = \frac{1}{N} \sum_{l,i} \log(\lambda_{l,i}^{\max}) \alpha_{l,i}$, instead of the product norm.



We can predict trends in the test accuracy—*without peeking at the test data!*

“pip install weightwatcher”

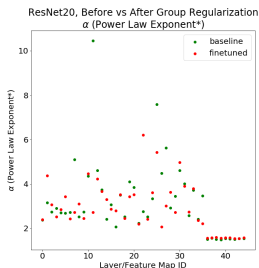
Predicting test accuracies better: Distilled Models

M&M: "Heavy-Tailed Universality Predicts Trends in Test Accuracies ... Pre-Trained ..." <https://arxiv.org/abs/1901.08278>

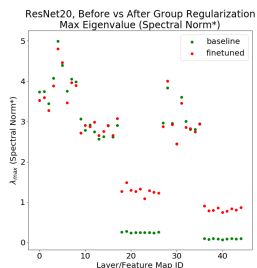
Question: Is the **weighted PL metric** simply a repackaging of the **product norm**?

Answer: No!

For some Intel Distiller models, the Spectral Norm behaves atypically, and α does not change noticeably



(a) PL Exponents (α)



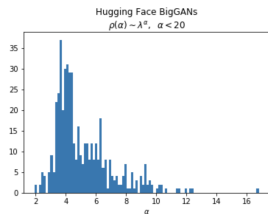
(b) Spectral Norm (λ_{max})

Figure: (a) PL exponents α and (b) Spectral Norm (λ_{max}) for each layer of ResNet20, with Intel Distiller Group Regularization method applied (before and after).

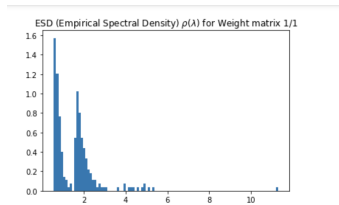
WeighWatcher: ESDs of GANs

GANs also display Heavy Tailed ESDs, however:

- There are more exceptions
- Many ESDs appear to display significant rank collapse and only weak correlations



(a) Histogram of α s



(b) Anomalous ESD.

Figure: (a) Distribution of all power law exponents α for DeepMind's BigGAN (Huggingface implementation). (b) Example of anomalous ESD.

Summary of “Validating and Using”

Some things so far:

- We can: **explain** the generalization gap.
- We can: “**pip install weightwatcher**” and use this source tool.
- We can: **predict** trends in test accuracies on production-quality models.

Some things we are starting to look at:

- Better metrics for monitoring and/or improving training.
- Better metrics for robustness to, e.g., adversarial perturbation, model compression, etc., that don’t involve looking at data.
- Better phenomenological load-like and temperature-like metrics to guide data collection, algorithm/parameter/hyperparameter selection, etc.
- What else?

Join us:

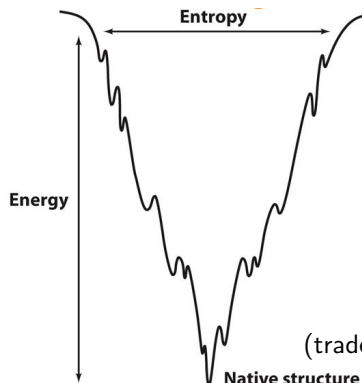
- “**pip install weightwatcher**”—contribute to the repo.^{‡‡}

^{‡‡} *Don’t do everything from scratch in a non-reproducible way. Make it reproducible!* 🔍 ↻

Outline

- 1 Prehistory and History
 - Older Background
 - A Very Simple Deep Learning Model
 - More Immediate Background
- 2 Preliminary Results
 - Regularization and the Energy Landscape
 - Preliminary Empirical Results
 - Gaussian and Heavy-tailed Random Matrix Theory
- 3 Developing a Theory for Deep Learning
 - More Detailed Empirical Results
 - An RMT-based Theory for Deep Learning
 - Tikhonov Regularization versus Heavy-tailed Regularization
- 4 Validating and Using the Theory
 - Varying the Batch Size: Explaining the Generalization Gap
 - Using the Theory: `pip install weightwatcher`
 - Diagnostics at Scale: Predicting Test Accuracies
- 5 More General Implications and Conclusions

Implications: RMT and Deep Learning



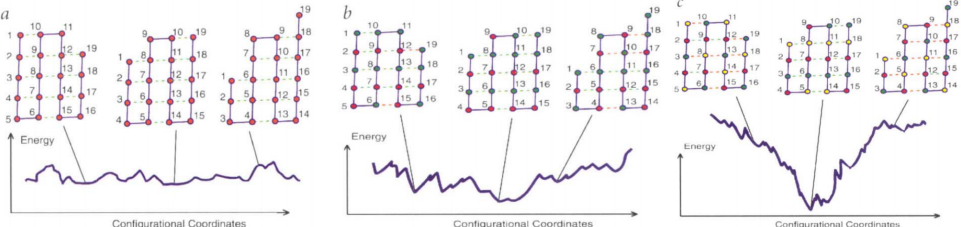
- Where are the local minima?
- How is the Hessian behaved?
- Are simpler models misleading?
- Can we design better learning strategies?

(tradeoff between Energy and Entropy minimization)

How can RMT be used to understand the Energy Landscape?

Implications: Minimizing Frustration and Energy Funnels

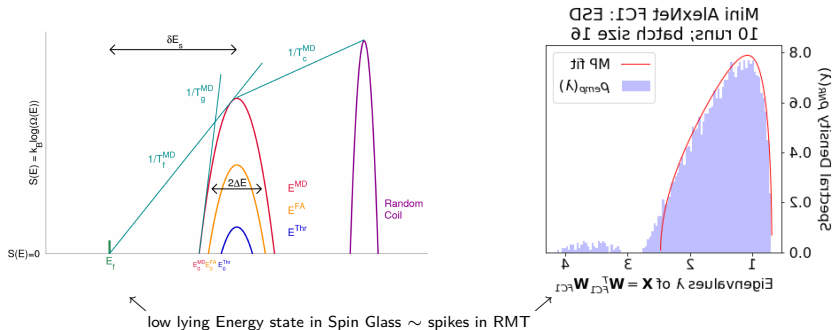
As simple as can be?, Wolynes, 1997



Energy Landscape Theory: “random heteropolymer” versus “natural protein” folding

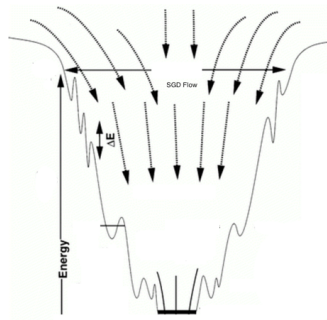
Implications: The Spin Glass of Minimal Frustration

<https://calculatedcontent.com/2015/03/25/why-does-deep-learning-work/>



Implications: Rugged Energy Landscapes of Heavy-tailed Models

Martin and Mahoney <https://arxiv.org/abs/1710.09553>



Spin Glasses with Heavy Tails?

- Local minima do **not** concentrate near the ground state (Cizeau and Bouchaud 1993)
- Configuration space with a “rugged convexity”

Contrast with (Gaussian) Spin Glass model of Choromanska et al. 2015

If Energy Landscape is ruggedly funneled, then no “problems” with local minima!

Conclusions: “pip install weightwatcher”

Statistical mechanics and neural networks

- Long history
- Revisit due to recent “crisis” in why deep learning works
- Use ideas from statistical mechanics of strongly-correlated systems
- Develop a theory that is designed to be used

Main Empirical/Theoretical Results

- Use Heavy-tailed RMT to construct a operational theory of DNN learning
- Evaluate effect of implicit versus explicit regularization
- Exhibit all 5+1 phases by adjusting batch size: explain the generalization gap
- Methodology: Observations → Hypotheses → Build a Theory → Test the Theory.

Many Implications:

- Explain the generalization gap
- Rationalize claims about rugged convexity of Energy Landscape
- Predict test accuracies in state-of-the-art models
- “pip install weightwatcher”

If you want more ... “pip install weightwatcher” ...

Background paper:

- Rethinking generalization requires revisiting old ideas: statistical mechanics approaches and complex learning behavior
(<https://arxiv.org/abs/1710.09553>)

Main paper (full):

- Implicit Self-Regularization in Deep Neural Networks: Evidence from Random Matrix Theory and Implications for Learning
(<https://arxiv.org/abs/1810.01075>)
- Code: <https://github.com/CalculatedContent/ImplicitSelfRegularization>

Main paper (abridged):

- Traditional and Heavy-Tailed Self Regularization in Neural Network Models
(<https://arxiv.org/abs/1901.08276>)
- Code: <https://github.com/CalculatedContent/ImplicitSelfRegularization>

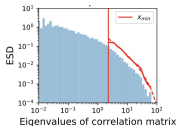
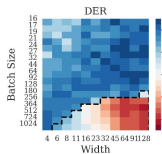
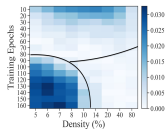
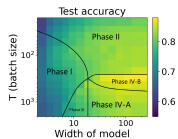
Applying the theory paper:

- Heavy-Tailed Universality Predicts Trends in Test Accuracies for Very Large Pre-Trained Deep Neural Networks
(<https://arxiv.org/abs/1901.08278>)
- Code: <https://github.com/CalculatedContent/PredictingTestAccuracies>
- <https://github.com/CalculatedContent/WeightWatcher>
- **“pip install weightwatcher”**

Outline

- 1 Weight Analysis and Heavy-Tailed Self-Regularization
- 2 Phenomenological Approach to Statistical Mechanics of Generalization**
- 3 Using Heavy-Tailed Self-Regularization
- 4 Random Matrix Theory for Modern ML
- 5 Putting It All Together
- 6 Conclusion

Loss landscape and weight analytics



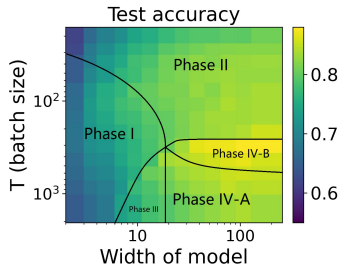
Part I. Phase Transitions

Part II. Pruning

Part III. Ensembling

Part IV. Weight analytics

Loss landscape and weight analytics



Part I. Phase Transitions

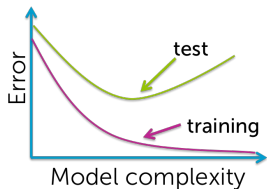
Part II. Pruning

Part III. Ensembling

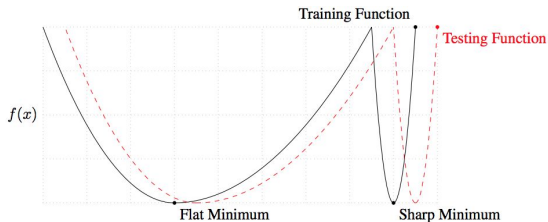
Part IV. Weight analytics

Local versus global in generalization

Prediction on unseen data



Flat local minima generalize better?

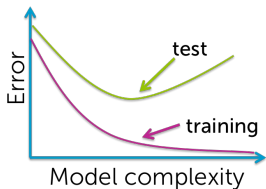


Flat vs sharp local minima

Keskar et al. 2017

Local versus global in generalization

Prediction on unseen data



Flat local minima generalize better?



Yes

vs

No

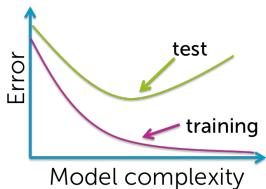


Keskar et al. '17
Neyshabur et al. '17
Jiang et al. '19
Foret et al. '20

Dinh et al. '17
Yao et al. '18
Granzio et al. '20
Zhang et al. '21

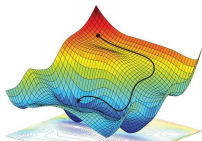
Local versus global in generalization

Prediction on unseen data



Paper reviews.

- Different data?
- Different architecture?
- Different Hyperparameters?



Is your result too **local**?

Main difficulty: Analyzing the global picture



Is your result too **local**?



Image source: Foundations of data science, Simons Institute

Our answer

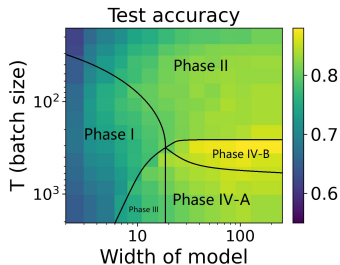
It depends on the phase!

Yang, Hodgkinson, Theisen, Zou, Gonzalez, Ramchandran, Mahoney, NeurIPS 2021

Our answer

Phase transitions in deep learning.

Different phases \rightarrow Different conclusions



Yang, Hodgkinson, Theisen, Zou, Gonzalez, Ramchandran, Mahoney, NeurIPS 2021

Our answer

Flat local minima generalize better?



Yes

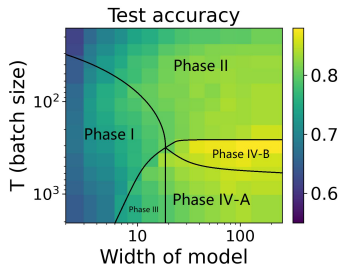
vs

No



Keskar et al. '17
Neyshabur et al. '17
Jiang et al. '19
Foret et al. '20

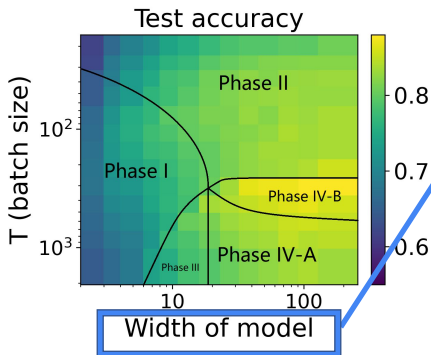
Dinh et al. '17
Yao et al. '18
Granzio et al. '20
Zhang et al. '21



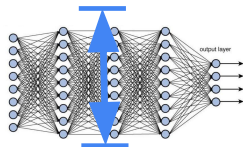
Different phases → Different conclusions

Yang, Hodgkinson, Theisen, Zou, Gonzalez, Ramchandran, Mahoney, NeurIPS 2021

What are the phases?



Width of

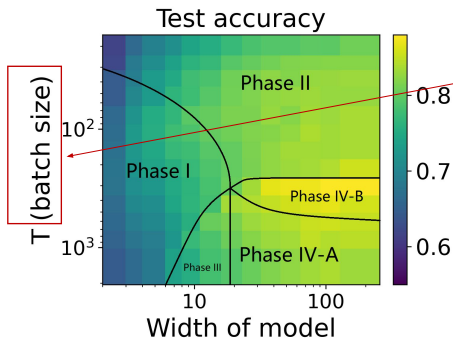


Other choices

- Quantity of data
- Quality of data

“Load parameter” [Martin & Mahoney, 2017]

What are the phases?

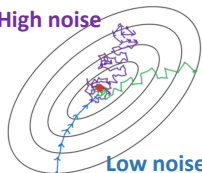


Batch size

SGD equation

$$\theta_{t+1} = \theta_t - \frac{\eta}{B} \sum_{i=1}^B g(x_i; \theta_t)$$

High noise



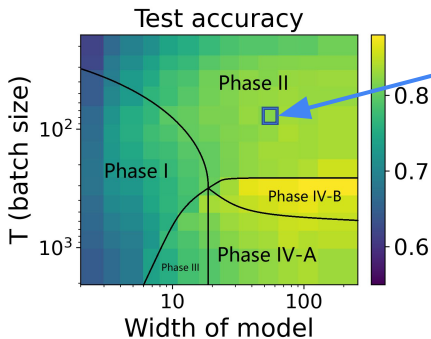
Other choices

- Learning rate
- L2 regularization

“Temperature”

[Martin & Mahoney, 2017]

What are the phases?

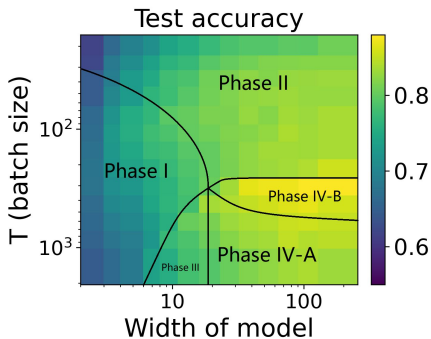


(width, batch size)

~200 pixels × 5 networks/pixel

Phase transitions?

What are the phases?



Five phases

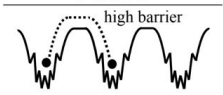
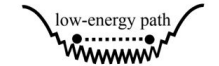

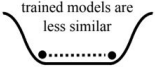
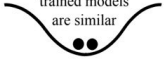
Five different loss landscapes

	Globally poorly-connected	Globally well-connected	
Locally sharp	Phase I high barrier	Phase II low-energy path	
Locally flat	Phase III high barrier	Phase IV-A trained models are less similar	Phase IV-B trained models are similar

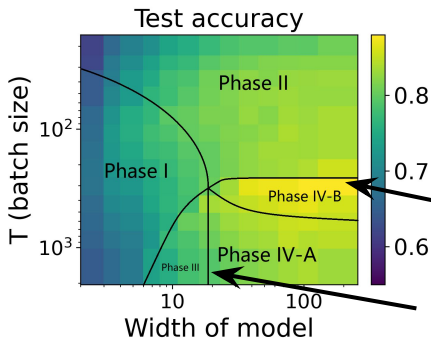
What are the phases?

Five phases

Five different loss landscapes

	Globally poorly-connected	Globally well-connected	
Locally sharp	<p>Phase I</p> 	<p>Phase II</p> 	
Locally flat	<p>Phase III</p> 	<p>Phase IV-A</p> 	<p>Phase IV-B</p> 

What are the phases?



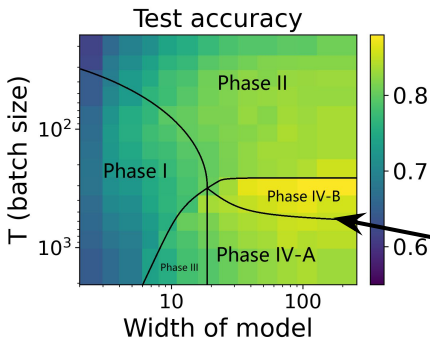
Five phases

Five different loss landscapes

	Globally poorly-connected	Globally well-connected	
Locally sharp	<p>Phase I</p>	<p>Phase II</p>	
Locally flat	<p>Phase III</p>	<p>Phase IV-A</p>	<p>Phase IV-B</p>

$2 \times 2 = 4$ phases

What are the phases?



Five phases

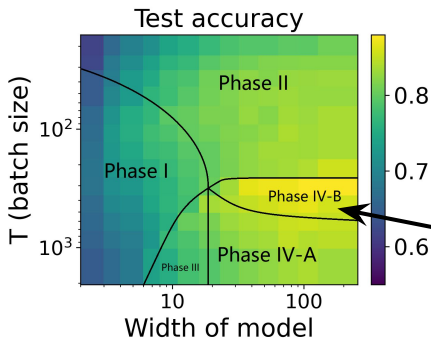
Five different loss landscapes

	Globally poorly-connected	Globally well-connected	
Locally sharp	Phase I high barrier	Phase II low-energy path	
Locally flat	Phase III high barrier	Phase IV-A trained models are less similar	Phase IV-B trained models are similar

Model similarity [Kornblith et al. 19][Jiang et al. 21]

2x2+1=5 phases

What are the phases?



Five phases

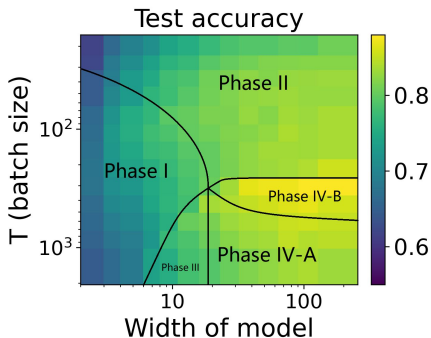
Five different loss landscapes

	Globally poorly-connected	Globally well-connected	
Locally sharp	Phase I high barrier	Phase II low-energy path	
Locally flat	Phase III high barrier	Phase IV-A trained models are less similar	Phase IV-B trained models are similar

Best phase

Try to get here!!

What are the phases?



Five phases

Five different loss landscapes

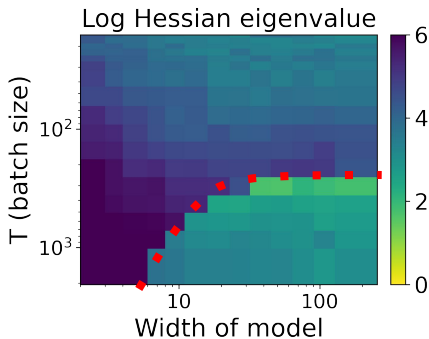
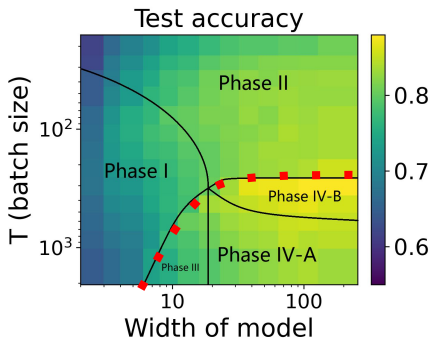
	Globally poorly-connected	Globally well-connected	
Locally sharp	Phase I high barrier	Phase II low-energy path	
Locally flat	Phase III high barrier	Phase IV-A trained models are less similar	Phase IV-B trained models are similar

Generalization measures

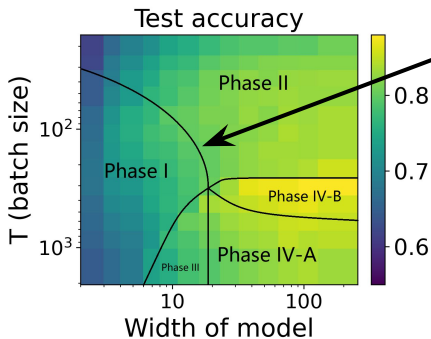
1. **Local** [Yao et al. 19]
2. **Global** [Garipov et al. 18][Kornblith et al. 19]

What are the phases?

$$H = \begin{pmatrix} \frac{\partial^2 \text{loss}}{\partial \theta_1^2} & \cdots & \frac{\partial^2 \text{loss}}{\partial \theta_1 \partial \theta_n} \\ \vdots & \ddots & \vdots \\ \frac{\partial^2 \text{loss}}{\partial \theta_n \partial \theta_1} & \cdots & \frac{\partial^2 \text{loss}}{\partial \theta_n^2} \end{pmatrix}$$



What are the phases?



Global structure

Mode connectivity [Garipov et al. 18]



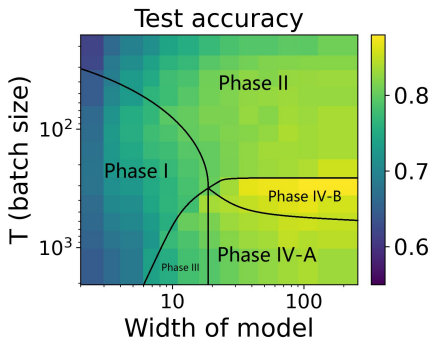
A hidden path in high dimensions.

What are the phases?

End of definitions...

How do we use them?

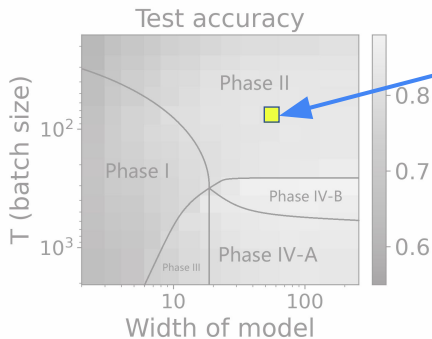
How to use the phases?



Full phase plot

If you study all settings...

How to use the phases?



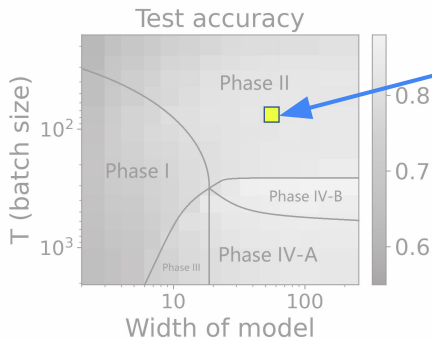
Your own CV / NLP problem

(width, batch size)

Step 1. Locate your problem on this map

Generalization measure: Landmarks

How to use the phases?



Your own CV / NLP problem

(width, batch size)

Step 1. Locate your problem on this map

Step 2. Find the best way to get to Phase IV-B

How to use the phases?

Step 1. Calculate measures. **Phase plot is not necessary!!**

Hessian	Mode connectivity	Model similarity	Phase	Treatment
large	negative	low	I	Larger network
large	positive	low	II	Smaller learning rate
small	negative	low	III	Larger network
small	Close to zero	low	IV-A	Buy more data
small	Close to zero	high	IV-B	Perfect

How to use the phases?

Step 2. Determine phase.

Hessian	Mode connectivity	Model similarity	Phase	Treatment
large	negative	low	I	Larger network
large	positive	low	II	Smaller learning rate
small	negative	low	III	Larger network
small	Close to zero	low	IV-A	Buy more data
small	Close to zero	high	IV-B	Perfect

How to use the phases?

Step 3. Provide treatment.

Hessian	Mode connectivity	Model similarity	Phase	Treatment
large	negative	low	I	Larger network
large	positive	low	II	Smaller learning rate
small	negative	low	III	Larger network
small	Close to zero	low	IV-A	Buy more data
small	Close to zero	high	IV-B	Perfect

How to use the phases?

Step 3. Provide treatment.

Hessian	Mode connectivity	Model similarity	Phase	Treatment
large	negative	low	I	Larger network
large	positive	low	II	Smaller learning rate
small	negative	low	III	Larger network
small	Close to zero	low	IV-A	Buy more data
small	Close to zero	high	IV-B	Perfect

Diagnose the failure

A Get more data?

B Larger model?

C Hyperparameters?



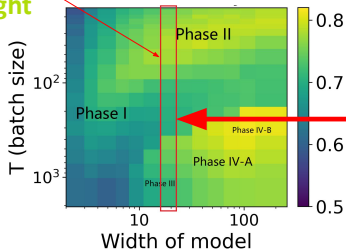
How to use the phases?

An example of using the phase.

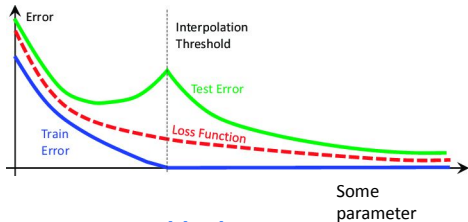
A different phase, a different story

Red column: Bright → dark →

bright



Training with randomized labels



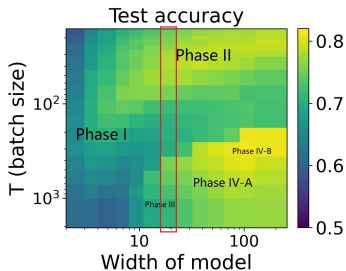
Double descent

[Belkin et al. 19][Hastie et al. 20]

[Nakkiran et al. 19]

...

A different phase, a different story



Flat local minima generalize better?



Yes

vs

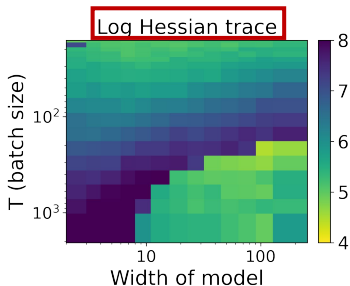
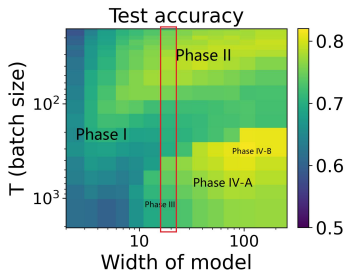
No



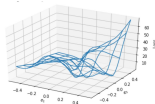
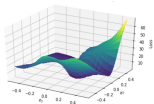
Keskar et al. '17
Neyshabur et al. '17
Jiang et al. '19
Foret et al. '20

Dinh et al. '17
Yao et al. '18
Granzio et al. '20
Zhang et al. '21

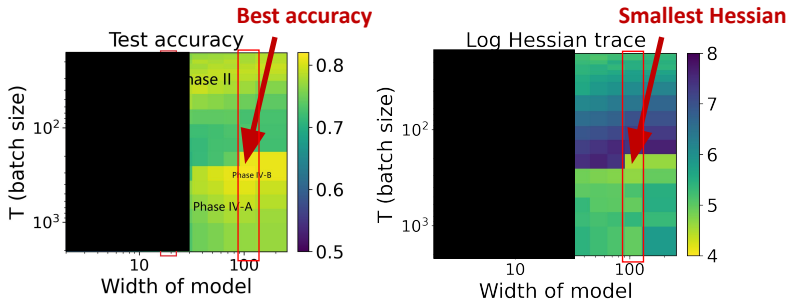
A different phase, a different story



Primary tool to measure flatness



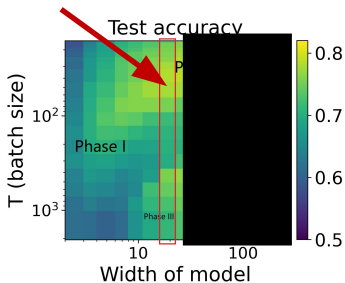
A different phase, a different story



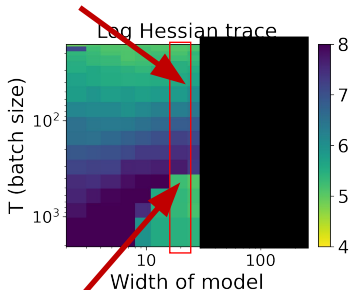
The flattest minimum gets the best accuracy

A different phase, a different story

Best accuracy



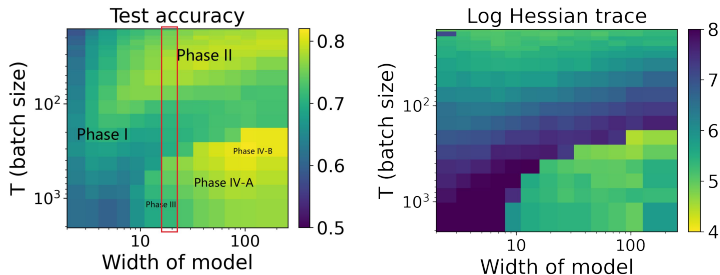
Not the best!!!



Best Hessian

The flattest minimum **does not** get the best accuracy.

A different phase, a different story



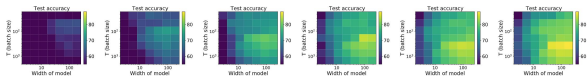
Our paper. The conclusion depends on the phase.

Yang, Hodgkinson, Theisen, Zou, Gonzalez, Ramchandran, Mahoney, NeurIPS 2021

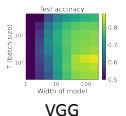
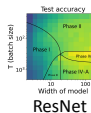
Is the result “local”?

Five phases exist.
Phase IV-B is the best.

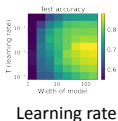
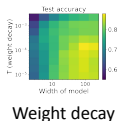
Different amount of data



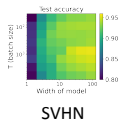
Different architectures



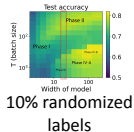
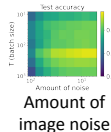
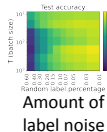
Different temperature parameters



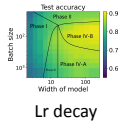
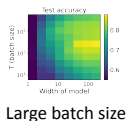
Different dataset



Quality of data

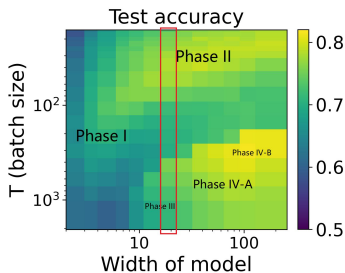


Different training approaches



Brief summary

- Phases \rightarrow different conclusions



Train to flat minima?



Yes

vs

No



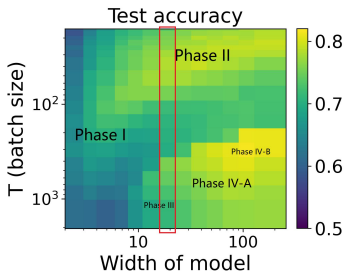
Keskar et al. '17
Neyshabur et al. '17
Jiang et al. '19
Foret et al. '20

Dinh et al. '17
Yao et al. '18
Granzio et al. '20
Zhang et al. '21

The conclusion depends on the phase.

Brief summary

- Phases → different conclusions
- Phases → different treatments



Diagnose the failure

A Get more data?

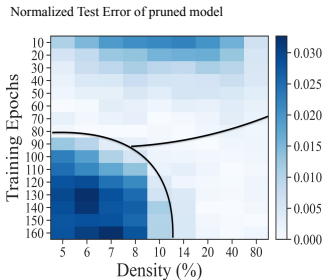
B Larger model?

C Hyperparameters?



A practitioner's map

Loss landscape and weight analytics



Part I. Phase Transitions

Part II. Pruning

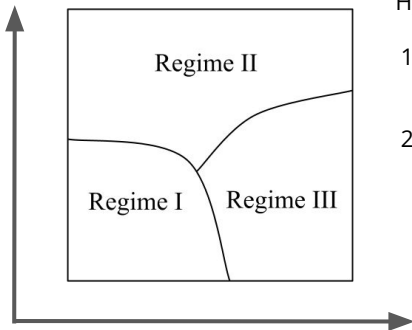
Part III. Ensembling

Part IV. Weight analytics

Phase transitions for network pruning

Training Epochs

(Temperature-like parameter)



Hypothesis

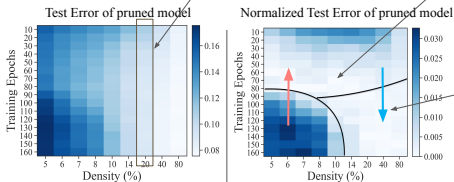
1. Does the multi-regime (phase) phenomenon exist?
2. Can we quantify these regimes with loss landscape metrics?

Modeling

For a target model density, which training epoch is optimal?

White pixel represents optimal training epoch for this model density (column)

Empirical Results for modeling



An interesting dichotomous phenomenon:
Increasing temperature better for low density,
decreasing temperature better for high density.

Experiment Setting: ResNet20/CIFAR-10

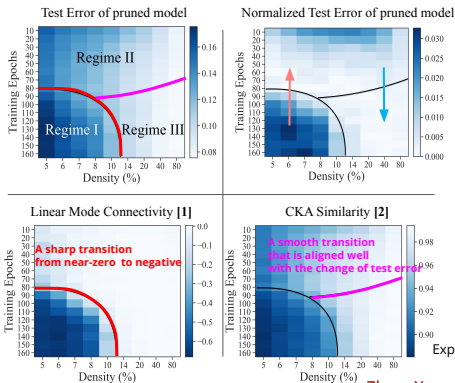
Zhou, Yang, Chang, Mahoney, 2022

- [1] Garipov, T., Izmailov, P., Podoprikin, D., Vetrov, D. P., & Wilson, A. G. (2018). Loss surfaces, mode connectivity, and fast ensembling of dnn. *Advances in neural information processing systems*, 31.
- [2] Kornblith, S., Norouzi, M., Lee, H., & Hinton, G. (2019, May). Similarity of neural network representations revisited. In *International Conference on Machine Learning* (pp. 3519-3529). PMLR.

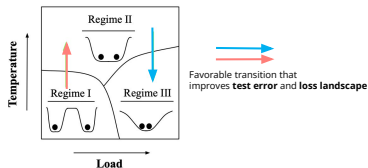
Modeling

Empirical Results for modeling

Taxonomizing



VSDL model for pruning



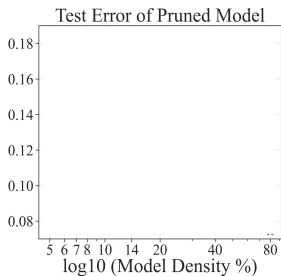
	poor	good	best
	Regime I 	Regime II 	Regime III
Connectivity	✗	✓	✓
Similarity	✗	✗	✓

Experiment Setting: ResNet20/CIFAR-10

Zhou, Yang, Chang, Mahoney, 2022

Application

Task: prune to different densities, and select the best training hyperparameter for each density



*(multiple markers in one column
represent repeated experiments)*

A conventional wisdom:

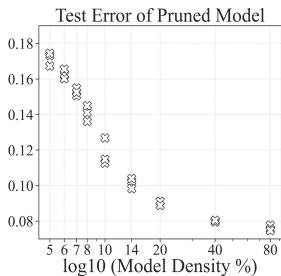
Train the dense model to best (lowest test error), and then prune

Experiment Setting: tuning training epochs for **ResNet20/CIFAR-10**

Zhou, Yang, Chang, Mahoney, 2022

Application

Baseline: test-error-based selection



☒ Select by test error of dense model

*(multiple markers in one column
represent repeated experiments)*

Everything looks good if we only look at the test error.

A conventional wisdom:

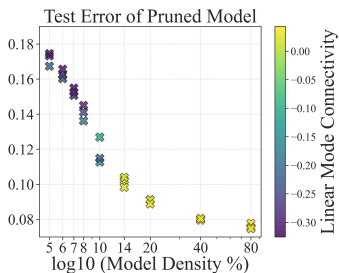
Train the dense model to best (lowest test error), and then prune

Experiment Setting: tuning training epochs for **ResNet20/CIFAR-10**

Zhou, Yang, Chang, Mahoney, 2022

Application

Three-regime model: loss landscape metric (linear mode connectivity)



⊗ Select by test error of dense model

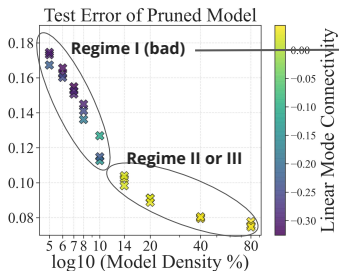
*(multiple markers in one column
represent repeated experiments)*

Experiment Setting: tuning training epochs for **ResNet20/CIFAR-10**

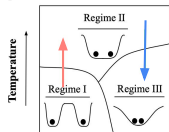
Zhou, Yang, Chang, Mahoney, 2022

Application

Three-regime model: loss landscape metric diagnoses the problem of baseline.



*Choice of hyperparameter is bad
conventional wisdom doesn't work*



	poor	good	best
	Regime I	Regime II	Regime III
Connectivity	✗	✓	✓
Similarity	✗	✗	✓

✗ Select by test error of dense model

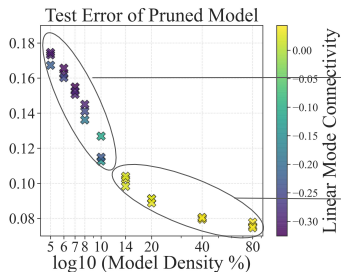
*(multiple markers in one column
represent repeated experiments)*

Experiment Setting: tuning training epochs for **ResNet20/CIFAR-10**

Zhou, Yang, Chang, Mahoney, 2022

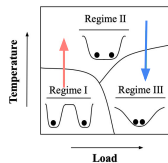
Application

Tuning the baseline by the three-regime based approach



Regime I:
Tune by **increasing** temperature
until $LMC \geq 0$

Regime II or III:
Tune by **decreasing** temperature
until CKA doesn't improve



	poor	good	best
	Regime I 	Regime II 	Regime III
Connectivity	✗	✓	✓
Similarity	✗	✗	✓

✗ Select by test error of dense model

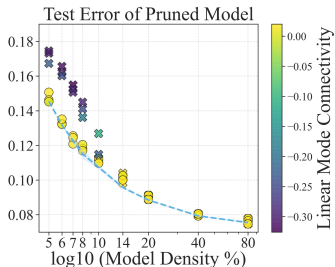
*(multiple markers in one column
represent repeated experiments)*

Experiment Setting: tuning training epochs for **ResNet20/CIFAR-10**

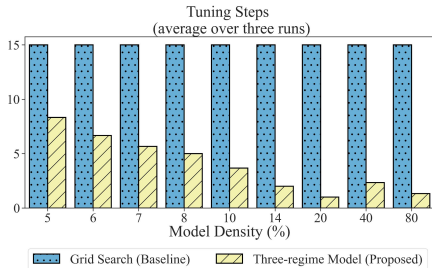
Zhou, Yang, Chang, Mahoney, 2022

Application

Results: Our approach can achieve the optimal performance as Grid Search, but **in fewer steps**.



- ⊗ Select by test error of dense model
 - Tuned by Three-regime Model (Proposed)
 - Tuned by Grid Search (Baseline)
- (multiple markers in one column represent repeated experiments)*



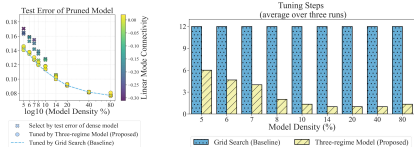
Experiment Setting: tuning training epochs for **ResNet20/CIFAR-10**

Zhou, Yang, Chang, Mahoney, 2022

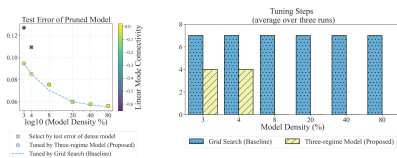
Generalizability

Our approach can work for different **hyperparameter**, **architectures** and **dataset**.

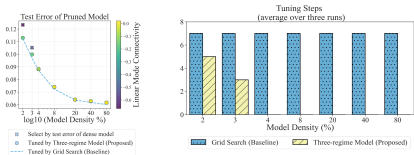
ResNet-20 on CIFAR-10 (tuning batch size)



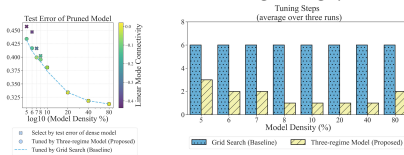
DenseNet-40 on CIFAR-10 (tuning training epochs)



VGG19 on CIFAR-10 (tuning training epochs)



ResNet-20 on CIFAR-100 (tuning training epochs)



Zhou, Yang, Chang, Mahoney, 2022

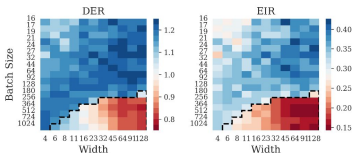
Brief summary

1. Conventional wisdom (test error based) doesn't work when we look at a different regime.
2. Three-regime based hyperparameter tuning is more efficient than grid search.

Next steps

1. How easy/hard is it to plant/detect back doors in different regimes?
2. A more challenging task: do hyperparameter search on both ``load'' and ``temperature''.

Loss landscape and weight analytics



Part I. Phase Transitions

Part II. Pruning

Theorem

$$\text{DER} \geq \text{EIR} \geq \frac{2(K-1)}{K} \text{DER} - \frac{3K-4}{K}$$

Part III. Ensembling

Part IV. Weight analytics

Ensembling

Focus on ensembles of classifiers $h \sim \rho$ where ρ could represent, e.g.:

1. A distribution over parameters obtained from independent runs of SGD, from either dependent (e.g. fine-tuning) or independent initializations
2. A finite set of classifiers h_1, \dots, h_M with weights ρ_1, \dots, ρ_M
3. A Bayesian posterior distribution over classifiers

We focus on the the majority-vote classifier:

$$h_{MV}(x) = \arg \max_y E_{h \sim \rho} [\mathbf{1}(h(x) = y)]$$

Theisen, Kim, Yang, Hodgkinson, Mahoney, 2022

Ensembling

In theory there is a large literature on ensembling, but most is either specialized to particular settings (like random forests), or is too weak to even guarantee that ensembling can help at all, much less accurately quantify how much it can help

In practice there is a wide variety of (sometimes contradictory) results regarding ensembles – notably for deep ensembles. Some work suggests ensembling is highly beneficial, others suggest it is less so, and particularly unnecessary for large, modern models

Theoretical question: Can we characterize when, and by how much, ensembling benefits?

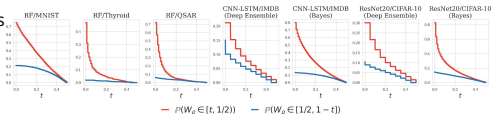
Empirical question: When can we expect ensembling to help significantly in practice?

Theisen, Kim, Yang, Hodgkinson, Mahoney, 2022

Ensembling theory

- Rules out pathological cases that limit previous theoretical analyses of ensembling
- Holds widely in practice

The competence assumption



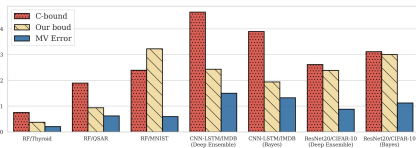
New, significantly sharper upper bounds on the majority-vote error rate

Theorem (first-order bound)

$$L(h_{MV}) \leq E_{h \sim \rho}[L(h)]$$

Theorem (second-order bound)

$$L(h_{MV}) \leq \frac{4(K-1)}{K} (E_{h \sim \rho}[L(h)] - \frac{1}{2} E_{h, h' \sim \rho}[\text{Dis}(h, h')])$$



Theisen, Kim, Yang, Hodgkinson, Mahoney, 2022

Ensembling theory

Characterizing the ensemble improvement rate with the disagreement-error ratio

Ensemble improvement rate

$$\text{EIR} = \frac{E_{h \sim \rho}[L(h)] - L(h_{MV})}{E_{h \sim \rho}[L(h)]}$$

Disagreement-error ratio

$$\text{DER} = \frac{E_{h, h' \sim \rho}[D(h, h')]}{E_{h \sim \rho}[L(h)]}$$

Theorem

$$\text{DER} \geq \text{EIR} \geq \frac{2(K-1)}{K} \text{DER} - \frac{3K-4}{K}$$

We identify two regimes:

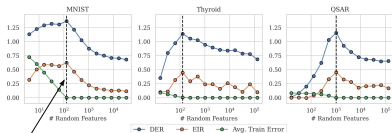
1. Ensembles are guaranteed to improve performance when DER is large, **disagreement > average error**
2. Ensembles are guaranteed to *not* improve performance too much when DER is small, **disagreement < average error**

Theisen, Kim, Yang, Hodgkinson, Mahoney, 2022

Phase transition

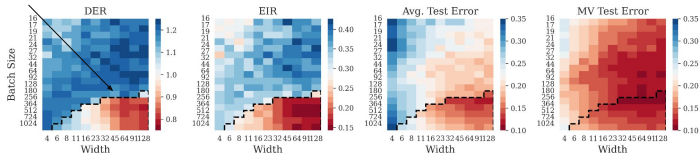
Ensemble improvement, DER become small in the interpolating regime

Bagged Random
Feature classifiers



- Ensembling becomes less useful for large models which can easily interpolate the training data (i.e., obtain zero training error)
- This corresponds to the fact that the disagreement-error ratio gets small in this regime

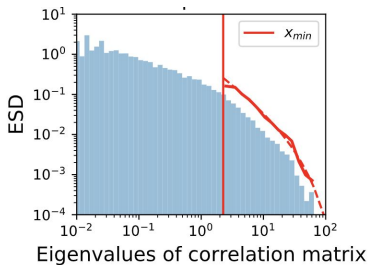
Interpolation threshold



ResNet18/CIFAR-10
Deep Ensembles

Theisen, Kim, Yang, Hodgkinson, Mahoney, 2022

Loss landscape and weight analytics



Part I. Phase Transitions

Part II. Pruning

Part III. Ensembling

Part IV. Weight analytics

Measures that do not require access to data

The screenshot shows the Hugging Face website interface. At the top, the Hugging Face logo is on the left, followed by a search bar containing "Search models, datasets, users...". Navigation links for "Models", "Datasets", "Spaces", "Docs", "Solutions", "Pricing", "Log In", and "Sign Up" are visible. On the left sidebar, there are sections for "Tasks" (listing various NLP tasks like Fill-Mask, Summarization, etc.) and "Libraries" (listing PyTorch, TensorFlow, JAX, etc.). The main content area is titled "Models 25,578" and includes a search bar and a sort dropdown set to "Most Downloads". A list of model cards is displayed, with the first card "bert-base-uncased" highlighted in red and overlaid with the text "25,578 pretrained models". Other model cards include "gpt2", "cardiffnlp/twitter-roberta-base-sentiment", "roberta-base", "bert-base-multilingual-cased", and "distilbert-base-uncased". On the right side of the model cards, the text "Albert", "Bert-base", and "DistilBert" is displayed in blue.

Hugging Face Search models, datasets, users... Models Datasets Spaces Docs Solutions Pricing Log In Sign Up

Tasks

- Fill-Mask
- Question Answering
- Summarization
- Table Question Answering
- Text Classification
- Text Generation
- Text2Text Generation
- Token Classification
- Translation
- Zero-Shot Classification
- Sentence Similarity

Libraries

- PyTorch
- TensorFlow
- JAX

Datasets

- wikipedia
- common_voice
- bookcorpus
- glue
- squad
- dcep_europarl_jrc-acquis
- conll2003
- oscar

Models 25,578 Search Models Ti Sort: Most Downloads

bert-base-uncased 25,578 pretrained models

gpt2

cardiffnlp/twitter-roberta-base-sentiment

roberta-base

bert-base-multilingual-cased

distilbert-base-uncased

Albert

Bert-base

DistilBert

Measures that do not require access to data

The screenshot shows the Hugging Face website interface. At the top, the navigation bar includes the Hugging Face logo, a search bar for models, datasets, and users, and links for Models, Datasets, Spaces, Docs, Solutions, Pricing, Log In, and Sign Up. On the left sidebar, there are sections for Tasks (Fill-Mask, Summarization, Text Classification, Text2Text Generation, Translation, Sentence Similarity) and Libraries (PyTorch, TensorFlow, JAX). The main content area displays a list of models, with the 'Models' tab selected and showing a count of 25,578. The first model listed is 'bert-base-uncased'. Overlaid on the image are three text annotations: '25,578 pretrained models' in red, 'Data is unavailable...' in blue, and 'Web-scale data Private datasets' in black.

Hugging Face Search models, datasets, users... Models Datasets Spaces Docs Solutions Pricing Log In Sign Up

Tasks

- Fill-Mask Question Answering
- Summarization Table Question Answering
- Text Classification Text Generation
- Text2Text Generation Token Classification
- Translation Zero-Shot Classification
- Sentence Similarity +13

Libraries

- PyTorch TensorFlow JAX +22

Datasets

- wikipedia common_voice bookcorpus
- glue squad deep_eurolp_jrc-acquis
- conll2003 cscar +680

Models 25,578 Search Models Ti Sort: Most Downloads

25,578 pretrained models

Data is unavailable...

Web-scale data Private datasets

bert-base-uncased
Fill-Mask · Updated May 18, 2021 · 4.61M · 13

gpt2
Text Generation · Updated May 19, 2021 · 4.13.5M · 30

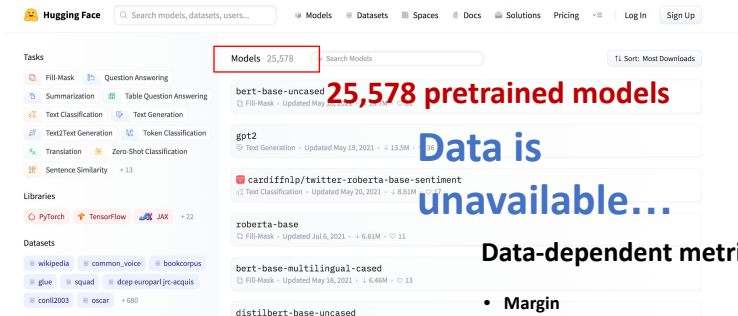
cardiffnlp/twitter-roberta-base-sentiment
Text Classification · Updated May 20, 2021 · 4.8.61M · 17

roberta-base
Fill-Mask · Updated Jul 6, 2021 · 4.6.61M · 11

bert-base-multilingual-cased
Fill-Mask · Updated May 18, 2021 · 4.6.46M · 13

distilbert-base-uncased

Measures that do not require access to data



The screenshot shows the Hugging Face website interface. At the top, the navigation bar includes the Hugging Face logo, a search bar for models, datasets, and users, and links for Models, Datasets, Spaces, Docs, Solutions, Pricing, Log In, and Sign Up. On the left sidebar, there are sections for Tasks (Fill-Mask, Summarization, Text Classification, Text2Text Generation, Translation, Sentence Similarity) and Libraries (PyTorch, TensorFlow, JAX). The main content area displays a list of models. A red box highlights the 'Models 25,578' count. The first model listed is 'bert-base-uncased', followed by 'gpt2', 'cardiffnlp/twitter-roberta-base-sentiment', 'roberta-base', 'bert-base-multilingual-cased', and 'distilbert-base-uncased'. The text 'Data is unavailable...' is overlaid in blue on the 'gpt2' and 'cardiffnlp/twitter-roberta-base-sentiment' model cards. The text 'Data-dependent metrics' is overlaid in black on the 'roberta-base' model card.

Models 25,578

25,578 pretrained models

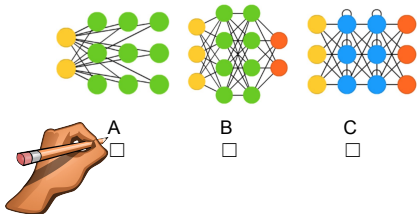
Data is unavailable...

Data-dependent metrics

- **Margin**
[Bartlett 17][Pitas 17]
- **PAC-Bayesian**
[Neyshabur 17][Mcalister 99]

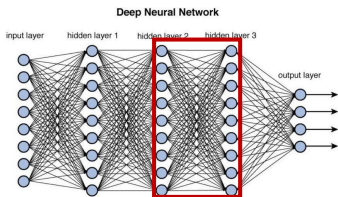
Measures that do not require access to data

Model selection without data



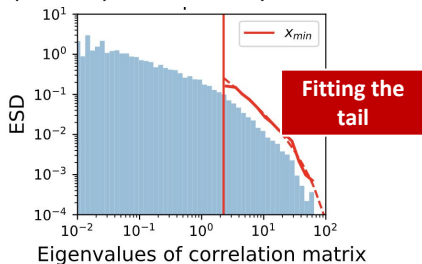
Which model should I use?

Generalization measures from statistical physics



Take the weight matrix W .

Empirical spectral density of $W^T W$.

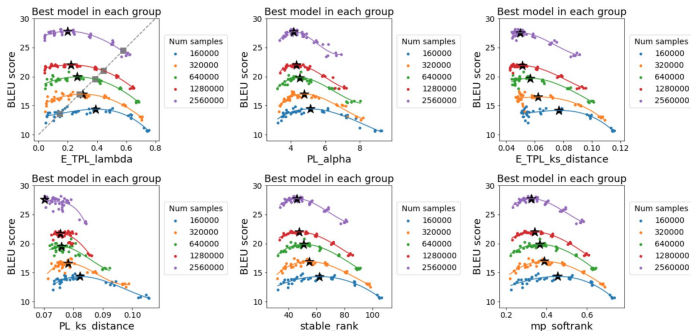


Exponentially-Truncated Power Law

$$p(x) = x^{-\beta} e^{-\lambda x}$$

Yang, Theisen, Hodgkinson, Gonzalez, Ramchandran, Martin, Mahoney, 2022, <https://arxiv.org/pdf/2202.02842.pdf>

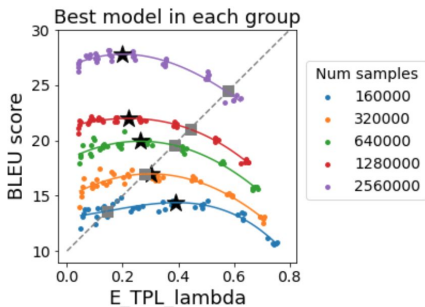
Model selection results using shape metrics



200 Transformers trained for neural machine translation.
BLEU score is better if it is larger.

Yang, Theisen, Hodgkinson, Gonzalez, Ramchandran, Martin, Mahoney, 2022, <https://arxiv.org/pdf/2202.02842.pdf>

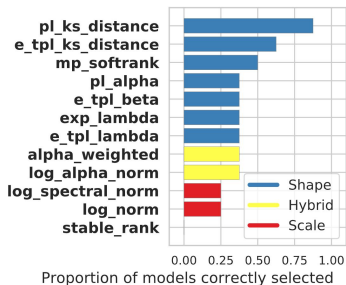
Model selection results using shape metrics



**200 Transformers trained for neural machine translation.
BLEU score is better if it is larger.**

Yang, Theisen, Hodgkinson, Gonzalez, Ramchandran, Martin, Mahoney, 2022, <https://arxiv.org/pdf/2202.02842.pdf>

Model selection results using shape metrics



Huggingface Transformers

Models
BERT-Tiny, BERT-Mini, BERT-Small, BERT-Medium, BERT-Base, BERT-Large
24 smaller BERT models (English only, uncased, trained with WordPiece masking)
GPT2, GPT2-medium, GPT2-large, GPT2-xl
ALBERT-base-v1, ALBERT-large-v1, ALBERT-xlarge-v1, ALBERT-xxlarge-v1
ALBERT-base-v2, ALBERT-large-v2, ALBERT-xlarge-v2, ALBERT-xxlarge-v2
T5-small, T5-base, T5-large
DialoGPT-small, DialoGPT-medium, DialoGPT-large
FlauBERT_small_cased, FlauBERT_base_cased, FlauBERT_large_cased
FunnelModel-small, FunnelModel-medium, FunnelModel-intermediate
FunnelModel-large, FunnelModel-xlarge

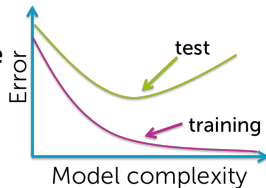
Yang, Theisen, Hodgkinson, Gonzalez, Ramchandran, Martin, Mahoney, 2022, <https://arxiv.org/pdf/2202.02842.pdf>

One issue of scale (norm-based) metrics

Model quality \neq Generalization gap?

Generalization gap = Train - Test

Model quality \approx Test-time performance



Generalization gap mentioned in the literature

“...More specifically, lower complexity should often imply smaller **generalization gap**.”

[Jiang et al. 2019]

“...~~great strides have been made towards identifying~~ notions of capacity that can be shown to formally control **generalization error**, $R_D(h) - \hat{R}_S(h)$ ”

[Dziugaite et al. 2020]

Predicting the **generalization gap** in deep networks with margin distributions,

[Jiang et al. 2018]

Towards task and architecture-independent **generalization gap** predictors,

[Yak et al. 2019]

“...The value of the output should ideally be larger for models that have larger **generalization gaps**.”

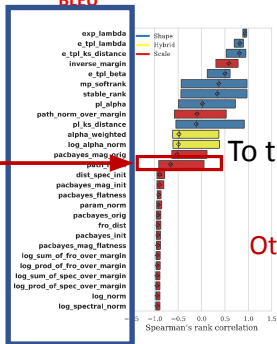
NeurIPS 2020 Competition: Predicting Generalization in Deep Learning (Version 1.1)

Model quality vs generalization gap

28 generalization measures

Each bar:
200 experiments

Correlations with
BLEU



Shape metrics

To the right means better

Other measures

Yang, Theisen, Hodgkinson, Gonzalez, Ramchandran, Martin, Mahoney, 2022, <https://arxiv.org/pdf/2202.02842.pdf>

Model quality vs generalization gap

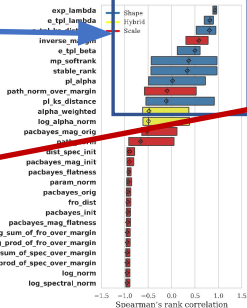
Model quality

- Shape metrics work better

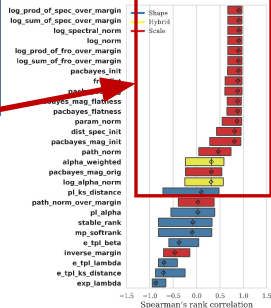
Generalization gap

- Existing measures work better

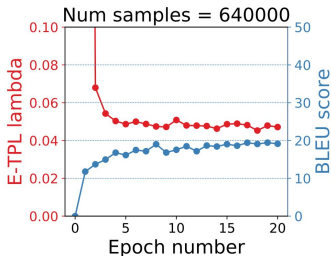
Correlations with
BLEU



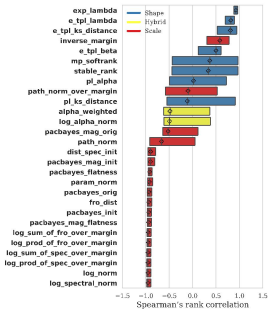
Correlations with **generalization gap**



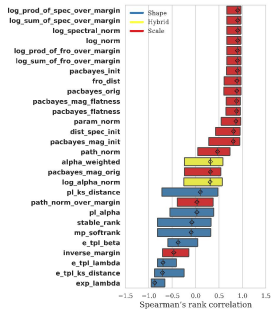
Time-wise correlation



Correlations with
BLEU



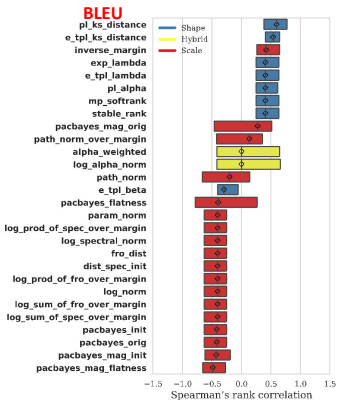
Correlations with **generalization gap**



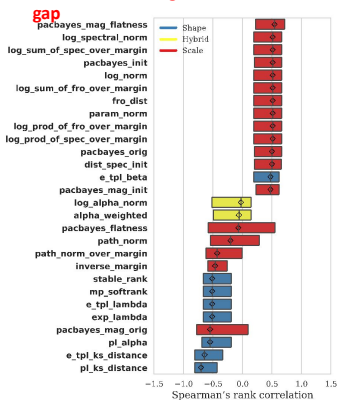
Yang, Theisen, Hodgkinson, Gonzalez, Ramchandran, Martin, Mahoney, 2022, <https://arxiv.org/pdf/2202.02842.pdf>

Correlation when changing learning rate

Correlations with



Correlations with **generalization**



Main conclusion for this section



Measures from statistical physics do not need data.



They predict model quality instead of generalization gap.

Model quality and generalization gap can be anti-correlated

Brief summary

- Model selection: measures that do not need data

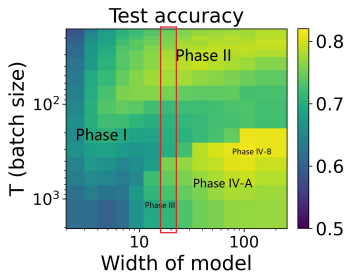


Which model should I use?



B

C



Phase transitions are great...but

Brief summary

- Model selection: measures that do not need data
- Shape metrics predict model quality



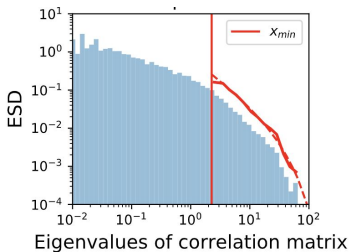
Which model should I use?



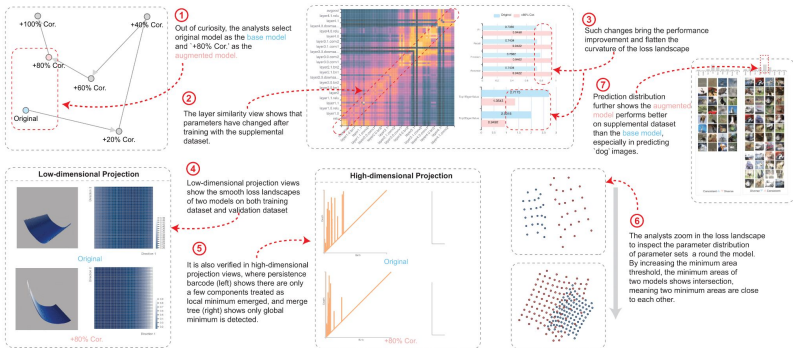
A

B

Model quality \neq Generalization gap



Visualizing loss landscape - LossLens



Slide shared by collaborator Tiankai Xie

Thanks!
Q&A

Outline

- 1 Weight Analysis and Heavy-Tailed Self-Regularization
- 2 Phenomenological Approach to Statistical Mechanics of Generalization
- 3 Using Heavy-Tailed Self-Regularization**
- 4 Random Matrix Theory for Modern ML
- 5 Putting It All Together
- 6 Conclusion

Outline

- 1 Introductory thoughts
- 2 Empirical results (to inform theory)
- 3 RMT and RMT-based Theory for Deep Learning
- 4 Using the Theory**
- 5 Expressing this in ML theory language (theorems!)
- 6 Conclusions

Using the theory

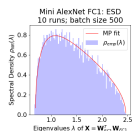
Different ways one could *use* a theory.

- Perform diagnostics for model validation, to develop hypotheses, etc.*
- Make predictions about model quality, generalization, transferability, etc.*
- Did post-training modifications damage my model?*
- Will buying more data help?*
- Will training longer help?*
- Will quantizing or distilling help?*
- Construct a regularizer to do model training.**

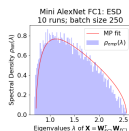
*Ideally, by peeking at very little or no data.

**If you have lots of data, lots of GPUs, etc.

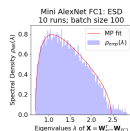
Batch Size Tuning: Exhibiting the Phases



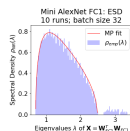
(a) Batch Size 500.



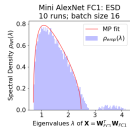
(b) Batch Size 250.



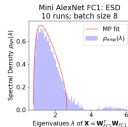
(c) Batch Size 100.



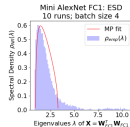
(d) Batch Size 32.



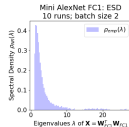
(e) Batch Size 16.



(f) Batch Size 8.



(g) Batch Size 4.



(h) Batch Size 2.

Figure: Varying Batch Size. ESD for Layer FC1 of MiniAlexNet. We exhibit all 5 of the main phases of training by varying only the batch size.

- **Decreasing** batch size **induces** strong correlations in \mathbf{W} , leading to a **more** implicitly-regularized model.
- **Increasing** batch size **washes out** strong correlations in \mathbf{W} , leading to a **less** implicitly-regularized model.

Predicting test accuracies ... lots of metrics ...

- **Average log norm** (a VC-like data-dependent capacity metric):

$$\langle \log \|\mathbf{W}\| \rangle = \frac{1}{N} \sum_{l,i} \log \|\mathbf{W}_{l,i}\| = \frac{1}{N} \sum_{l,i} \log(\lambda_{l,i}^{max})$$

- **Average alpha** (also data-dependent, from HT-SR theory):

$$\alpha = \frac{1}{N} \sum_{l,i} \alpha_{l,i}$$

- **Combine the two** into a weighted average (weighted to compensate for different size and scale of feature maps):

$$\hat{\alpha} = \frac{1}{N} \sum_{l,i} \log(\lambda_{l,i}^{max}) \alpha_{l,i}$$

- In a special case ($\alpha \approx 2$), for each layer:

$$\text{PL-Norm Relation: } \alpha \log \lambda^{max} \approx \log \|\mathbf{W}\|_F^2.$$

“pip install weightwatcher”

(The first) large-scale study (meta-analysis) of hundreds of SOTA pretrained models †

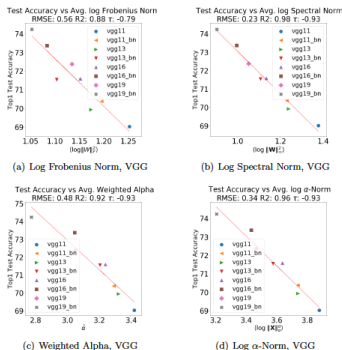


Figure 2: Comparison of Average Log Norm and Weighted Alpha quality metrics versus reported Top1 test accuracy for pretrained VGG models: VGG11, VGG13, VGG16, and VGG19, with and

Different metrics on **pre-trained VGG**.

Series	#	Metric	$(\log \ W\ _F^2)$	$(\log \ W\ _{\infty}^2)$	α	$(\log \ X\ _2^2)$
VGG	6	RMSE	0.56	0.23	0.48	0.34
		R^2	0.88	0.98	0.92	0.96
		Kendall- τ	-0.79	-0.93	-0.93	-0.93
ResNet	5	RMSE	0.9	0.97	0.61	0.66
		R^2	0.92	0.9	0.96	0.9
		Kendall- τ	-1.0	-1.0	-1.0	-1.0
ResNet-1K	19	RMSE	2.4	2.8	1.8	1.9
		R^2	0.81	0.74	0.89	0.88
		Kendall- τ	-0.79	-0.79	-0.89	-0.88
DenseNet	4	RMSE	0.3	0.11	0.16	0.21
		R^2	0.93	0.99	0.98	0.97
		Kendall- τ	-1.0	-1.0	-1.0	-1.0

Table 1: Quality metrics (for RMSE, smaller is better; for R^2 , larger is better; and for Kendall- τ rank correlation, larger magnitude is better) for reported Top1 test error for pretrained models in each architecture series. Column # refers to number of models. VGG, ResNet, and DenseNet were pretrained on ImageNet. ResNet-1K was pretrained on ImageNet-1K.

Summary statistics: **VGG; ResNet; DenseNet**.

	$\log \ \cdot \ _F^2$	$\log \ \cdot \ _{\infty}^2$	α	$\log \ \cdot \ _2^2$
RMSE (mean)	4.84	5.57	4.58	4.55
RMSE (std)	9.14	9.16	9.16	9.17
R^2 (mean)	3.9	3.85	3.89	3.89
R^2 (std)	9.34	9.36	9.34	9.34
Kendal-tau (mean)	3.84	3.77	3.86	3.85
Kendal-tau (std)	9.37	9.4	9.36	9.36

Table 3: Comparison of linear regression fits for different average Log Norm and Weighted Alpha metrics across 5 CV datasets, 17 architectures, covering 108 (out of over 400) different pretrained

Summary statistics: **hundreds of models**.

Lots more plots to prove we can “predict trends . . . without access . . .”

† “Predicting trends in the quality of state-of-the-art neural networks without access to training or testing data,” Martin,

Using a theory: on SOTA models

Analyzing pre-trained models.

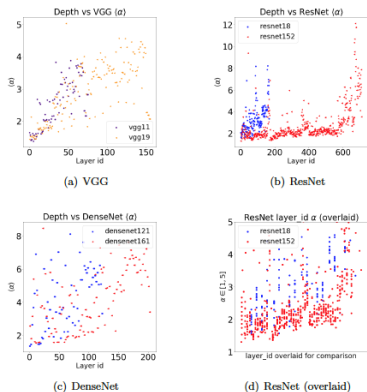


Figure 4: PL exponent (α) versus layer id, for the least and the most accurate models in VGG (a), ResNet (b), and DenseNet (c) series. (VGG is without BN; and note that the Y axes on

Alpha versus depth: VGG, ResNet, DenseNet.

Using a theory: on SOTA models

Analyzing pre-trained models.

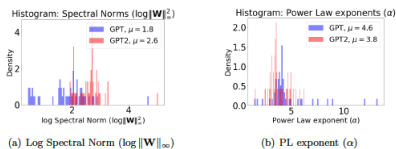


Figure 6: Histogram of PL exponents and Log Spectral Norms for weight matrices from the OpenAI GPT and GPT2-small pretrained models.

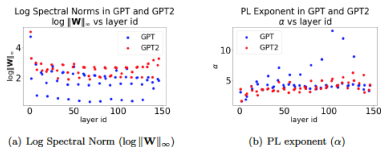


Figure 7: Log Spectral Norms (in (a)) and PL exponents (in (b)) for weight matrices from the OpenAI GPT and GPT2-small pretrained models. (Note that the quantities shown on each Y axis are different.) In the text, this is interpreted in terms of Scale Collapse and Correlation Flow.

Histogram and depth plots of $\alpha_{l,i}$ and $\lambda_{l,i}^{max}$.

Using a theory: easy to break popular SLT metrics

Easy to “break” popular SLT metrics

- they are *not* validated counterfactually
- they drive the development of models

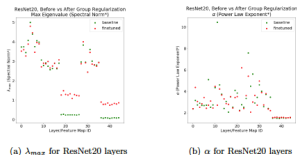


Figure 5: ResNet20, distilled with Group Regularization, as implemented in the `distiller` (`4D_regularized_5Lremoved`) pretrained models. Log Spectral Norm ($\log \lambda_{\max}$) and PL exponent (α) for individual layers, versus layer id, for both baseline (before distillation, green) and fine-tuned (after distillation, red) pretrained models.

Series	#	$(\log \ \mathbf{W}\ _F)$	$(\log \ \mathbf{W}\ _{\infty})$	$\bar{\alpha}$	$(\log \ \mathbf{X}\ _2)$
GPT	49	1.64	1.72	7.01	7.28
GPT2-small	49	2.04	2.54	9.62	9.87
GPT2-medium	98	2.08	2.58	9.74	10.01
GPT2-large	146	1.85	1.99	7.67	7.94
GPT2-xl	194	1.86	1.92	7.17	7.51

Table 2: Average value for the average Log Norm and Weighted Alpha metrics for pretrained OpenAI GPT and GPT2 models. Column # refers to number of layers treated. Averages do

GPTx series: how does a model trained to “bad” data differ from one trained to “good” data?

Intel’s distillation “broke” their models.

Using a theory: leads to predictions

Based on analyzing hundreds of pre-trained SOTA models:

- **“Correlation flow”**:
 - ▶ “Shape” of ESD of adjacent layers, as well as overlap between eigenvectors of adjacent layers, should be well-aligned.
- **“Scale collapse”**:
 - ▶ “Size” of ESD of one or more layers changes dramatically, while the size of other layers changes very little, as a function of some perturbation of a model, during training (or post-training modification).
- **“Correlation traps”**:
 - ▶ Spuriously large eigenvalues[§] may appear, and they may even be important for model convergence.

We can measure these quantities with Weightwatcher—so can you!

[§]Eigenvalues not due to signal in the data—we have theorems-style theory for Hessians (“Hessian Eigenspectra of More Realistic Nonlinear Models,” Liao and Mahoney, <https://arxiv.org/abs/2103.01519>), but it’s still open for Weights

More publicly-available data

A contest (Predicting Generalization in Deep Learning, NeurIPS 2020).

Our experiences:

- based on a “fantastic” paper (considered *many* metrics, but not α or $\hat{\alpha}$)
- nominally about *causes* of generalization; but, like most ML contests,
 - ▶ ensemblization—good way to win
 - ▶ information leakage—hard to avoid
 - ▶ augment data—good way to win
 - ▶ (But none of those tell us about generalization.)
- big difference between 0 error and ≈ 0 error
- not worth competing in*
- thanks to organizers for releasing data*

*since we want to understand causes of good model performance

Models and metrics

Models and tasks: can segment models by architecture parameters or solver parameters.

Series	#	(L)	Batch Sizes	Dropout	Weight Decay	Conv Widths
Task1 “task1_v4” (VGG-like)	0xx	4	8, 32, 512	0.0, 0.5	0.0, 0.001	256, 512
	1xx	5	8, 32, 512	0.0, 0.5	0.0, 0.001	256, 512
	2xx	5	8, 32, 512	0.0, 0.5	0.0, 0.001	256, 512
	5xx	8	8, 32, 512	0.0, 0.5	0.0, 0.001	256, 512
	6xx	8	8, 32, 512	0.0, 0.5	0.0, 0.001	256, 512
	7xx	9	8, 32, 512	0.0, 0.5	0.0, 0.001	256, 512
Task2 “task2_v1” (Network-in-network)	2xx	13	32, 512, 1024	0.0, 0.25, 0.5	0.0, 0.001	512
	6xx	7	32, 512, 1024	0.0, 0.25, 0.5	0.0, 0.001	512
	9xx	10	32, 512, 1024	0.0, 0.25, 0.5	0.0, 0.001	512
	10xx	10	32, 512, 1024	0.0, 0.25, 0.5	0.0, 0.001	512

Table 1: Overview of models (from [5, 6]) we considered.

Best-performing metrics.

Complexity Metric	Average	Ref.	Need access to data?	Need access to initial weights?	Need access to GPUs?
Alpha	$\langle \alpha \rangle$	(here)	No	No	No
QualityOfAlphaFit	D_{KS}	(here)	No	No	No
LogSpectralNorm	$\langle \log_{10} \ \mathbf{W}\ _2^2 \rangle$	([12])	No	No	No
LogFrobeniusNorm	$\langle \log_{10} \ \mathbf{W}\ _F^2 \rangle$	([12])	No	No	No
AlphaHat	$\hat{\alpha}$	([7])	No	No	No
LogAlphaShattenNorm	$\langle \log_{10} \ \mathbf{W}\ _{2\alpha}^{2\alpha} \rangle$	([7])	No	No	No
DistanceFromInit	Δ_{init}	([12])	No	Yes	No
TrainingAccuracy	N/A	N/A	Yes	No	No
Sharpness	N/A	([12])	Yes	No	Yes
SVDSmoothing	N/A	(here)	Yes	No	No

Table 2: Overview of model quality metrics. Based on our initial analysis of Contest models, we propose and demonstrate the quality of Alpha, QualityOfAlphaFit, and SVDSmoothing. For several of the metrics, we refer to a recent summary paper [12] rather than original references.

Size versus shape

Size (norm) and shape (fitted HT parameters) are different ...

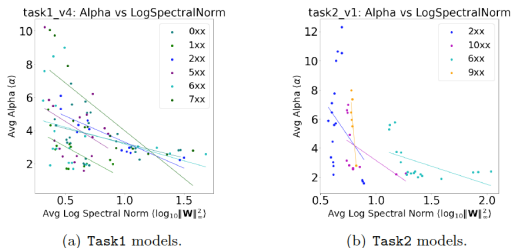


Figure 2: Comparison of the Alpha and LogSpectralNorm metrics, for Task1 and Task2 models.

... and there is a lot of **heterogeneity across tasks/subtasks**.

Extracting shape parameters from HT ESDs

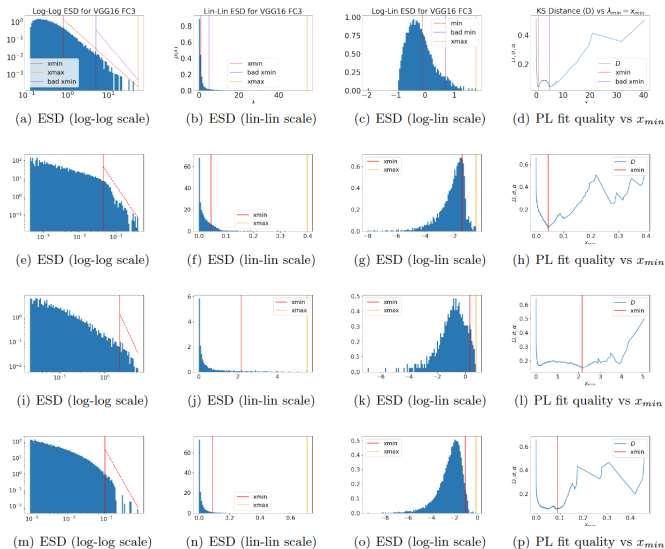


Figure 1: Illustration of the role of the shape of the ESD in determining the PL parameter α in

Training versus testing

Training and testing error often anti-correlated ...

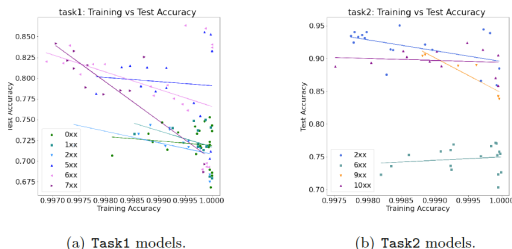


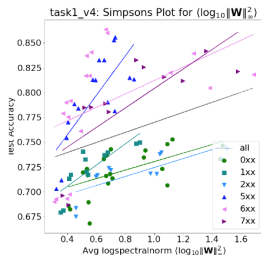
Figure 5: Relationship between training accuracy and testing accuracy for **Task1** and **Task2** models. One would expect a positive correlation or (if the training error is very close to zero) at least not a negative correlation. In many cases, they are strongly anti-correlated. See also Table 4.

... and there is a lot of **heterogeneity across tasks/subtasks**.

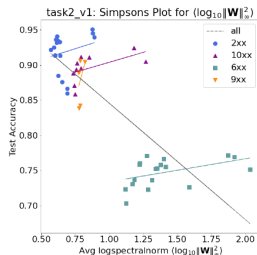
Simpson's paradox (1 of 2)

Within sub-group: vary solver parameters.

Between sub-groups: vary architecture.



(a) Task1.



(b) Task2.

Figure 6: Illustration of Simpson's paradox. Test accuracy versus LogSpectralNorm , for Task1 and Task2, segmented by model group. Note the overall trend is downward (line not explicitly shown), while the trend for each subgroup is upward. This is especially prominent for Task2.

LogSpectralNorm for better models is:

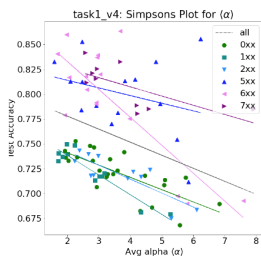
Task1: larger within *and* between sub-groups.

Task2: larger within—*and smaller between*—sub-groups.

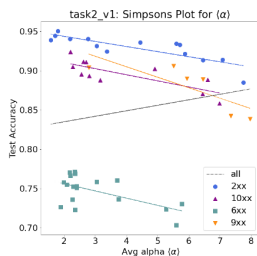
Simpson's paradox (2 of 2)

Within sub-group: vary solver parameters.

Between sub-groups: vary architecture.



(a) Task1.



(b) Task2.

Figure 7: Illustration of Simpson's paradox. Test accuracy versus Alpha, for Task1 and Task2,

Alpha for better models is:

Task1: smaller within *and* between sub-groups.

Task2: smaller within sub-groups—*but larger between sub-groups*.

Lessons learned ...

Extracting causal insight?

- Don't invent causal metrics.
- Don't look for “one size fits all” metric.
- We identified Simpson's paradoxes—and then we used them and domain knowledge to identify causes of good performance.
- A cautionary tale ...

Size versus shape more generally:

- Construct data-dependent versions of size versus shape.
- SVDSmoothing—if training data fit exactly, feed data through low-rank approximation. (No GPUs!)

What more can we do?

Future directions (*all of which demand a practical theory*):

- Training/testing curves gives limited insight:
 - ▶ don't take into account hyperparameter fiddling;
 - ▶ don't correlate with robustness/accuracy/fairness/etc.
- No access to data / optimization protocols / hyperparameter values / etc.:
 - ▶ can I evaluate systems-motivated model adjustments?
 - ▶ batch size, edge, distillation, etc. (without training/retraining)?
- Model user is not a model developer:
 - ▶ sanity check: did you give me a bad/damaged model?
 - ▶ robustness check: can I look for backdoor adversarial attacks, etc.?
- Data costs money:
 - ▶ Do I have enough data?
 - ▶ Should I spend money on analysts or machines or data?

If AI/ML is to become an industrial process, beyond FAAMG, it will have to be compartmentalized to scale: Group-A develops; Group-B validates; and Group-C deploys

Conclusions

“Practical theory” is not an oxymoron:

- not all theory is practical, but some is

“Practical theory” is theory for practical things:

- like data
- like SOTA DNNs

“Practical theory” can be used to address practical questions:

- is my network fully optimized?
- should I buy more data?
- can I use labels and/or domain knowledge more efficiently?
- can I design better ensembles, or improve model post-modification?
- is my pre-trained SOTA DNN overparameterized or underparameterized?

*If you want more ... “**pip install weightwatcher**” ...*

Outline

- 1 Weight Analysis and Heavy-Tailed Self-Regularization
- 2 Phenomenological Approach to Statistical Mechanics of Generalization
- 3 Using Heavy-Tailed Self-Regularization
- 4 Random Matrix Theory for Modern ML**
- 5 Putting It All Together
- 6 Conclusion

Random Matrix Theory for Machine Learning: new intuitions, improved methods, and beyond

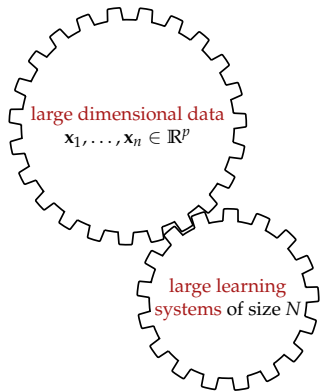
Michael. W. Mahoney

March 20, 2023

Outline

- 1 Introduction
- 2 Sample covariance matrix for large dimensional data: from LNN to modern RMT
- 3 RMT for Modern Machine Learning: linear models
- 4 RMT for Modern Machine Learning: nonlinear models
 - A random matrix perspective of the “curse of dimensionality”
 - Kernel spectral clustering for large dimensional data
 - A random matrix approach to large neural networks and random features

Understanding the mechanism of large dimensional machine learning



- ▶ Big Data era: exploit large n, p, N
- ▶ **counterintuitive** phenomena when $n \gg p$, e.g., the “*curse of dimensionality*”
- ▶ complete **change** of understanding of many ML algorithms
- ▶ Random Matrix Theory provides the tools!

From low to high dimensional machine learning

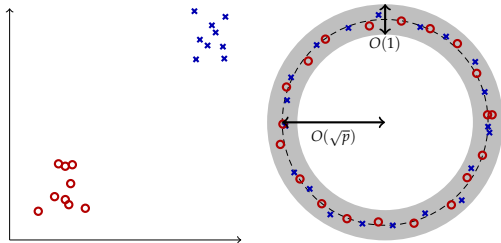


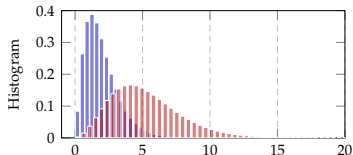
Figure: Visual representation of classification in (left) small and (right) large dimensions.

- ▶ **low dimension:** data vectors $\mathbf{x}_i \in \mathbb{R}^p, p = 2, 3$, gathered in different “groups” can be classified using **distance-based** approach
- ▶ **high dimension:**
 - (i) **easy** or **trivial** scenario where low dimensional intuition holds and a **pairwise** distance-based classification approach via, e.g., Johnson–Lindenstrauss lemma, is efficient;
 - (ii) **hard** or **non-trivial** scenario where such intuition **collapses**: data vectors at approximately the **same** Euclidean distance, **regardless** their arising from *same or different* classes.

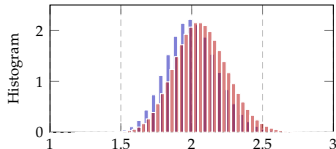
Non-trivial high dimensional classification beyond the JL regime

In the high dimensional regime where data dimension p and sample size n both large, a **dual** phenomenon:

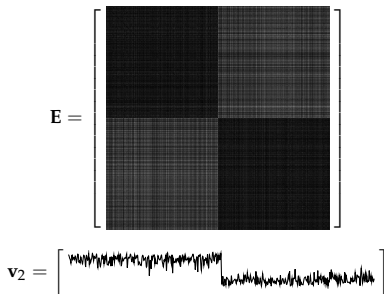
- (i) data points **not pairwise classifiable**: Euclidean distance between **any** two data points $\mathbf{x}_i \in \mathcal{C}_a$ and $\mathbf{x}_j \in \mathcal{C}_b$ approximately constant $\approx \tau = O(1)$ independent of their classes $\mathcal{C}_a, \mathcal{C}_b$: $\|\mathbf{x}_i - \mathbf{x}_j\|^2/p = \tau + o(1)$ as $n, p \rightarrow \infty$ and data pairs *neither close nor far* from each other;
- (ii) classification remains possible by exploiting the **spectral** information of large Euclidean distance **matrix** $\mathbf{E} = \{\|\mathbf{x}_i - \mathbf{x}_j\|^2/p\}_{i,j=1}^n$, thanks to a **collective** behavior of all data belonging to same (and **large**) classes.



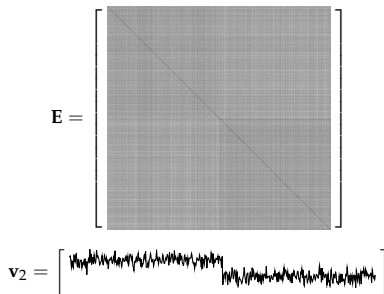
(a) $p = 5$



(b) $p = 250$



(a) $p = 5$



(b) $p = 250$

Figure: Euclidean distance matrices \mathbf{E} , the histogram of the entries of \mathbf{E} , and the second top eigenvectors \mathbf{v}_2 , for small (**left**, $p = 5$) and large (**right**, $p = 250$) dimensional data $\mathbf{X} = [\mathbf{x}_1, \dots, \mathbf{x}_n] \in \mathbb{R}^{p \times n}$ with $\mathbf{x}_1, \dots, \mathbf{x}_{n/2} \in \mathcal{C}_1$ and $\mathbf{x}_{n/2+1}, \dots, \mathbf{x}_n \in \mathcal{C}_2$ for $n = 5000$ and different values of p .

Four ways to characterize sample covariance matrices

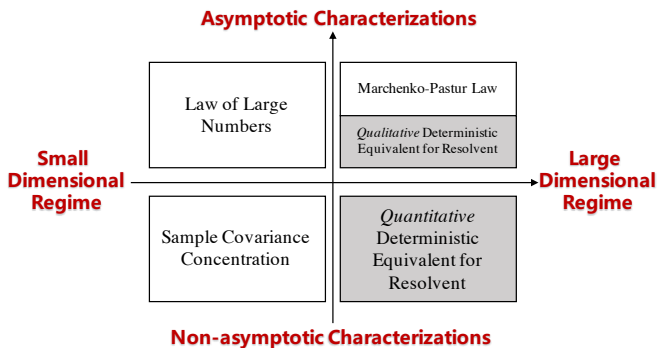


Figure: Different ways to characterize the sample covariance matrix $\hat{\mathbf{C}} = \frac{1}{n}\mathbf{X}\mathbf{X}^T$.

Small-dimensional characterizations of SCM

Theorem (Asymptotic Law of Large Numbers for SCM)

Let p be fixed, and let $\mathbf{X} \in \mathbb{R}^{p \times n}$ be a random matrix with independent sub-gaussian columns $\mathbf{x}_i \in \mathbb{R}^p$ such that $\mathbb{E}[\mathbf{x}_i] = \mathbf{0}$ and $\mathbb{E}[\mathbf{x}_i \mathbf{x}_i^\top] = \mathbf{I}_p$. Then one has,

$$\|\hat{\mathbf{C}} - \mathbf{I}_p\|_2 \rightarrow 0, \quad (1)$$

almost surely, as $n \rightarrow \infty$.

- ▶ the sub-gaussianity is in fact **not** necessary, and is stated to align with the following result

Theorem (Concentration of sample covariance, [Ver18, Theorem 4.6.1])

Let $\mathbf{X} \in \mathbb{R}^{p \times n}$ be a random matrix with i.i.d. sub-gaussian columns $\mathbf{x}_i \in \mathbb{R}^p$ such that $\mathbb{E}[\mathbf{x}_i] = \mathbf{0}$ and $\mathbb{E}[\mathbf{x}_i \mathbf{x}_i^\top] = \mathbf{I}_p$, one has, with probability at least $1 - 2 \exp(-t^2)$ for any $t \geq 0$ that

$$\|\hat{\mathbf{C}} - \mathbf{I}_p\|_2 \leq C_1 \max(\delta, \delta^2), \quad \delta = C_2(\sqrt{p/n} + t/\sqrt{n}) \quad (2)$$

for some constants $C_1, C_2 > 0$ independent of n, p .

- ▶ non-asymptotic and high probability characterization
- ▶ however, **not precise** in the $p \sim n$ regime, since $\delta = O(\sqrt{p/n}) = O(1)$

Sample covariance matrix in the large n, p regime

- ▶ For $\mathbf{x}_i \sim \mathcal{N}(\mathbf{0}, \mathbf{C})$, estimate **population covariance** $\mathbf{C} \in \mathbb{R}^{p \times p}$ from n data samples $\mathbf{X} = [\mathbf{x}_1, \dots, \mathbf{x}_n] \in \mathbb{R}^{p \times n}$.
- ▶ Maximum likelihood sample covariance matrix with **entry-wise** convergence

$$\hat{\mathbf{C}} = \frac{1}{n} \sum_{i=1}^n \mathbf{x}_i \mathbf{x}_i^T = \frac{1}{n} \mathbf{X} \mathbf{X}^T \in \mathbb{R}^{p \times p}, \quad [\hat{\mathbf{C}}]_{ij} \rightarrow [\mathbf{C}]_{ij}$$

almost surely as $n \rightarrow \infty$: optimal for $n \gg p$ (or, for p “small”).

- ▶ In the regime $n \sim p$, conventional wisdom breaks down:
for $\mathbf{C} = \mathbf{I}_p$ with $n < p$, $\hat{\mathbf{C}}$ has at least $p - n$ **zero eigenvalues**.

$$\|\hat{\mathbf{C}} - \mathbf{C}\| \not\rightarrow 0, \quad n, p \rightarrow \infty$$

\Rightarrow eigenvalue **mismatch** and **not** consistent! \Rightarrow **matrix norms not equivalent in large dimensions!**

- ▶ due to $\|\mathbf{A}\|_\infty \leq \|\mathbf{A}\| \leq p \|\mathbf{A}\|_\infty$ for $\mathbf{A} \in \mathbb{R}^{p \times p}$ and $\|\mathbf{A}\|_\infty \equiv \max_{ij} |\mathbf{A}_{ij}|$.

Large-dimensional characterizations of SCM: eigenvalues

Definition (Empirical Spectral Distribution, ESD)

For a symmetric matrix $\mathbf{X} \in \mathbb{R}^{p \times p}$, the *empirical spectral distribution (ESD)* or *empirical spectral measure* $\mu_{\mathbf{X}}$ of \mathbf{X} is defined as the normalized counting measure of the eigenvalues $\lambda_1(\mathbf{X}), \dots, \lambda_p(\mathbf{X})$ of \mathbf{X} ,

$$\mu_{\mathbf{X}} \equiv \frac{1}{p} \sum_{i=1}^p \delta_{\lambda_i(\mathbf{X})}, \quad (3)$$

where δ_x represents the Dirac measure at x .

Theorem (Marčenko-Pastur law, [MP67])

Under the same setting, as $n, p \rightarrow \infty$ with $p/n \rightarrow c \in (0, \infty)$, with probability one, the empirical spectral measure $\mu_{\hat{\mathbf{C}}} \equiv \frac{1}{p} \sum_{i=1}^p \delta_{\lambda_i(\hat{\mathbf{C}})}$ of $\hat{\mathbf{C}} \equiv \frac{1}{n} \mathbf{X} \mathbf{X}^T$ converges weakly to a probability measure μ given explicitly by

$$\mu(dx) = (1 - c^{-1})^+ \delta_0(x) + \frac{1}{2\pi c x} \sqrt{(x - E_-)^+ (E_+ - x)^+} dx \quad (4)$$

where $E_{\pm} = (1 \pm \sqrt{c})^2$ and $(x)^+ = \max(0, x)$, and is known as the *Marčenko-Pastur law*.

Large-dimensional behaviors diverge from small-dimensional behaviors

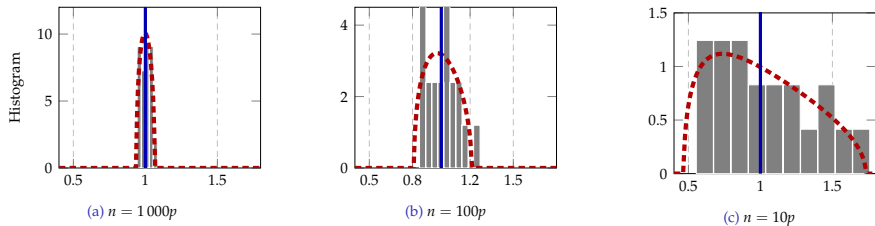


Figure: Histogram of the eigenvalues of \hat{C} versus the limiting Marčenko-Pastur law in Theorem 4, for X having standard Gaussian entries with $p = 20$ and different $n = 1000p, 100p, 10p$ from left to right.

Large-dimensional behaviors diverge from small-dimensional behaviors

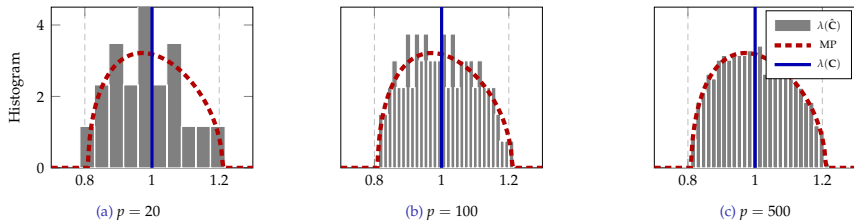


Figure: Histogram of the eigenvalues of $\hat{\mathbf{C}}$ versus the Marčenko-Pastur law, for \mathbf{X} having standard Gaussian entries with $n = 100p$ and different $p = 20, 100, 500$ from left to right.

When is one in the random matrix regime? Almost always!

What about $n = 100p$? For $\mathbf{C} = \mathbf{I}_p$, as $n, p \rightarrow \infty$ with $p/n \rightarrow c \in (0, \infty)$: the Marčenko–Pastur law

$$\mu(dx) = (1 - c^{-1})^+ \delta(x) + \frac{1}{2\pi cx} \sqrt{(x - E_-)^+ (E_+ - x)^+} dx$$

where $E_- = (1 - \sqrt{c})^2$, $E_+ = (1 + \sqrt{c})^2$ and $(x)^+ \equiv \max(x, 0)$. **Close match!**

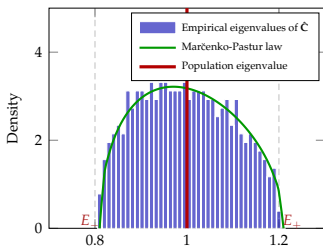


Figure: Eigenvalue distribution of $\hat{\mathbf{C}}$ versus Marčenko–Pastur law, $p = 500$, $n = 50000$.

- ▶ eigenvalues span on $[E_- = (1 - \sqrt{c})^2, E_+ = (1 + \sqrt{c})^2]$.
- ▶ for $\mathbf{n} = 100\mathbf{p}$, on a range of $\pm 2\sqrt{c} = \pm 0.2$ around the **population eigenvalue 1**.

Beyond eigenvalue distribution: a modern RMT approach via the resolvent

Definition (Resolvent)

For a symmetric matrix $\mathbf{X} \in \mathbb{R}^{n \times n}$, the resolvent $\mathbf{Q}_{\mathbf{X}}(z)$ of \mathbf{X} is defined, for $z \in \mathbb{C}$ not an eigenvalue of \mathbf{X} , as

$$\mathbf{Q}_{\mathbf{X}}(z) \equiv (\mathbf{X} - z\mathbf{I}_p)^{-1}. \quad (5)$$

The matrix $\mathbf{Q}_{\mathbf{X}}(z)$ will often simply be denoted $\mathbf{Q}(z)$ when there is no ambiguity.

Table: A list of different matrix functionals and how they can be evaluated via the resolvent.

Objects of interest	Functionals of resolvent $\mathbf{Q}_{\mathbf{X}}(z)$
ESD $\mu_{\mathbf{X}}$ of \mathbf{X}	Stieltjes transform $m_{\mu_{\mathbf{X}}}(z) = \frac{1}{p} \text{tr} \mathbf{Q}_{\mathbf{X}}(z)$ as the trace of $\mathbf{Q}_{\mathbf{X}}$
Linear spectral statistics (LSS): $f_{\mathbf{X}} \equiv \frac{1}{p} \sum_i f(\lambda_i(\mathbf{X}))$	Integration of trace of $\mathbf{Q}_{\mathbf{X}}$ (via Cauchy's integral) $-\frac{1}{2\pi i} \oint_{\Gamma} f(z) \frac{1}{p} \text{tr} \mathbf{Q}_{\mathbf{X}}(z) dz$
Projections of eigenvectors $\mathbf{v}^{\top} \mathbf{u}$ and $\mathbf{v}^{\top} \mathbf{U}$ onto some given $\mathbf{v} \in \mathbb{R}^p$	Bilinear form $\mathbf{v}^{\top} \mathbf{Q}_{\mathbf{X}}(z) \mathbf{v}$ of $\mathbf{Q}_{\mathbf{X}}$
General matrix functional $F(\mathbf{X}) = \sum_i f(\lambda_i(\mathbf{X})) \mathbf{v}_1^{\top} \mathbf{u}_i \mathbf{u}_i^{\top} \mathbf{v}_2$ involving both eigenvalues and eigenvectors of \mathbf{X}	Integration of bilinear form of $\mathbf{Q}_{\mathbf{X}}(z)$ $-\frac{1}{2\pi i} \oint_{\Gamma} f(z) \mathbf{v}_1^{\top} \mathbf{Q}_{\mathbf{X}}(z) \mathbf{v}_2 dz$

Old and new school RMT

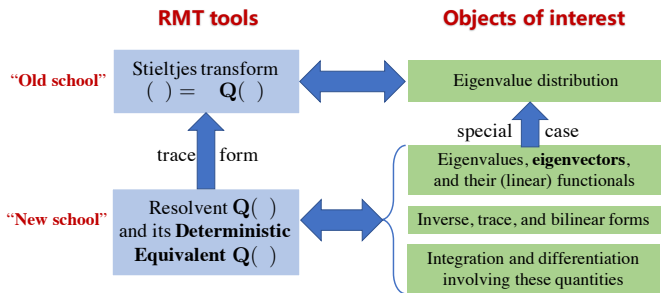


Figure: Different objects of interest and their corresponding technical tools for “old” and “new school” RMT.

Modern RMT: deterministic equivalents for resolvent

Definition (Deterministic Equivalent)

We say that $\bar{\mathbf{Q}} \in \mathbb{R}^{p \times p}$ is an $(\varepsilon_1, \varepsilon_2, \delta)$ -deterministic equivalent for the symmetric random matrix $\mathbf{Q} \in \mathbb{R}^{p \times p}$ if, for deterministic matrix $\mathbf{A} \in \mathbb{R}^{p \times p}$ and vectors $\mathbf{a}, \mathbf{b} \in \mathbb{R}^p$ of unit norms (spectral and Euclidean, respectively), we have, with probability at least $1 - \delta(p)$ that

$$\left| \frac{1}{p} \operatorname{tr} \mathbf{A}(\mathbf{Q} - \bar{\mathbf{Q}}) \right| \leq \varepsilon_1(p), \quad \left| \mathbf{a}^\top (\mathbf{Q} - \bar{\mathbf{Q}}) \mathbf{b} \right| \leq \varepsilon_2(p), \quad (6)$$

for some non-negative functions $\varepsilon_1(p), \varepsilon_2(p)$ and $\delta(p)$ that decrease to zero as the dimension $p \rightarrow \infty$.

- ▶ *non-asymptotic* and holds for any p ,
- ▶ taking $p \rightarrow \infty$ leads to
 - (i) $\frac{1}{p} \operatorname{tr} \mathbf{A}(\mathbf{Q} - \bar{\mathbf{Q}}) \rightarrow 0, \mathbf{a}^\top (\mathbf{Q} - \bar{\mathbf{Q}}) \mathbf{b} \rightarrow 0$ in probability as $p \rightarrow \infty$; and
 - (ii) if the failure probability $\delta(p) = O(p^{-\ell})$ for some $\ell > 1$, by Borel–Cantelli lemma $\frac{1}{p} \operatorname{tr} \mathbf{A}(\mathbf{Q} - \bar{\mathbf{Q}}) \rightarrow 0, \mathbf{a}^\top (\mathbf{Q} - \bar{\mathbf{Q}}) \mathbf{b} \rightarrow 0$ almost surely as $p \rightarrow \infty$.
- ▶ to denote this **asymptotic** deterministic equivalent relation, use

$$\mathbf{Q} \leftrightarrow \bar{\mathbf{Q}}. \quad (7)$$

An asymptotic deterministic equivalent for SCM resolvent

Theorem (An asymptotic deterministic equivalent for resolvent, [CL22, Theorem 2.4])

Let $\mathbf{X} \in \mathbb{R}^{p \times n}$ be a random matrix having i.i.d. sub-gaussian entries of zero mean and unit variance, and denote $\mathbf{Q}(z) = (\frac{1}{n}\mathbf{X}\mathbf{X}^T - z\mathbf{I}_p)^{-1}$ the resolvent of $\frac{1}{n}\mathbf{X}\mathbf{X}^T$ for $z \in \mathbb{C}$ not an eigenvalue of $\frac{1}{n}\mathbf{X}\mathbf{X}^T$. Then, as $n, p \rightarrow \infty$ with $p/n \rightarrow c \in (0, \infty)$, the (sequence of) deterministic matrix $\bar{\mathbf{Q}}(z)$ is a Deterministic Equivalent of the (sequence of) random resolvent matrix $\mathbf{Q}(z)$, i.e.,

$$\mathbf{Q}(z) \leftrightarrow \bar{\mathbf{Q}}(z), \quad \bar{\mathbf{Q}}(z) = m(z)\mathbf{I}_p, \quad (8)$$

with $m(z)$ the unique valid Stieltjes transform as solution to

$$czm^2(z) - (1 - c - z)m(z) + 1 = 0. \quad (9)$$

Theorem (A non-asymptotic deterministic equivalent for resolvent, [Der+21, Proposition 4])

Under the same setting and notations as above with $z < 0$, there exists some universal constant $C_1, C_2 > 0$ depending only on the sub-gaussian norm of the entries of \mathbf{X} and $|z|$, such that for any $\varepsilon \in (0, 1)$, if $n \geq (C_1 + \varepsilon)p$, one has

$$\|\mathbb{E}[\mathbf{Q}(z)] - \bar{\mathbf{Q}}(z)\|_2 \leq \frac{C_2}{\varepsilon} \cdot n^{-\frac{1}{2}}, \quad \bar{\mathbf{Q}}(z) = m(z)\mathbf{I}_p. \quad (10)$$

RMT for ML: linear models

Table: Roadmap of RMT for large-dimensional linear ML models .

Problem	Small dimension	Large dimension
Low rank approximation $\hat{\mathbf{X}}$ of info-plus-noise matrix \mathbf{X}	smooth decay of $\ \mathbf{X} - \hat{\mathbf{X}}\ _2 / \ \mathbf{X}\ _2 \simeq (1 + \ell)^{-1}$	sharp transition of $\ \mathbf{X} - \hat{\mathbf{X}}\ _2 / \ \mathbf{X}\ _2$ at $\ell = c + \sqrt{c}$
Classification of binary Gaussian mixtures of distance in means $\Delta\mu$	pairwise \simeq spectral approach	pairwise \ll spectral approach
Linear least squares regression risk as $n \uparrow$	bias = 0 and variance $\propto n^{-1}$	monotonic bias and non-monotonic variance

Low-rank approximation

Proposition (Relative spectral error of low-rank approximation)

Let $\mathbf{X} = \ell \mathbf{u}\mathbf{u}^\top + \frac{1}{n} \mathbf{Z}\mathbf{Z}^\top \in \mathbb{R}^{p \times n}$ be an additive spiked random matrix, for $\mathbf{u} \in \mathbb{R}^p$ some deterministic signal of unit norm, i.e., $\|\mathbf{u}\| = 1$, $\ell \geq 0$ the informative “signal strength”, and \mathbf{Z} having i.i.d. sub-gaussian entries of zero mean and unit variance, and let $\hat{\mathbf{X}} = \lambda_1(\mathbf{X}) \hat{\mathbf{u}}_1 \hat{\mathbf{u}}_1^\top$ the optimal rank-one approximation of \mathbf{X} given by its top eigenvalue-eigenvector pair $(\lambda_1(\mathbf{X}), \hat{\mathbf{u}}_1)$. Then, one has,

(i) in the **small-dimensional** regime, for p fixed and $n \rightarrow \infty$ that

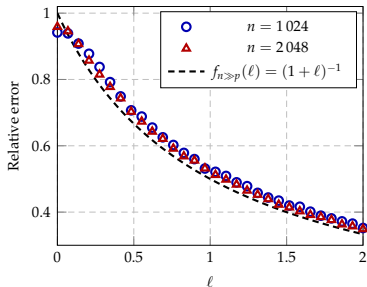
$$\frac{\|\mathbf{X} - \hat{\mathbf{X}}\|_2}{\|\mathbf{X}\|_2} \rightarrow f_{n \gg p}(\ell) \equiv \frac{1}{1 + \ell}, \quad (11)$$

almost surely; and

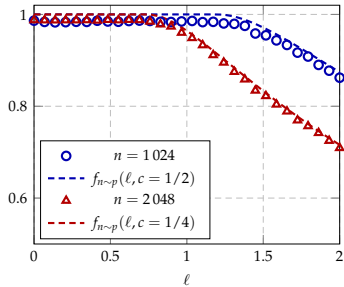
(ii) in the **large-dimensional** regime, as $n, p \rightarrow \infty$ with $p/n \rightarrow c \in (0, \infty)$ that

$$\frac{\|\mathbf{X} - \hat{\mathbf{X}}\|_2}{\|\mathbf{X}\|_2} \rightarrow f_{n \sim p}(\ell, c) \equiv \begin{cases} \frac{(1 + \sqrt{c})^2}{1 + \ell + \frac{c}{\ell - c}}, & \ell > c + \sqrt{c} \\ 1, & \ell \leq c + \sqrt{c} \end{cases} \quad (12)$$

almost surely.



(a) $p = 4$



(b) $p = 512$

Figure: Relative error in spectral norm $\|\mathbf{X} - \hat{\mathbf{X}}\|_2 / \|\mathbf{X}\|_2$ of rank-one approximation, for additive model $\mathbf{X} = \ell \mathbf{u}\mathbf{u}^\top + \frac{1}{n} \mathbf{Z}\mathbf{Z}^\top$ with standard Gaussian $\mathbf{Z}_{ij} \sim \mathcal{N}(0, 1)$, $\|\mathbf{u}\| = 1$, and $\hat{\mathbf{X}} = \lambda_1(\mathbf{X}) \hat{\mathbf{u}}_1 \hat{\mathbf{u}}_1^\top$ the optimal rank-one approximation of \mathbf{X} , as a function of ℓ for $p = 4$ and $p = 512$, $n = 1024$ and 2048 . Results averaged over 30 independent runs.

Classification

Proposition (Fundamental limits in classification: pairwise versus spectral approach)

For Gaussian mixture classification between $\mathcal{N}(\boldsymbol{\mu}_1, \mathbf{I}_p)$ and $\mathcal{N}(\boldsymbol{\mu}_2, \mathbf{I}_p)$ with $\Delta\boldsymbol{\mu} = \boldsymbol{\mu}_1 - \boldsymbol{\mu}_2$, one has, for some constant $C > 0$ independent of p ,

- (i) based on a pairwise (Euclidean) distance comparison approach, one is able to separate binary Gaussian mixtures satisfying $\|\Delta\boldsymbol{\mu}\| \geq Cp^{1/4}$; and
- (ii) based on an eigenspectral approach, one is able to separate a closer distance of $\|\Delta\boldsymbol{\mu}\| \geq C$, which is, up to a constant factor, the minimum distance possible.

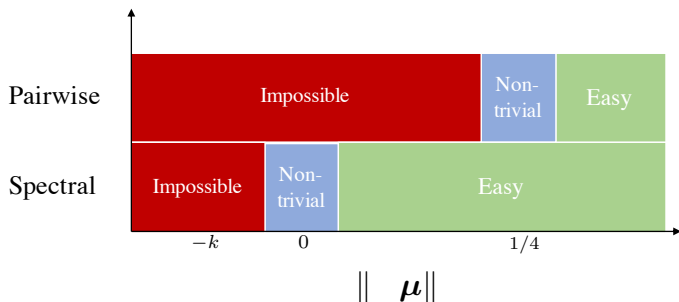


Figure: Illustration of different regimes in separating a binary GMM based on the distance in means $\|\Delta\mu\|$, for both pairwise distance comparison and spectral approaches.

“Curse of dimensionality”: loss of relevance of Euclidean distance

- ▶ Binary Gaussian mixture classification $\mathbf{x} \in \mathbb{R}^p$:

$$\mathcal{C}_1 : \mathbf{x} \sim \mathcal{N}(\boldsymbol{\mu}_1, \mathbf{C}_1), \text{ versus } \mathcal{C}_2 : \mathbf{x} \sim \mathcal{N}(\boldsymbol{\mu}_2, \mathbf{C}_2);$$

- ▶ Neyman-Pearson test: classification is possible **only** when [CLM18]

$$\|\boldsymbol{\mu}_1 - \boldsymbol{\mu}_2\| \geq C_\mu, \text{ or } \|\mathbf{C}_1 - \mathbf{C}_2\| \geq C_C \cdot p^{-1/2}$$

for some constants $C_\mu, C_C > 0$.

- ▶ In this **non-trivial** setting, for $\mathbf{x}_i \in \mathcal{C}_a, \mathbf{x}_j \in \mathcal{C}_b$:

$$\max_{1 \leq i \neq j \leq n} \left\{ \frac{1}{p} \|\mathbf{x}_i - \mathbf{x}_j\|^2 - \frac{2}{p} \text{tr} \mathbf{C}^\circ \right\} \xrightarrow{a.s.} 0$$

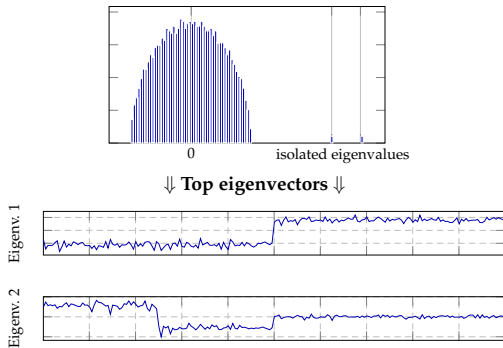
as $n, p \rightarrow \infty$ (i.e., $n \sim p$), for $\mathbf{C}^\circ \equiv \frac{1}{2}(\mathbf{C}_1 + \mathbf{C}_2)$, **regardless** of the classes $\mathcal{C}_a, \mathcal{C}_b$! (In fact even for $n = p^m$.)

⇒ Direct consequence to various **distance-based** machine learning methods (e.g., kernel spectral clustering)!

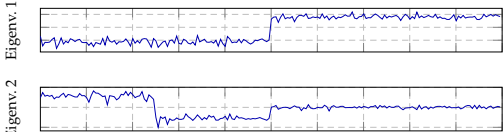
¹Romain Couillet, Zhenyu Liao, and Xiaoyi Mai. “Classification asymptotics in the random matrix regime”. In: *2018 26th European Signal Processing Conference (EUSIPCO)*. IEEE, 2018, pp. 1875–1879

Reminder on kernel spectral clustering

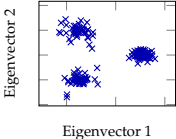
Two-step classification of n data points based on distance kernel matrix $\mathbf{K} \equiv \{f(\|\mathbf{x}_i - \mathbf{x}_j\|^2/p)\}_{i,j=1}^n$:



Reminder on kernel spectral clustering



↓ **K-dimensional representation** ↓



↓

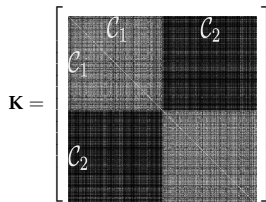
EM or k-means clustering.
(Three classes/clusters in this example.)

Visualization of kernel matrices for large dimensional Gaussian data

Objective: “cluster” Gaussian data $\mathbf{x}_1, \dots, \mathbf{x}_n \in \mathbf{R}^p$ into \mathcal{C}_1 or \mathcal{C}_2 .

Consider Gaussian kernel matrix $\mathbf{K}_{ij} = \exp(-\|\mathbf{x}_i - \mathbf{x}_j\|^2/2p)$ and the second top eigenvectors \mathbf{v}_2 for small (**left**) and large (**right**) dimensional data.

(a) $p = 5, n = 500$



(b) $p = 250, n = 500$

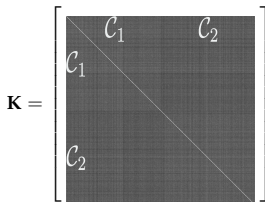
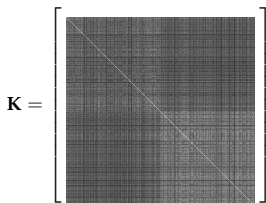
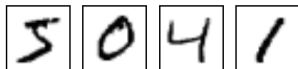


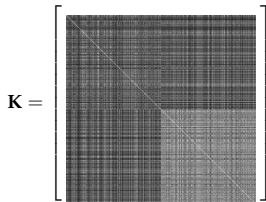
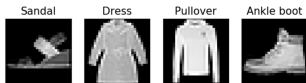
Figure: Kernel matrices \mathbf{K} and the second top eigenvectors \mathbf{v}_2 for small (left, $p = 5, n = 500$) and large (right, $p = 250, n = 500$) dimensional data.

Kernel matrices for large dimensional real-world data

(a) MNIST



(b) Fashion-MNIST



A spectral viewpoint of large kernel matrices in large dimensions

- ▶ “local” **linearization** of *nonlinear* kernel matrices in large dimensions, e.g., Gaussian kernel matrix $\mathbf{K}_{ij} = \exp(-\|\mathbf{x}_i - \mathbf{x}_j\|^2/2p)$ with $\mathbf{C}_1 = \mathbf{C}_2 = \mathbf{I}_p$ (e.g., $\mathcal{C}_1 : \mathbf{x}_i = \boldsymbol{\mu}_1 + \mathbf{z}_i$ versus $\mathcal{C}_2 : \mathbf{x}_j = \boldsymbol{\mu}_2 + \mathbf{z}_j$) so that

$$\|\mathbf{x}_i - \mathbf{x}_j\|^2/p \xrightarrow{a.s.} 2, \text{ and } \mathbf{K} = \exp\left(-\frac{2}{2}\right) \left(\mathbf{1}_n \mathbf{1}_n^\top + \frac{1}{p} \mathbf{Z}^\top \mathbf{Z}\right) + g(\|\boldsymbol{\mu}_1 - \boldsymbol{\mu}_2\|) \frac{1}{p} \mathbf{j} \mathbf{j}^\top + * + o_{\|\cdot\|}(1)$$

with Gaussian matrix $\mathbf{Z} = [\mathbf{z}_1, \dots, \mathbf{z}_n] \in \mathbb{R}^{p \times n}$ and $\mathbf{j} = [\mathbf{1}_{n/2}; -\mathbf{1}_{n/2}]$, the class-information vector

- ▶ **accumulated effect** of small “hidden” statistical information ($\|\boldsymbol{\mu}_1 - \boldsymbol{\mu}_2\|$ in this case)

Therefore

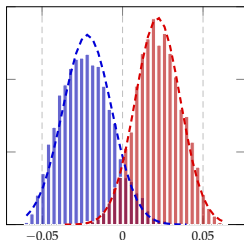
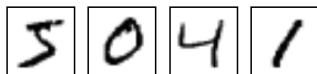
- ▶ entry-wise:

$$\mathbf{K}_{ij} = \exp(-1) \left(1 + \underbrace{\frac{1}{p} \mathbf{z}_i^\top \mathbf{z}_j}_{O(p^{-1/2})} \right) \pm \underbrace{\frac{1}{p} g(\|\boldsymbol{\mu}_1 - \boldsymbol{\mu}_2\|)}_{O(p^{-1})} + *, \text{ so that } \frac{1}{p} g(\|\boldsymbol{\mu}_1 - \boldsymbol{\mu}_2\|) \ll \frac{1}{p} \mathbf{z}_i^\top \mathbf{z}_j,$$

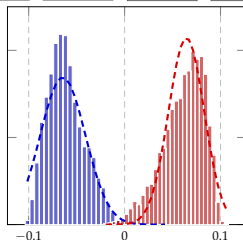
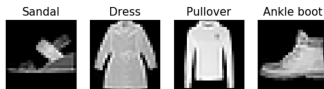
- ▶ spectrum-wise: (i) $\|\mathbf{K} - \exp(-1) \mathbf{1}_n \mathbf{1}_n^\top\| \not\rightarrow 0$; (ii) $\|\frac{1}{p} \mathbf{Z}^\top \mathbf{Z}\| = O(1)$ and $\|g(\|\boldsymbol{\mu}_1 - \boldsymbol{\mu}_2\|) \frac{1}{p} \mathbf{j} \mathbf{j}^\top\| = O(1)$!
- ▶ **Same** phenomenon as the sample covariance example: $[\hat{\mathbf{C}} - \mathbf{C}]_{ij} \rightarrow 0 \not\Rightarrow \|\hat{\mathbf{C}} - \mathbf{C}\| \rightarrow 0!$

⇒ With **modern RMT**, we **understand** kernel spectral clustering (**eigenvectors!**) for large dimensional data!

Numerical results on kernel-based least squares SVM (LS-SVM)



(a) MNIST



(b) Fashion-MNIST

Figure: Empirical histogram of LS-SVM soft output versus RMT prediction, $n = 2048$, $p = 784$, $\gamma = 1$ with Gaussian kernel, for MNIST (left, 7 versus 9) and Fashion-MNIST (right, 8 versus 9) data. Results averaged over 30 runs.

Reminder on random features and neural networks

- ▶ kernel matrices $\mathbf{K} \in \mathbb{R}^{n \times n}$ from **pairwise** comparison of n data points: expensive for n large
- ▶ **idea**: find easy-to-compute $\hat{\mathbf{K}}$ to approximate \mathbf{K} , e.g., $\|\hat{\mathbf{K}} - \mathbf{K}\|$ is small
- ▶ **example**: random Fourier feature [RR08] $\Sigma^T = [\cos(\mathbf{W}\mathbf{X})^T, \sin(\mathbf{W}\mathbf{X})^T] \in \mathbb{R}^{2N \times n}$ of data $\mathbf{X} = [\mathbf{x}_1, \dots, \mathbf{x}_n] \in \mathbb{R}^{p \times n}$ with standard Gaussian $\mathbf{W} \in \mathbb{R}^{N \times p}$, i.e., $\mathbf{W}_{ij} \sim \mathcal{N}(0, 1)$
- ▶ approximates Gaussian kernel $\exp(-\|\mathbf{x}_i - \mathbf{x}_j\|^2/2)$: **entry-wise** convergence of RFF Gram $\frac{1}{N}[\Sigma^T \Sigma]_{ij} \rightarrow [\mathbf{K}_{\text{Gauss}}]_{ij}$ Gaussian kernel matrix as number of features $N \rightarrow \infty$
- ▶ **proof**: (strong) law of large numbers:

$$\begin{aligned} \frac{1}{N}[\Sigma^T \Sigma]_{ij} &= \frac{1}{N} \sum_{k=1}^N \cos(\mathbf{x}_i^T \mathbf{w}_k) \cos(\mathbf{x}_j^T \mathbf{w}_k) + \sin(\mathbf{x}_i^T \mathbf{w}_k) \sin(\mathbf{x}_j^T \mathbf{w}_k) \\ &\rightarrow \mathbb{E}_{\mathbf{w} \sim \mathcal{N}(\mathbf{0}, \mathbf{I}_p)} [\cos(\mathbf{x}_i^T \mathbf{w}) \cos(\mathbf{x}_j^T \mathbf{w}) + \sin(\mathbf{x}_i^T \mathbf{w}) \sin(\mathbf{x}_j^T \mathbf{w})] = [\mathbf{K}_{\cos} + \mathbf{K}_{\sin}]_{ij} = [\mathbf{K}_{\text{Gauss}}]_{ij} \end{aligned}$$

for $\mathbf{K}_{\cos} = e^{-\frac{1}{2}(\|\mathbf{x}_i\|^2 + \|\mathbf{x}_j\|^2)} \cosh(\mathbf{x}_i^T \mathbf{x}_j)$ and $\mathbf{K}_{\sin} = e^{-\frac{1}{2}(\|\mathbf{x}_i\|^2 + \|\mathbf{x}_j\|^2)} \sinh(\mathbf{x}_i^T \mathbf{x}_j)$.

³Ali Rahimi and Benjamin Recht. "Random Features for Large-Scale Kernel Machines". In: *Advances in Neural Information Processing Systems*. Vol. 20. NIPS'08. Curran Associates, Inc., 2008, pp. 1177–1184

Random features-based ridge regression and neural networks

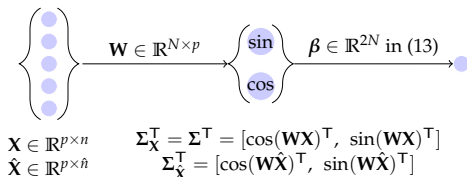


Figure: Illustration of random Fourier features regression model.

- ▶ RFF ridge regressor $\beta \in \mathbb{R}^{2N}$ given by, for regularization penalty $\gamma \geq 0$,

$$\beta \equiv \frac{1}{n} \Sigma \left(\frac{1}{n} \Sigma^T \Sigma + \gamma \mathbf{I}_n \right)^{-1} \mathbf{y} \cdot \mathbf{1}_{2N > n} + \left(\frac{1}{n} \Sigma \Sigma^T + \gamma \mathbf{I}_{2N} \right)^{-1} \frac{1}{n} \Sigma \mathbf{y} \cdot \mathbf{1}_{2N < n}. \quad (13)$$

- ▶ **Performance:** training and test Mean Squared Error (MSE): $E_{\text{train}} = \frac{1}{n} \|\mathbf{y} - \Sigma_{\mathbf{X}}^T \beta\|^2$ and $E_{\text{test}} = \frac{1}{\hat{n}} \|\hat{\mathbf{y}} - \Sigma_{\hat{\mathbf{X}}}^T \beta\|^2$, with $\Sigma_{\hat{\mathbf{X}}}^T \in \mathbb{R}^{\hat{n} \times 2N}$ RFFs of a test set $(\hat{\mathbf{X}}, \hat{\mathbf{y}})$ of size \hat{n} .
- ▶ **single-hidden-layer neural network with cos + sin activations**, connected to neural tangent kernel (NTK)

³ Arthur Jacot, Franck Gabriel, and Cl  ment Hongler. "Neural Tangent Kernel: Convergence and Generalization in Neural Networks". In: *Advances in Neural Information Processing Systems*. Vol. 31. NIPS'18. Curran Associates, Inc., 2018, pp. 8571–8580

Random Fourier features approximate Gaussian kernel, but in which sense?

- ▶ [RR08]: **entry-wise** convergence of RFF Gram $\frac{1}{N}[\Sigma^T \Sigma]_{ij} \rightarrow [\mathbf{K}_{\text{Gauss}}]_{ij}$ Gaussian kernel matrix as $N \rightarrow \infty$
- ▶ again, **not true** in **spectral norm** sense, i.e., $\|\Sigma^T \Sigma / N - \mathbf{K}_{\text{Gauss}}\| \not\rightarrow 0$ unless $N \gg n$
 - e.g., $\Sigma^T \Sigma \in \mathbb{R}^{n \times n}$ of rank **at most** N if $N \leq n$, while $\mathbf{K}_{\text{Gauss}}$ of rank n (for distinct \mathbf{x}_i)
 - **significant impact** on various RFF-based algorithms

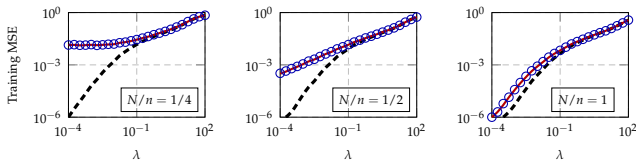


Figure: Training MSEs of RFF ridge regression on MNIST data (class 3 versus 7) as a function of regression penalty λ .

- ▶ **effective kernel** can be derived with RMT in the large n, p, N regime
- ▶ provides precise **training** and **test** performances of RFF for **any** ratio N/n , more **practical** and more **flexible**, recover Gaussian kernel result with $N/n \rightarrow \infty$
- ▶ **data-dependent** theory with **no** strong assumption on data

Sharp analysis of RFF ridge regression performance via RMT

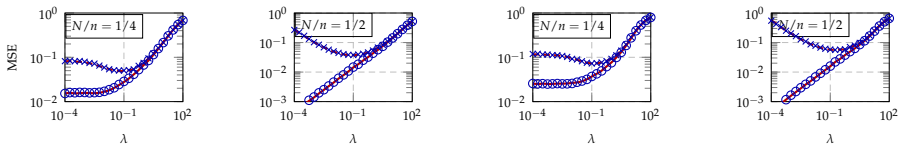


Figure: MSEs of RFF ridge regression on Fashion- (left two) and Kannada-MNIST (right two).

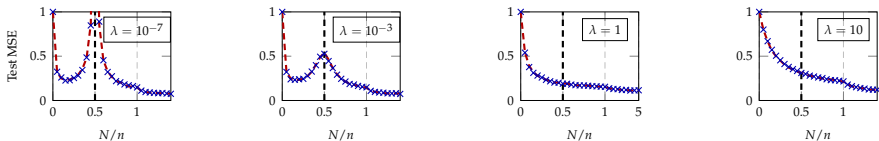
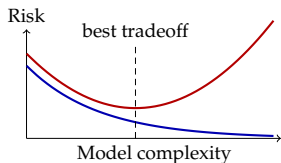
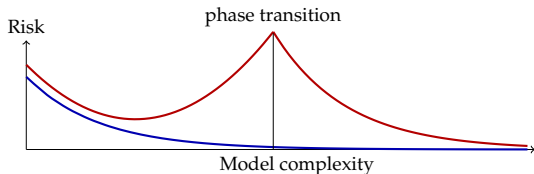


Figure: Test MSEs of RFF regression as a function of the ratio N/n , on MNIST data set.

“Recap” for double descent phenomenon for over-parameterized models



(a) Classical U-shaped risk



(b) Modern “double descent” risk

Figure: Comparison between training risk (blue) and true/test risk (red).

- ▶ empirically observed for various large-scale machine learning models, e.g., RF-based methods, decision trees, ensemble methods, and deep NNs
- ▶ **proved** here for RFF on **real-world** data!
- ▶ **phase transition** from under- to over-param of resolvent $(\Sigma^T \Sigma + \lambda \mathbf{I}_n)^{-1}$ in the ridgeless $\lambda \rightarrow 0$ limit

⁴Mikhail Belkin et al. “Reconciling modern machine-learning practice and the classical bias–variance trade-off”. In: *Proceedings of the National Academy of Sciences* 116.32 (2019), pp. 15849–15854

⁵Trevor Hastie et al. “Surprises in High-Dimensional Ridgeless Least Squares Interpolation”. In: *arXiv* (2019). eprint: 1903.08560

Take-away messages and references

Take-away messages:

- ▶ RF methods: classical statistical learning theory provides performance guarantee for $N \gg n, p$
- ▶ here we derive (limiting) kernel in the more **practical large n, p, N regime**
- ▶ fast tuning of regularization parameter λ
- ▶ double descent theory for novel understanding of **over-parameterized** neural networks

References:

- ▶ **Zhenyu Liao**, Romain Couillet, and Michael W Mahoney. “A random matrix analysis of random Fourier features: beyond the Gaussian kernel, a precise phase transition, and the corresponding double descent”. In: *Advances in Neural Information Processing Systems (NeurIPS)*. vol. 33. Curran Associates, Inc., 2020, pp. 13939–13950
- ▶ **Zhenyu Liao**, Romain Couillet, and Michael W Mahoney. “Sparse quantized spectral clustering”. In: *The Ninth International Conference on Learning Representations (ICLR 2021)*. 2021
- ▶ Michal Dereziński et al. “Sparse sketches with small inversion bias”. In: *Proceedings of Thirty Fourth Conference on Learning Theory (COLT)*. vol. 134. PMLR, 15–19 Aug 2021, pp. 1467–1510
- ▶ Cosme Louart, **Zhenyu Liao**, and Romain Couillet. “A random matrix approach to neural networks”. In: *Annals of Applied Probability* 28.2 (2018), pp. 1190–1248
- ▶ Song Mei and Andrea Montanari. “The Generalization Error of Random Features Regression: Precise Asymptotics and the Double Descent Curve”. In: *Communications on Pure and Applied Mathematics* (2021)
- ▶ **Zhenyu Liao** and Michael W. Mahoney. “Hessian Eigenspectra of More Realistic Nonlinear Models”. In: *Advances in Neural Information Processing Systems (NeurIPS)*. 2021

RMT for machine learning: from theory to practice!

Random matrix theory (RMT) for machine learning:

- ▶ **change of intuition** from small to large dimensional learning paradigm!
- ▶ **better understanding** of existing methods: why they work if they do, and what the issue is if they do not
- ▶ **improved novel methods** with performance guarantee!

Thank you! Q & A?

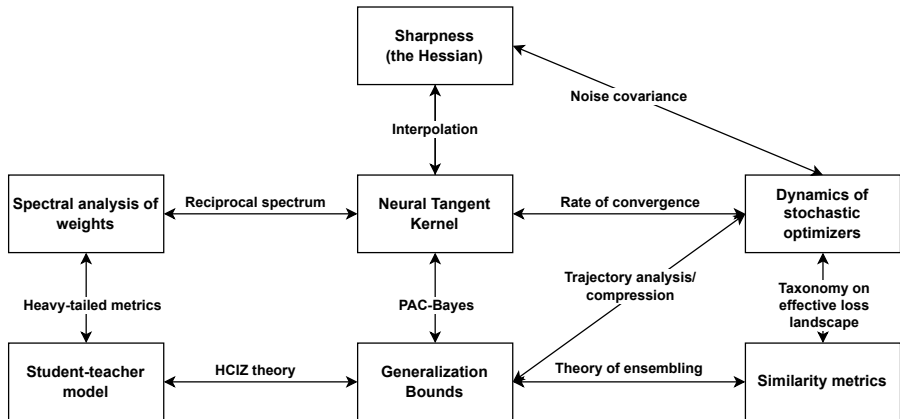
Outline

- 1 Weight Analysis and Heavy-Tailed Self-Regularization
- 2 Phenomenological Approach to Statistical Mechanics of Generalization
- 3 Using Heavy-Tailed Self-Regularization
- 4 Random Matrix Theory for Modern ML
- 5 Putting It All Together**
- 6 Conclusion

Why does deep learning work?

Why does deep learning work?

*Power law spectra are essential
to the story*



Outline

- 1 Weight Analysis and Heavy-Tailed Self-Regularization
- 2 Phenomenological Approach to Statistical Mechanics of Generalization
- 3 Using Heavy-Tailed Self-Regularization
- 4 Random Matrix Theory for Modern ML
- 5 Putting It All Together
- 6 Conclusion**

Conclusion

- Weight Analysis and Heavy-Tailed Self-Regularization
- Phenomenological Approach to Statistical Mechanics of Generalization
- Using Heavy-Tailed Self-Regularization
- Random Matrix Theory for Modern ML
- Putting It All Together