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

Outline

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

RBMs = Hopfield + temperature + backprop:

RBMs 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 langscapes, 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

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 $\gamma = P\left[|\epsilon_t - \epsilon| > \delta\right]$
  - 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 $P\left[\max_{h\in\mathcal{F}} |\epsilon_t(h) - \epsilon(h)| > \delta\right] \leq 2\,|\mathcal{F}|\, e^{-2m\delta^2}$
  - 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*: $m, N \to \infty$ 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

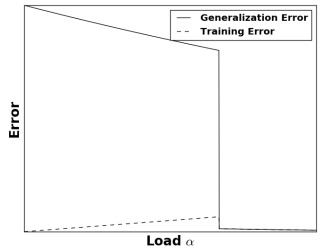
Rethinking generalization requires revisiting old ideas

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

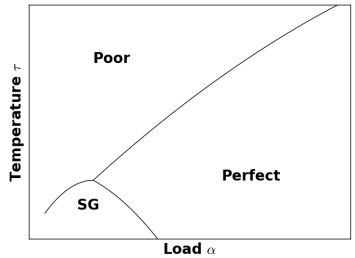
Very Simple Deep Learning (VSDL) model:

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

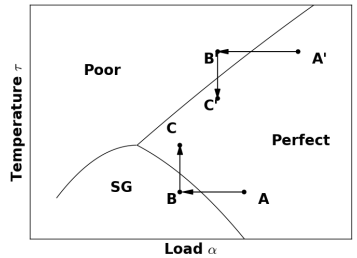
Nearly any non-trivial model[‡] exhibits "phase diagrams," with *qualitatively* different generalization properties, for different parameter values.



(a) Training/generalization error.



(b) Learning phases in τ-α plane.



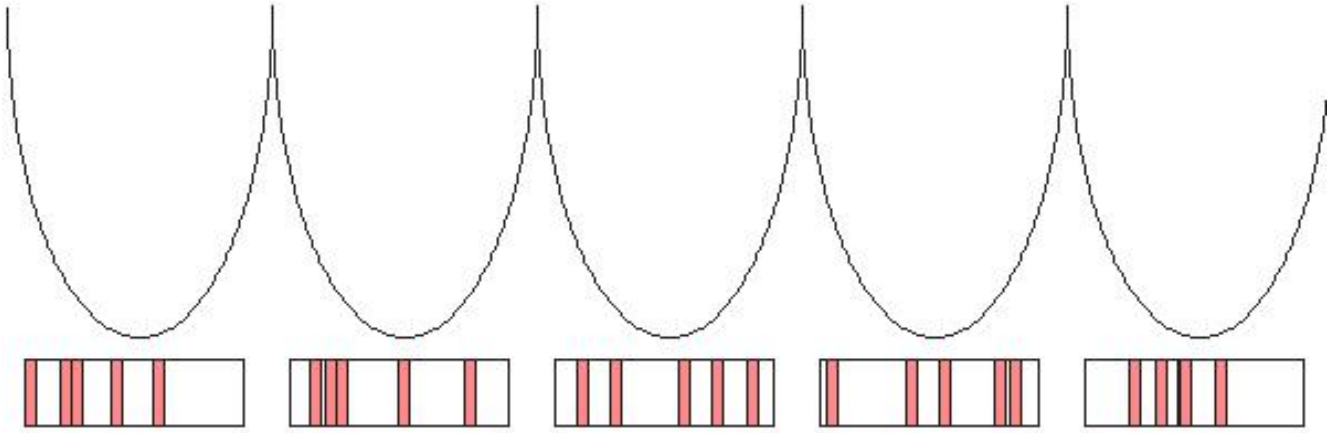
(c) Noisifying data and adjusting knobs.

[‡] *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

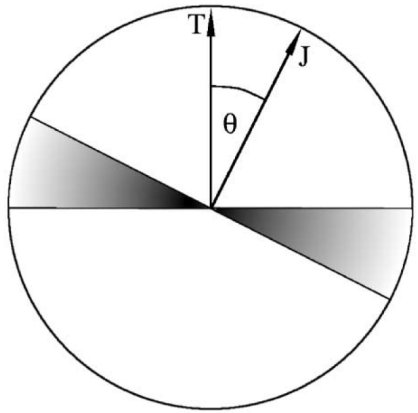
Stat Mech Setup: Student Teacher Model

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

Given N labeled data points

- Imagine a Teacher Network **T** that maps data to labels
- Learning finds a Student **J** similar to the Teacher **T**
- Consider all possible Student Networks **J** for all possible teachers **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: $\delta(Tr[\mathbf{J}^2] - N)$
- Teacher Overlap (Potential): $\delta(\frac{1}{N} Tr[\mathbf{J}^\dagger \mathbf{T}] - cos(\pi\epsilon))$
- Teacher Phase Space Volume (Density of States):

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

Comparison to traditional Statistical Mechanics:

- Phase Space Volume, free particles:

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

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

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

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

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 intersting)
- 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): $\|\mathbf{Y}^\dagger \mathbf{X}\|_F^2$

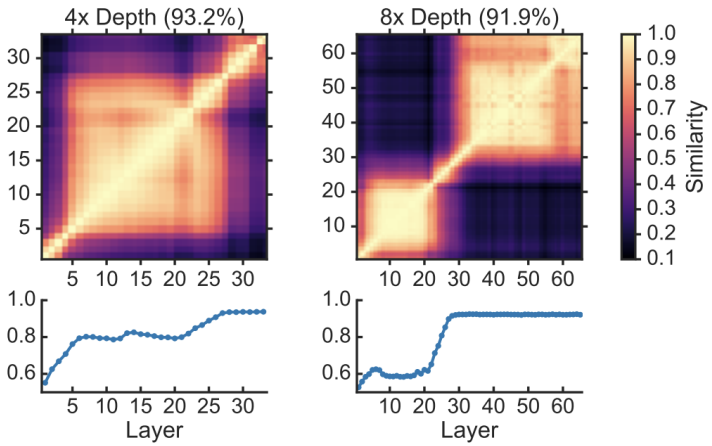


Figure: Diagnostic Tool for both individual and comparative DNNs

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)

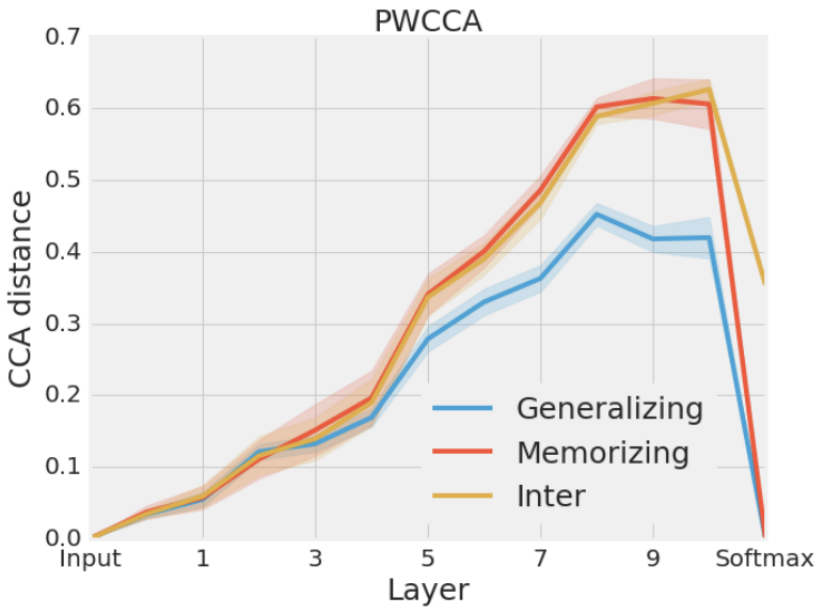


Figure: Generalizing networks converge to more similar solutions than memorizing networks.

Outline

Motivations: Theoretical AND Practical

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

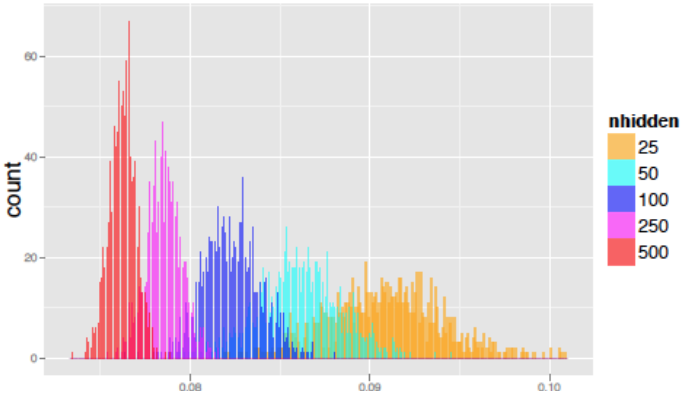
- convex versus non-convex optimization?
- explicit/implicit regularization?
- is / why is / when is deep better?
- VC theory versus Statistical Mechanics theory?
- ...

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



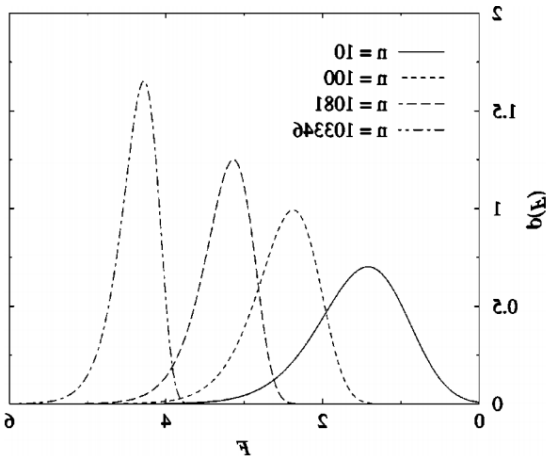
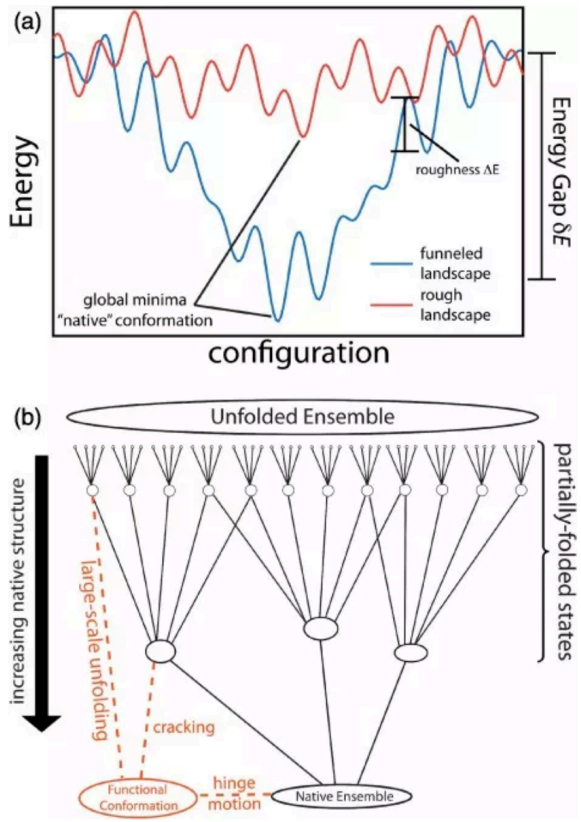


FIG. 1. Plot of the foldability distribution $p(\mathcal{E})$ for different numbers of compact states ($n = 10$, 100, 1081, 10 346), calculated using the random energy model.

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

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:

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

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

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

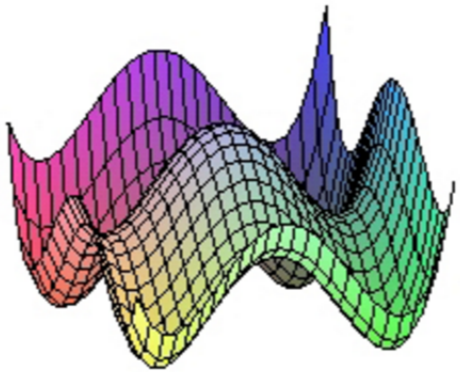
E_{DNN} is "the" *Energy Landscape*:

- 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

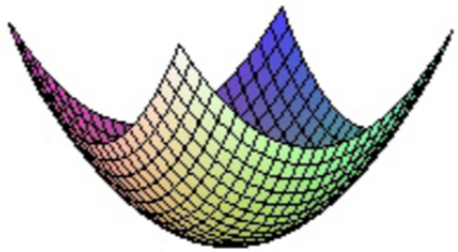
Problem: How can this possibly work?

Expected



Highly non-convex?

Observed

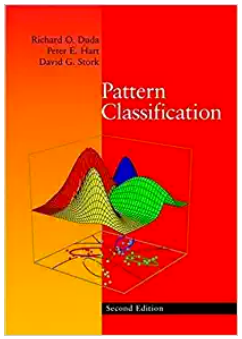


Apparently not!

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

Problem: Local Minima?

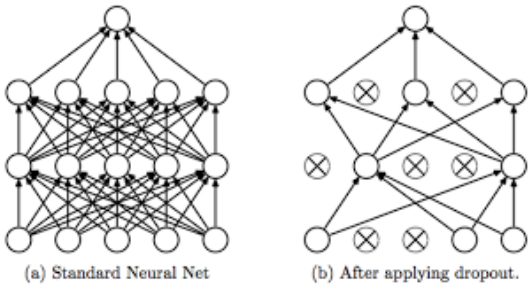
Duda, Hart and Stork, 2000



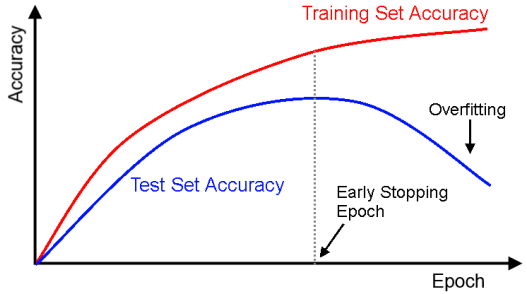
Whereas in low-dimensional spaces, local minima can be plentiful, in high dimension, the problem of local minima is different: The high-dimensional space may afford more ways (dimensions) for the system to "get around" a barrier or local maximum during learning. The more superfluous the weights, the 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

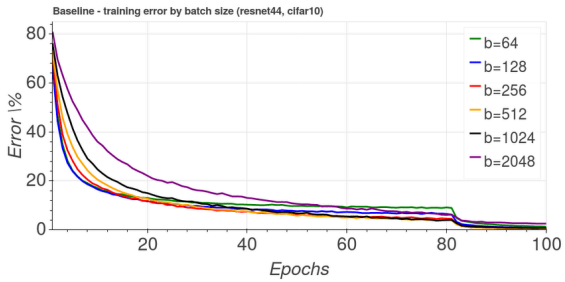
Motivations: what is regularization?



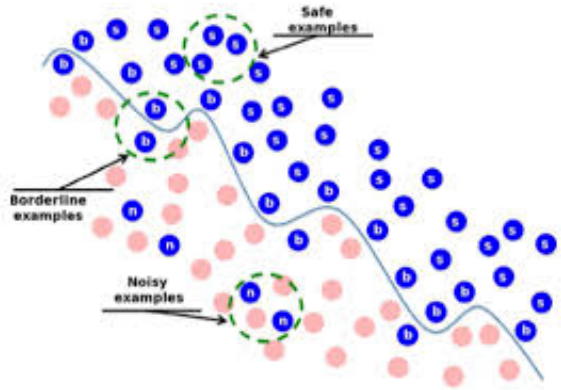
(a) Dropout.



(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

Outline

Basics of Regularization

Ridge Regression / Tikhonov-Phillips Regularization

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

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

$$\min_{\mathbf{x}} \|\hat{\mathbf{W}}\mathbf{x} - \mathbf{y}\|_2^2 + \alpha\|\hat{\mathbf{x}}\|_2^2 \qquad \textit{familiar optimization problem}$$

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 optimiation 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

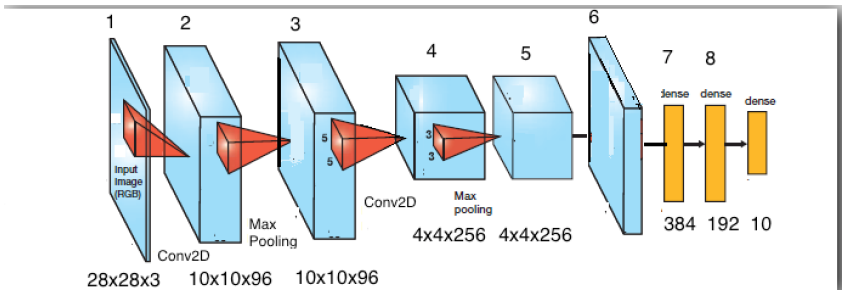
Question: What happens to the layer weight matrices $\mathbf{W}_L$?

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

Retrained LeNet5 on MINST using Keras.

Two other small models:

- 3-Layer MLP
- Mini AlexNet



Conv2D MaxPool Conv2D MaxPool FC1 FC2 FC

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

- AlexNet, Inception, etc.

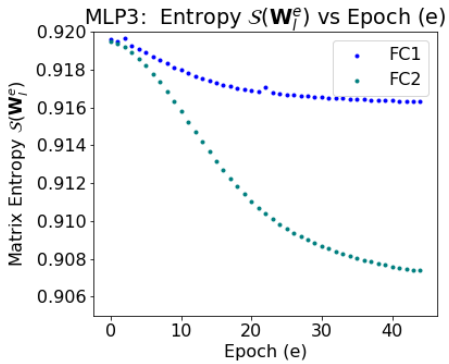
Outline

Matrix complexity: Matrix Entropy and Stable Rank

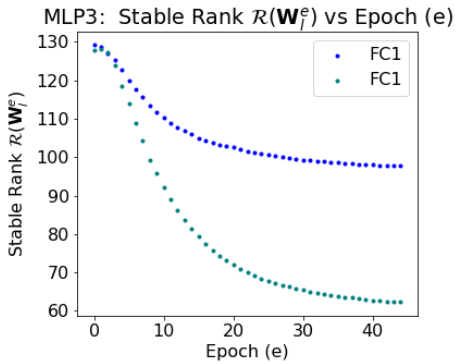
$$\mathbf{W} = \mathbf{U}\Sigma\mathbf{V}^T \qquad \nu_i = \Sigma_{ii} \qquad p_i = \nu_i^2 / \sum_i \nu_i^2$$

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

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



(e) MLP3 Entropies.



(f) MLP3 Stable Ranks.

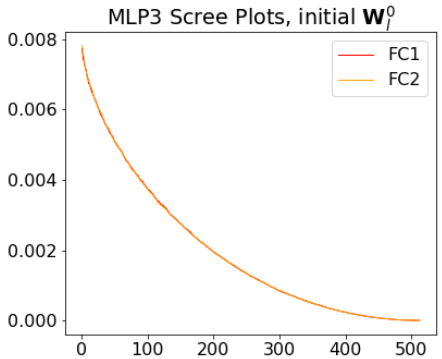
Figure: Matrix Entropy & Stable Rank show transition during Backprop training.

Matrix complexity: Scree Plots

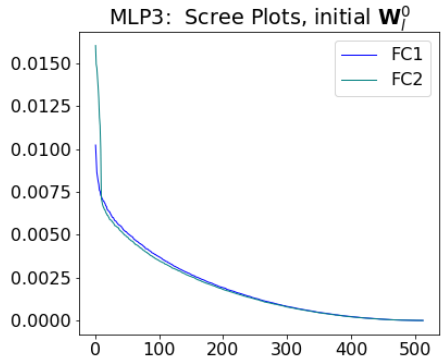
$$\mathbf{W} = \mathbf{U}\Sigma\mathbf{V}^T \qquad \nu_i = \Sigma_{ii} \qquad p_i = \nu_i^2 / \sum_i \nu_i^2$$

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

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



(a) Initial Scree Plot.



(b) Final Scree Plot.

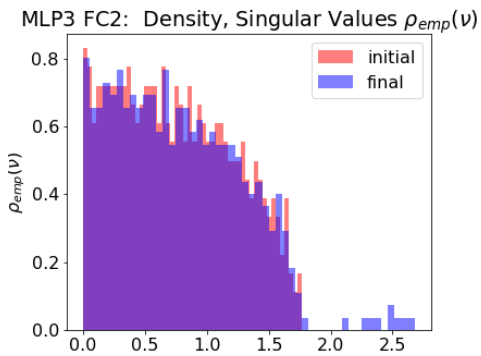
Figure: Scree plots for initial and final configurations for MLP3.

Matrix complexity: Singular/Eigen Value Densities

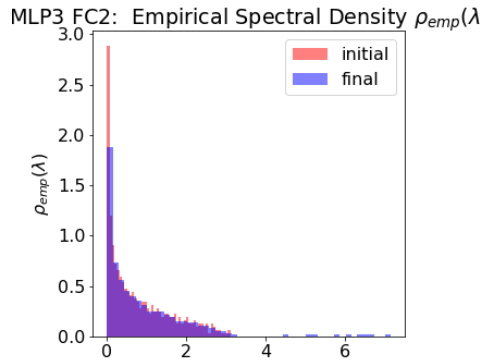
$$\mathbf{W} = \mathbf{U}\Sigma\mathbf{V}^T \qquad \nu_i = \Sigma_{ii} \qquad p_i = \nu_i^2 / \sum_i \nu_i^2$$

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

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

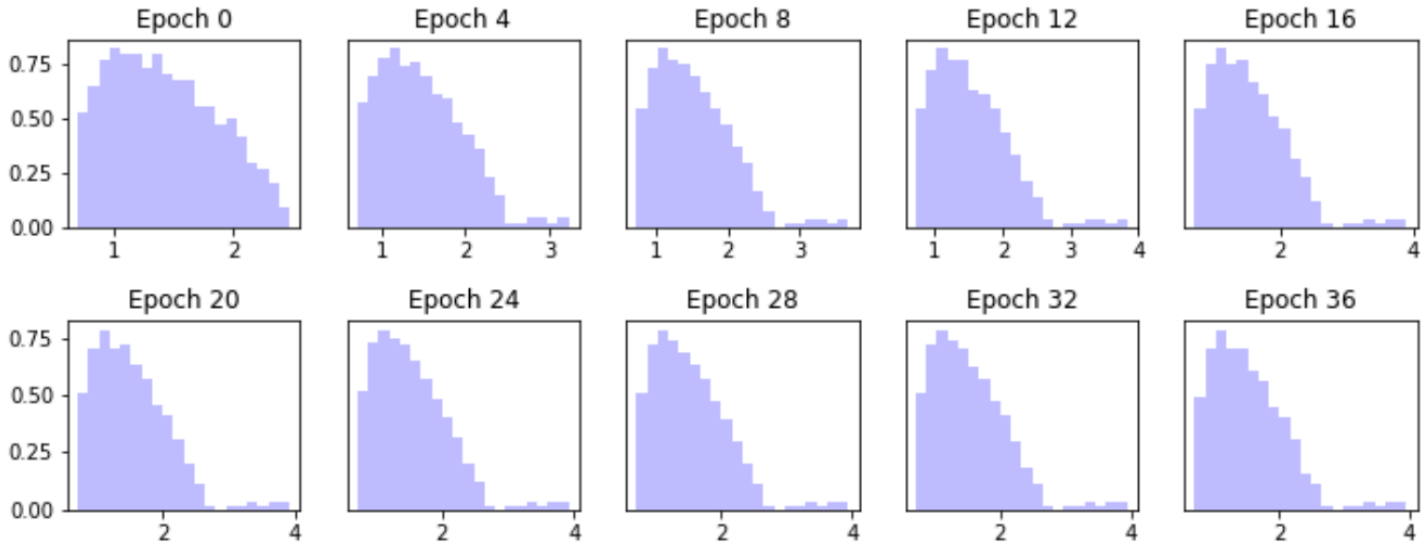
Empirical Spectral Density (ESD: eigenvalues of $X = \mathbf{W}_L^T \mathbf{W}_L$)

```
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 $X = \mathbf{W}_L^T \mathbf{W}_L$)

Eopch 0: Random Matrix



Eopch 36: Random + Spiles

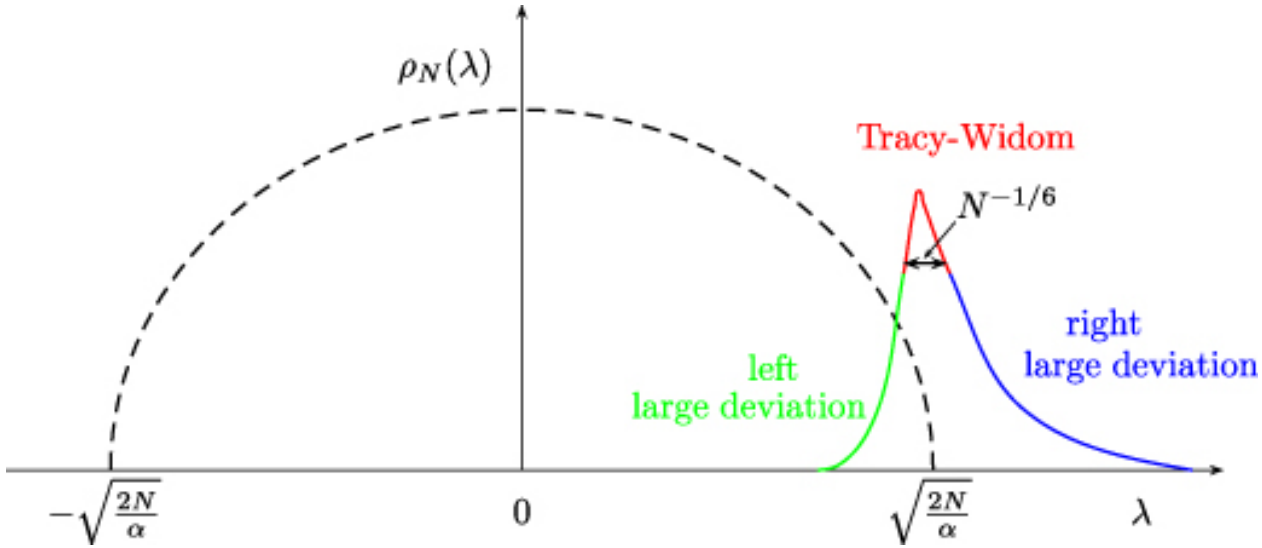
Entropy decrease corresponds to:

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

Outline

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 $W_{ij} \sim N(0, \sigma^2_{mp})$.

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

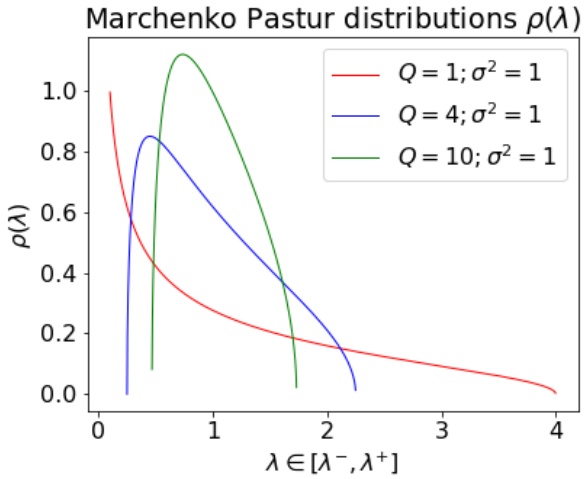
$$\rho_N(\lambda) := \frac{1}{N}\sum_{i=1}^{M}\delta(\lambda - \lambda_i)$$

$$\xrightarrow[Q \text{ fixed}]{N\to\infty} \begin{cases} \dfrac{Q}{2\pi\sigma^2_{mp}}\dfrac{\sqrt{(\lambda^+ - \lambda)(\lambda - \lambda^-)}}{\lambda} & \text{if } \lambda \in [\lambda^-, \lambda^+] \\ 0 & \text{otherwise.} \end{cases}$$

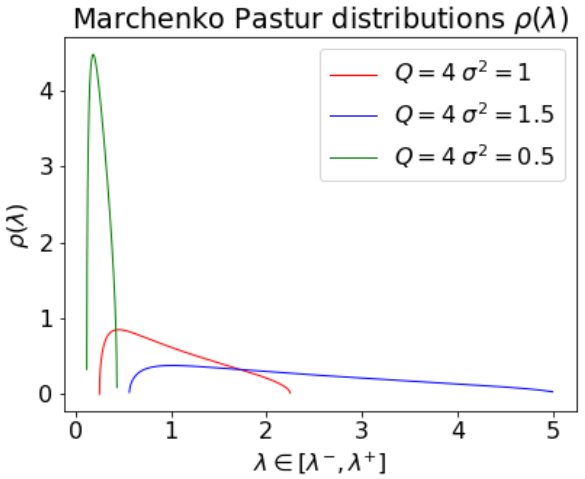
with well-defined edges (which depend on Q, the aspect ratio):

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

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 $\rho(\lambda),\ N \to \infty$	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 $N \to \infty$ behavior.

Fitting Heavy-tailed Distributions

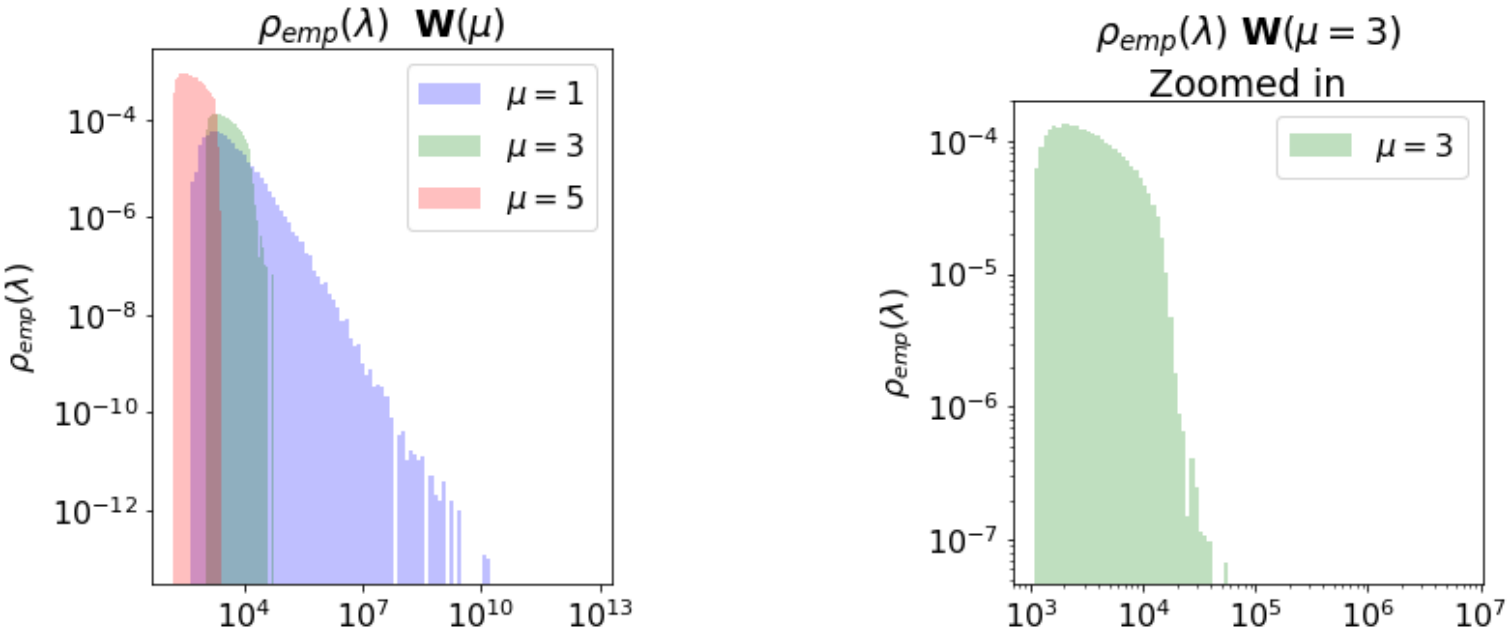
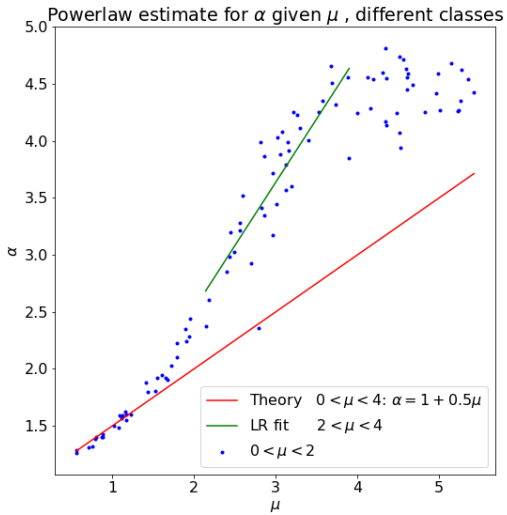
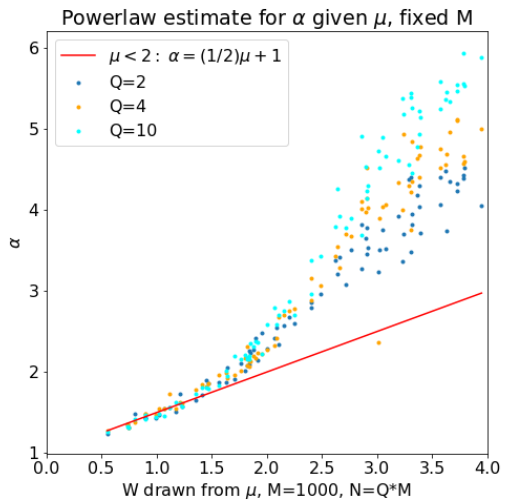


Figure: The log-log histogram plots of the ESD for three Heavy-Tailed random matrices $\mathbf{M}$ with same aspect ratio $Q = 3$, with $\mu = 1.0, 3.0, 5.0$, corresponding to the three Heavy-Tailed Universality classes ($0 < \mu < 2$ vs $2 < \mu < 4$ and $4 < \mu$).

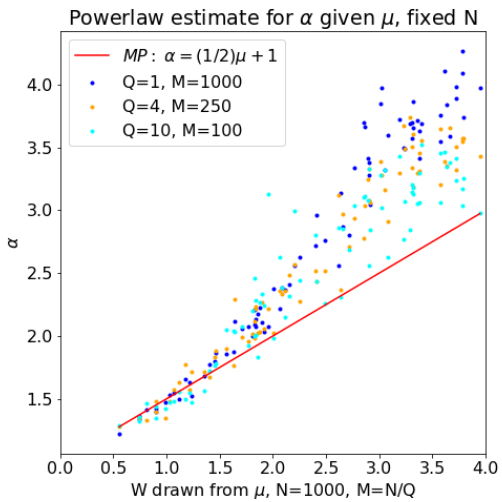
Non-negligible finite size effects



(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

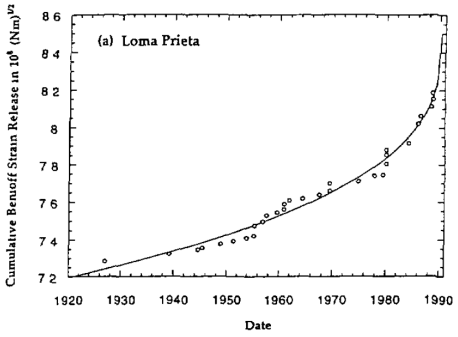
- Phenomenological work by physicists (Bouchard, Potters, Sornette, 90s)
- Theorem proving by mathematicians (Auffinger, Ben Arous, Burda, Peche, 00s)
- Universality of Power Laws, Levy-based dynamics, finite-size attractors, etc.

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_f	1990.3 ± 4.1	1998.8 ± 19.7

(b)

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

[‖] More sophisticated Renormalization Group (RG) analysys uses complex critical exponents, giving log-peripdic corrections.

Heavy-Tailed Universality: Market Crashes

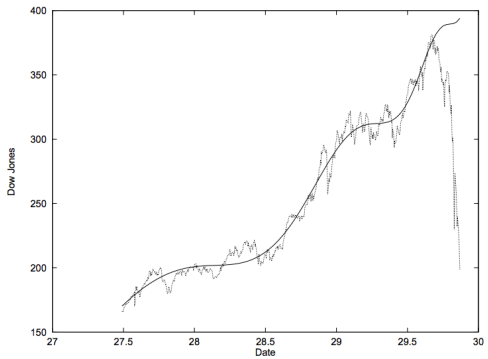
"Why Stock Markets Crash: Critical Events in Complex Financial Systems" by D. Sornette (book, 2003)

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

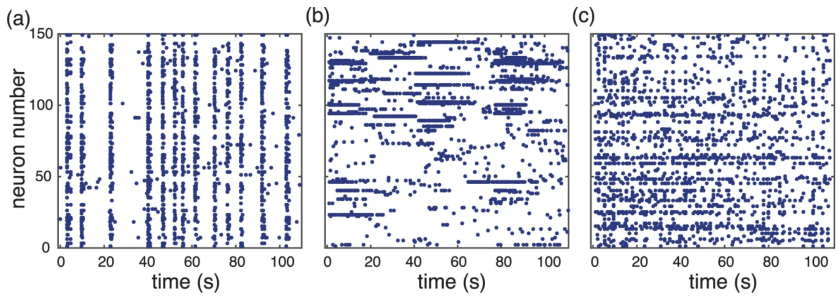
crash	t_c	t_{max}	t_{min}	drop	β	ω_1	λ	A	B	C	Var
1929 (WS)	30.22	29.65	29.87	47%	0.45	7.9	2.2	571	−267	14.3	56
1985 (DEM)	85.20	85.15	85.30	14%	0.28	6.0	2.8	3.88	1.16	−0.77	0.0028
1985 (CHF)	85.19	85.18	85.30	15%	0.36	5.2	3.4	3.10	−0.86	−0.055	0.0012
1987 (WS)	87.74	87.65	87.80	30%	0.33	7.4	2.3	411	−165	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

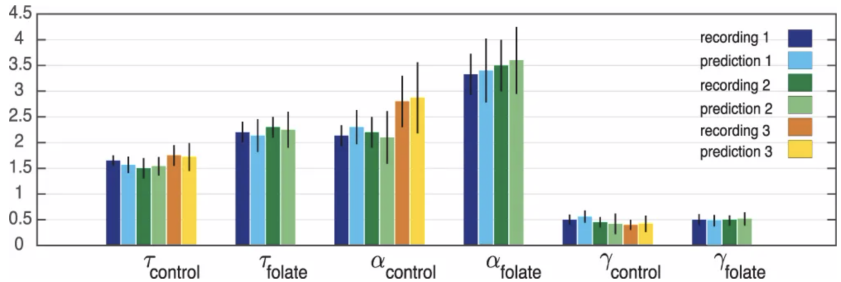
Heavy-Tailed Universality: Neuronal Avalanches

Neuronal avalanche dynamics indicates different universality classes in neuronal cultures; Scienfic 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 scalaing model

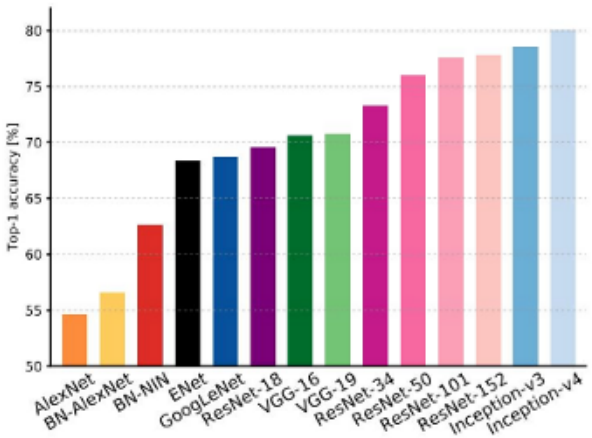
Outline

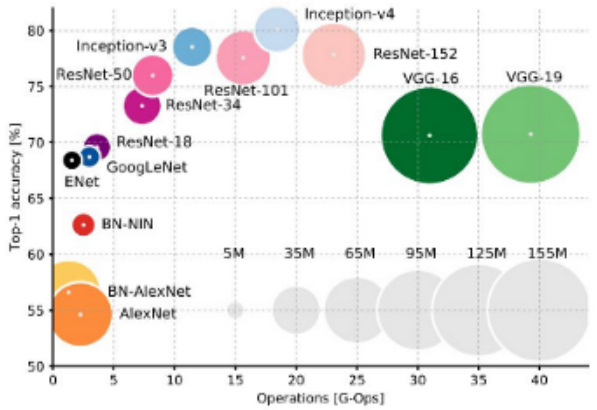
Outline

Experiments: just apply this to pre-trained models

https://medium.com/@siddharthdas_32104/cnns-architectures-lenet-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%	



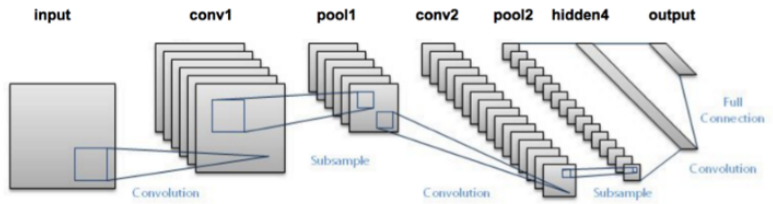


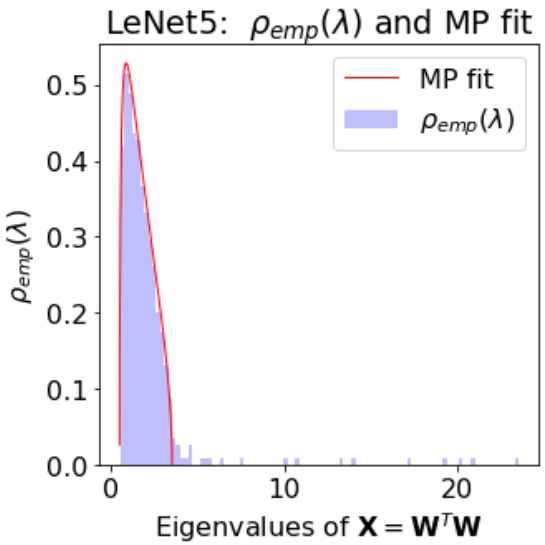
An Analysis of Deep Neural Network Models for Practical Applications, 2017.

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)





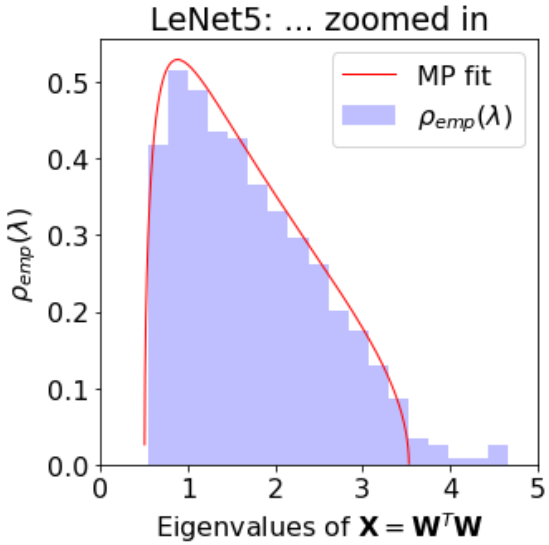
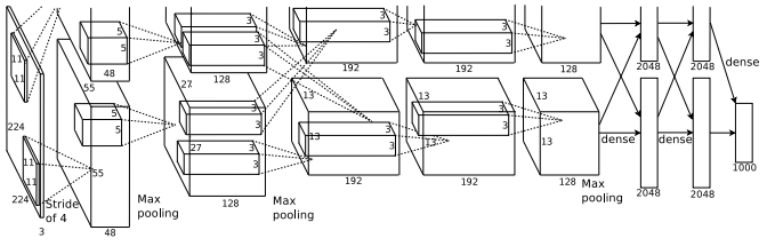
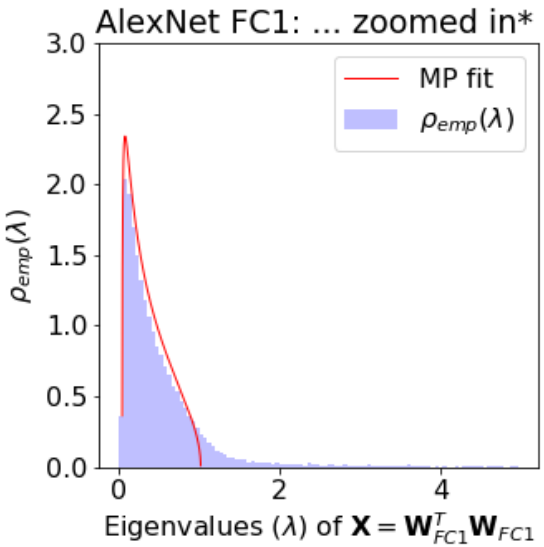


Figure: Full and zoomed-in ESD for LeNet5, Layer FC1.

Marchenko-Pastur Bulk + Spikes

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





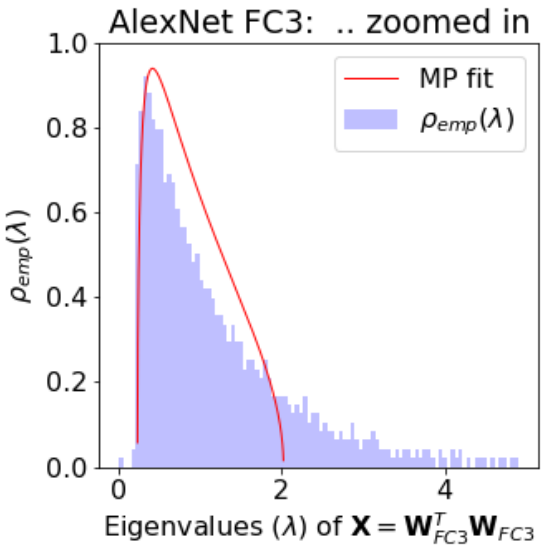
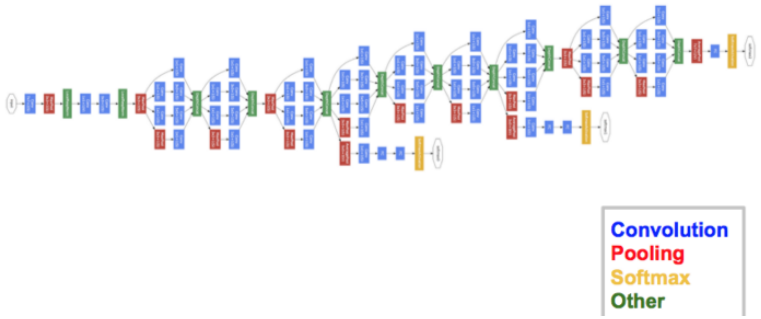
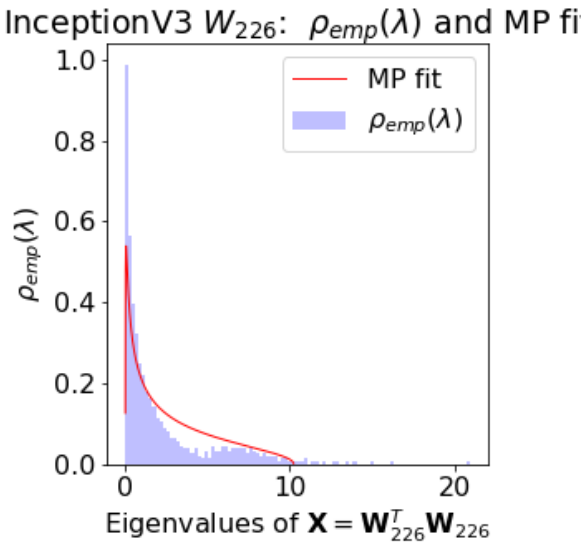


Figure: Zoomed-in ESD for Layer FC1 and FC3 of AlexNet.

Marchenko-Pastur Bulk-decay + Heavy-tailed

RMT: InceptionV3 (a particularly unusual example)





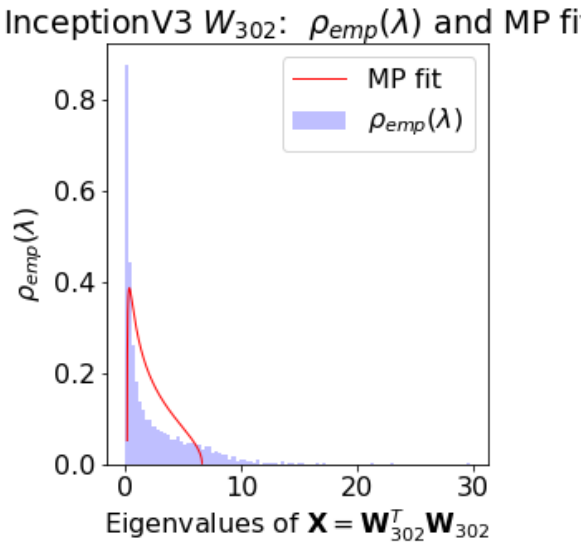
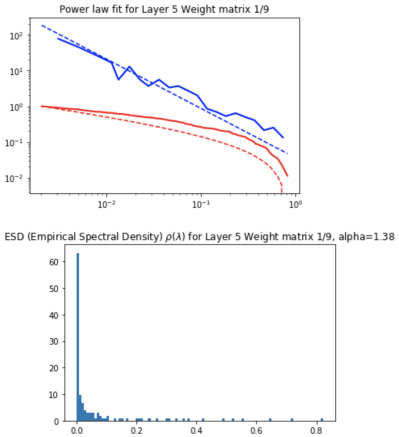


Figure: ESD for Layers L226 and L302 in InceptionV3, as distributed w/ pyTorch.

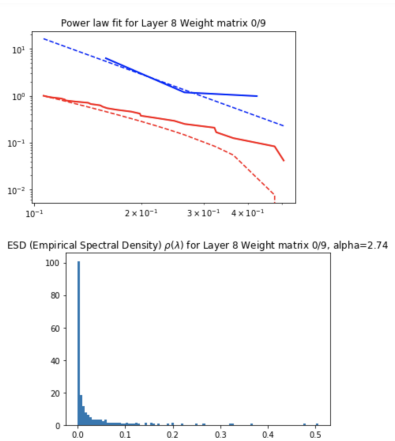
Marchenko-Pastur bulk decay, onset of Heavy Tails

Convolutional 2D Feature Maps

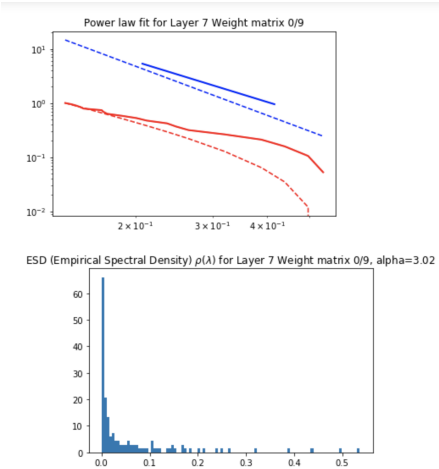
We analyze Conv2D layers by extracting the feature maps individually, i.e., A $(3 \times 3 \times 64 \times 64)$ Conv2D layer yields 9 (64×64) Feature Maps



(a) $\alpha = 1.38$



(b) $\alpha = 2.74$

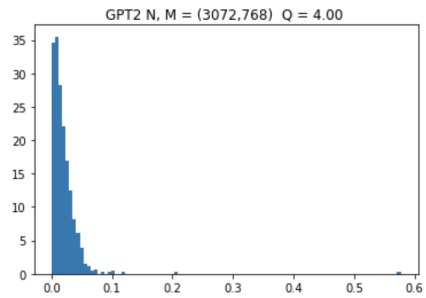


(c) $\alpha = 3.02$

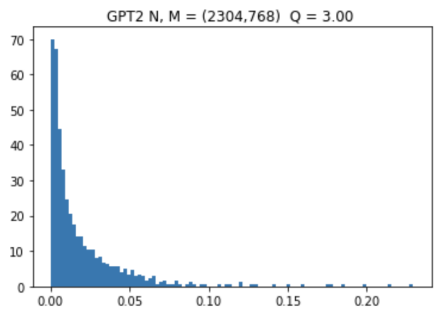
Figure: Select Feature Maps from different Conv2D layers of VGG16. Fits shown with PDF (blue) and CDF (red)

Open AI GPT2 Attention Matrices

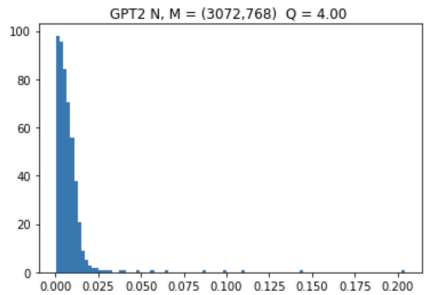
NLP Embedding and Attention Matrices are Dense/Linear, but generally have large aspect ratios



(a) $\alpha =$



(b) $\alpha =$

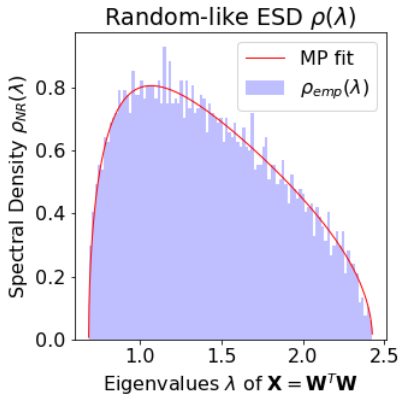


(c) $\alpha =$

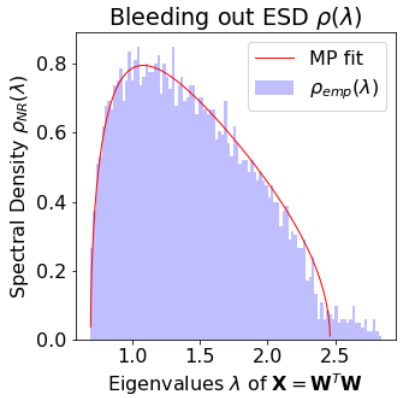
Figure: Selected ESDs from Open AI GPT2 (Huggingface implementation)

Outline

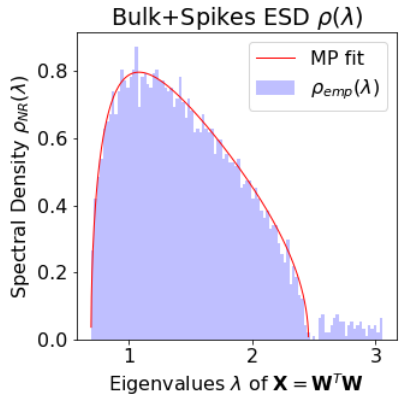
RMT-based 5+1 Phases of Training



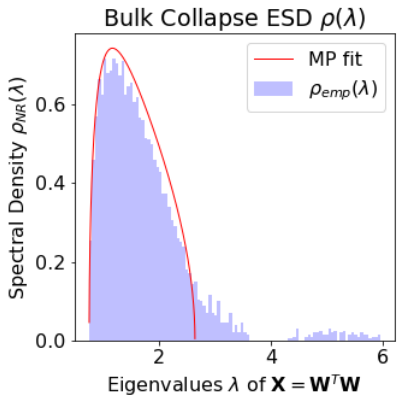
(a) RANDOM-LIKE.



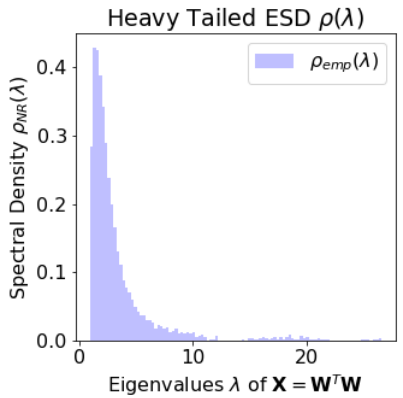
(b) BLEEDING-OUT.



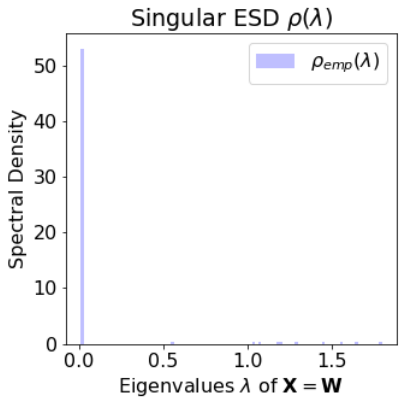
(c) BULK+SPIKES.



(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:

$$\mathbf{W} \simeq \mathbf{W}^{rand} + \Delta^{sig}. \tag{1}$$

	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

Bulk+Spikes: Small Models $\sim$ Tikhonov regularization

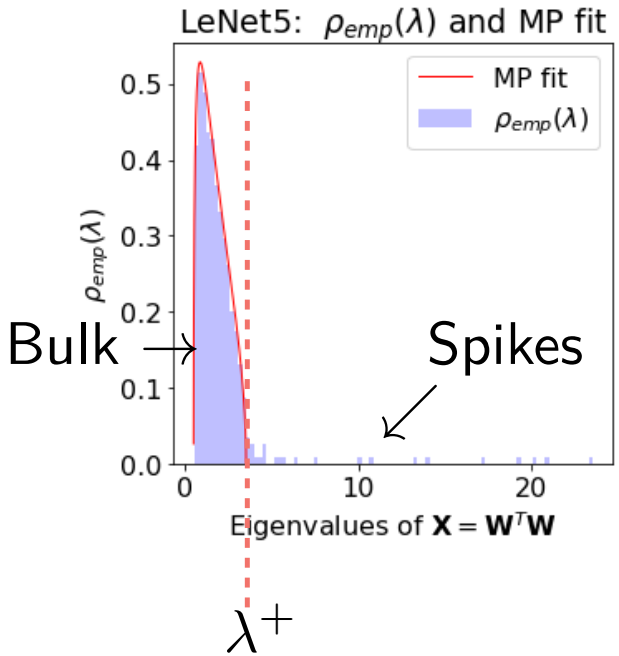
Low-rank perturbation

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

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

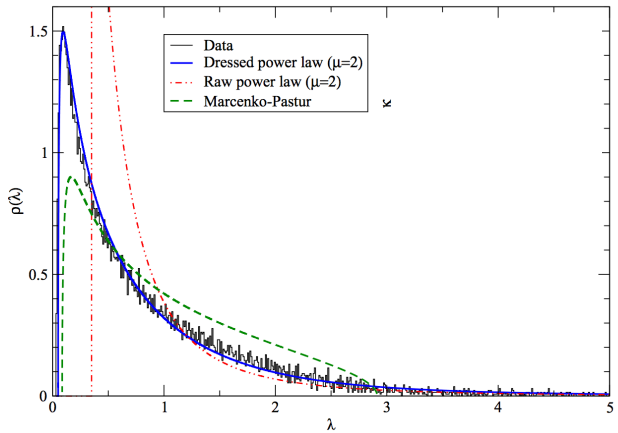
Heavy-tailed Self-regularization

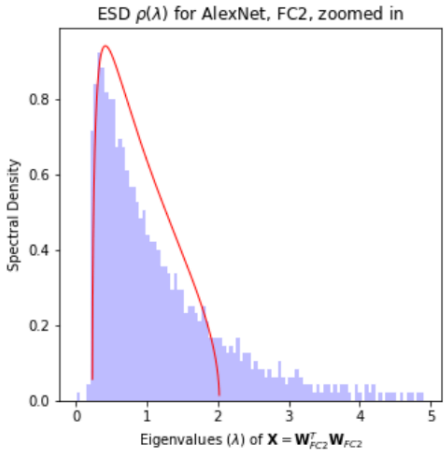
W is *strongly-correlated* and highly non-random

- 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

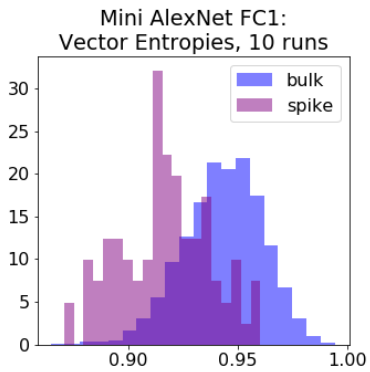
Summary of what we "suspect" today

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

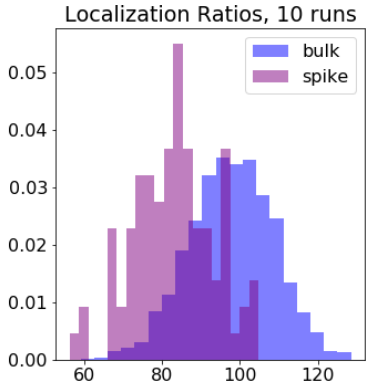
"All" larger, modern DNNs exhibit novel Heavy-tailed self-regularization

Spikes: carry more "information" than the Bulk

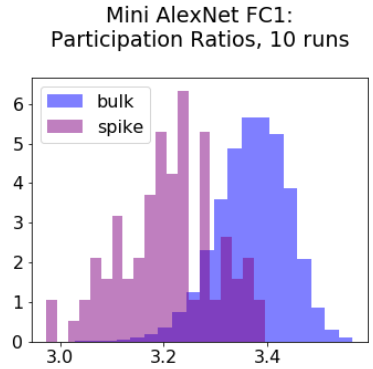
Spikes have less entropy, are more localized than bulk.



(a) Vector Entropies.



(b) Localization Ratios.

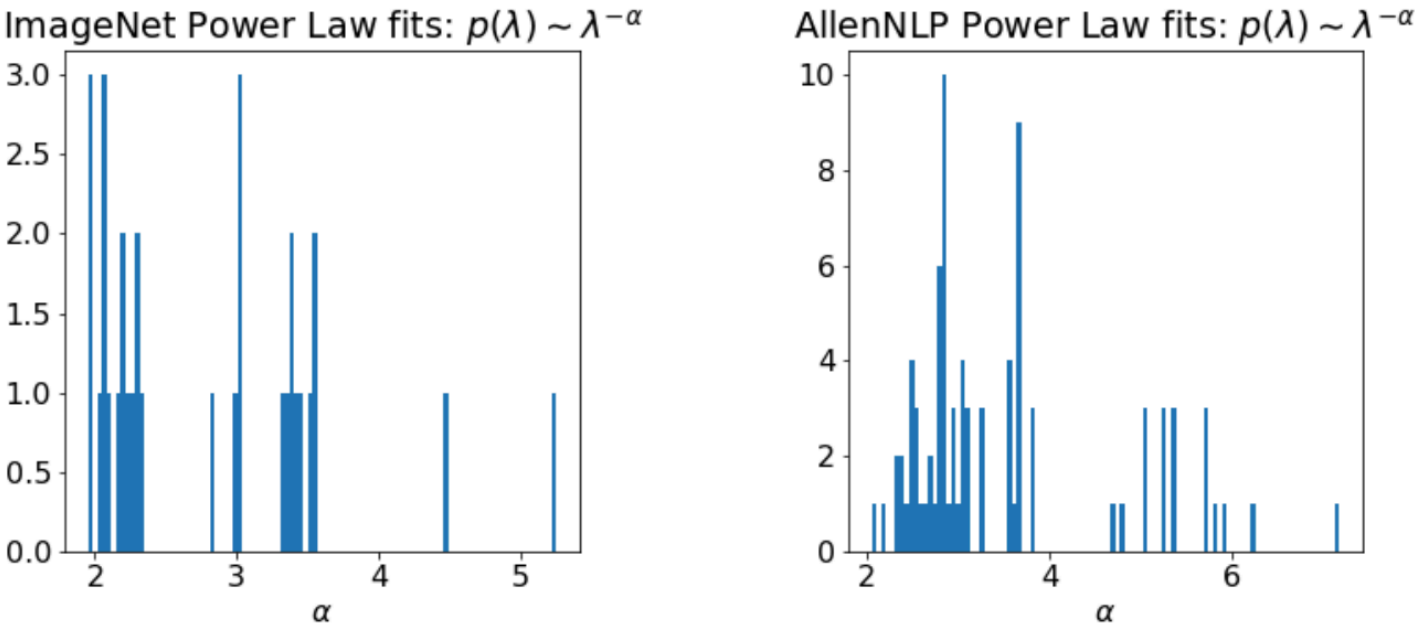


(c) Participation Ratios.

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

Information begins to concentrate in the spikes.

Power Law Universality: ImageNet and AllenNLP



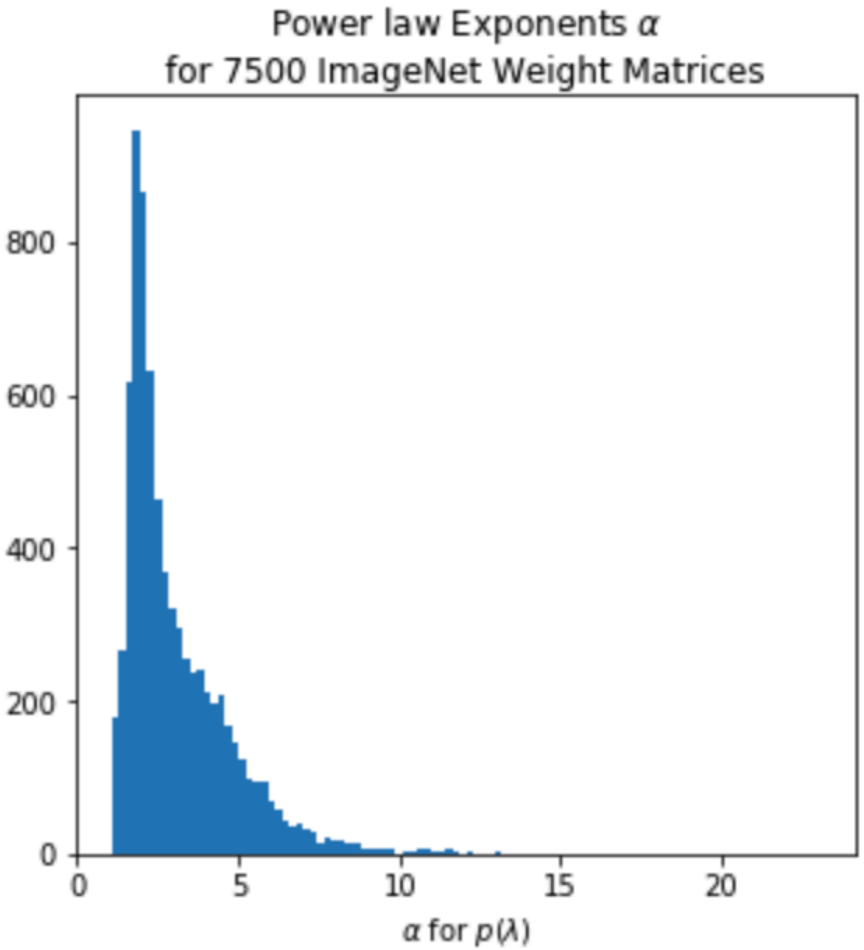
(a) ImageNet pyTorch models

(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

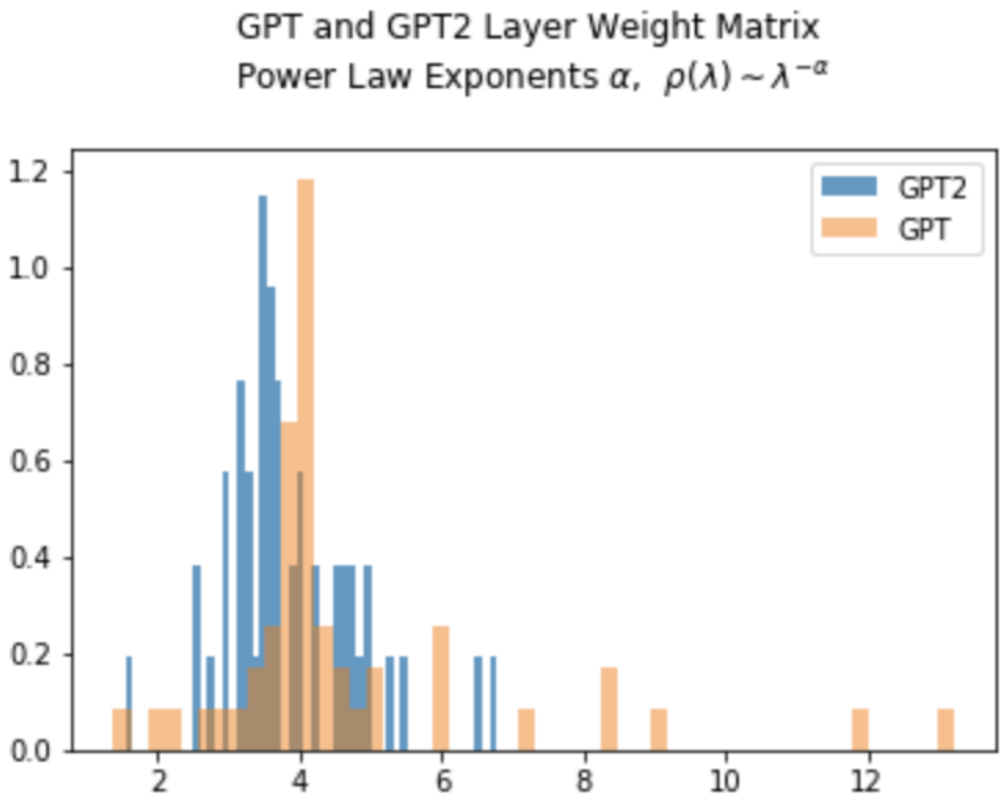
Power Law Universality: ImageNet



- 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 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

Outline

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 => decrease generalization accuracy
- Large batch sizes => 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

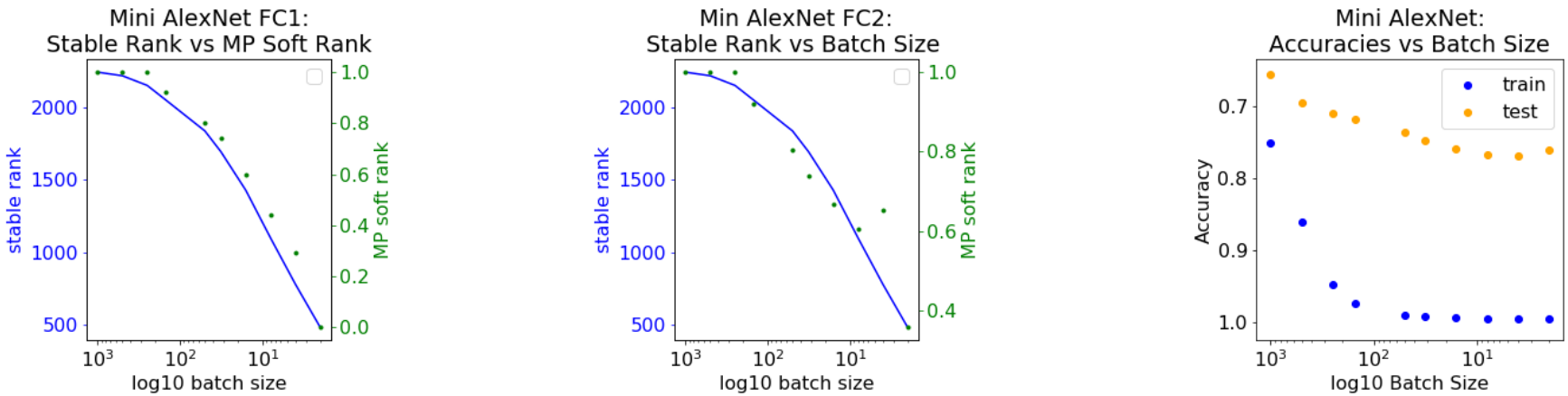
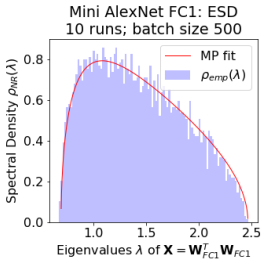


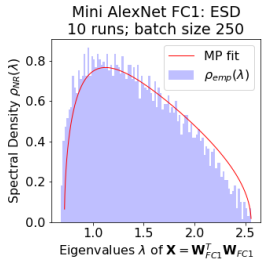
Figure: Varying Batch Size: Stable Rank and MP Softrank for FC1 and FC2 Training and Test Accuracies versus Batch Size for MiniAlexNet.

- *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* $\mathbf{W}$.

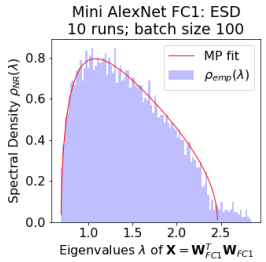
Batch Size Tuning: Generalization Gap



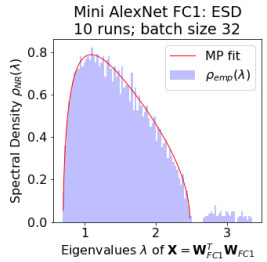
(a) Batch Size 500.



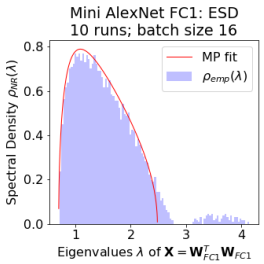
(b) Batch Size 250.



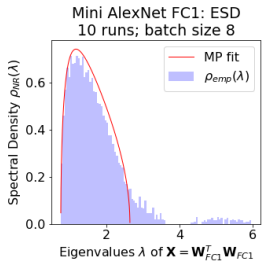
(c) Batch Size 100.



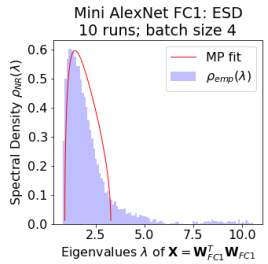
(d) Batch Size 32.



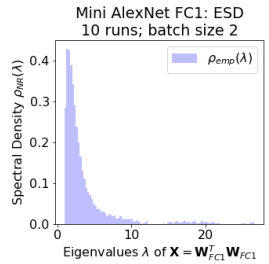
(e) Batch Size 16.



(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.*

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?

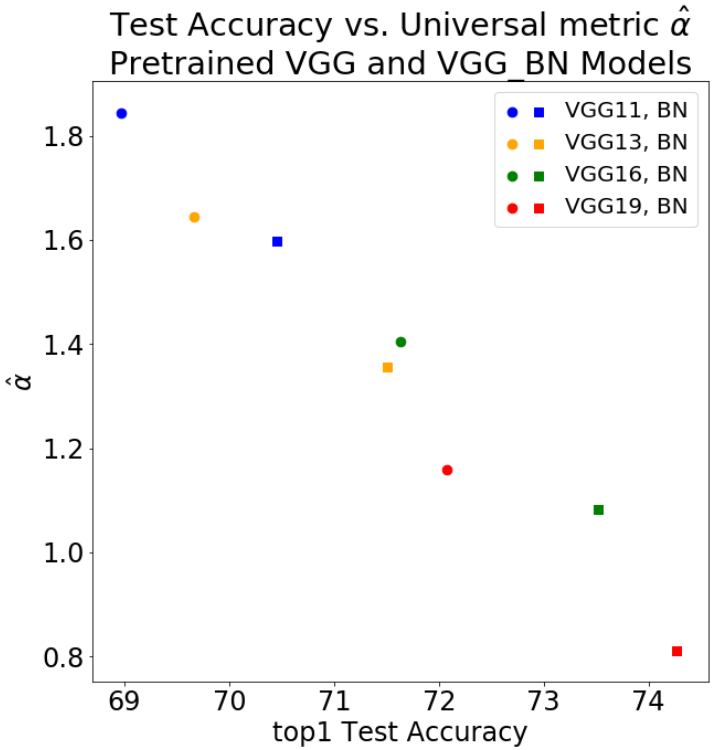
Why does deep learning work?

Outline

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, pretained 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 to analyze DNNs with our heavy tailed α and weighted $\hat{\alpha}$ metric (and other userful 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

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.

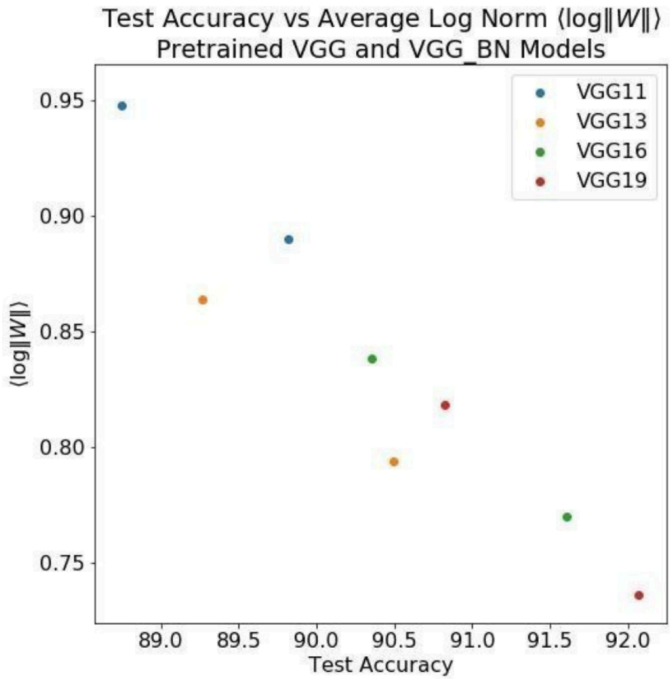
But how does it perform on state-of-the-art production-quality models?

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

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

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

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



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:

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

- Examine norm-powerlaw relations:

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

- Argue[††] that:

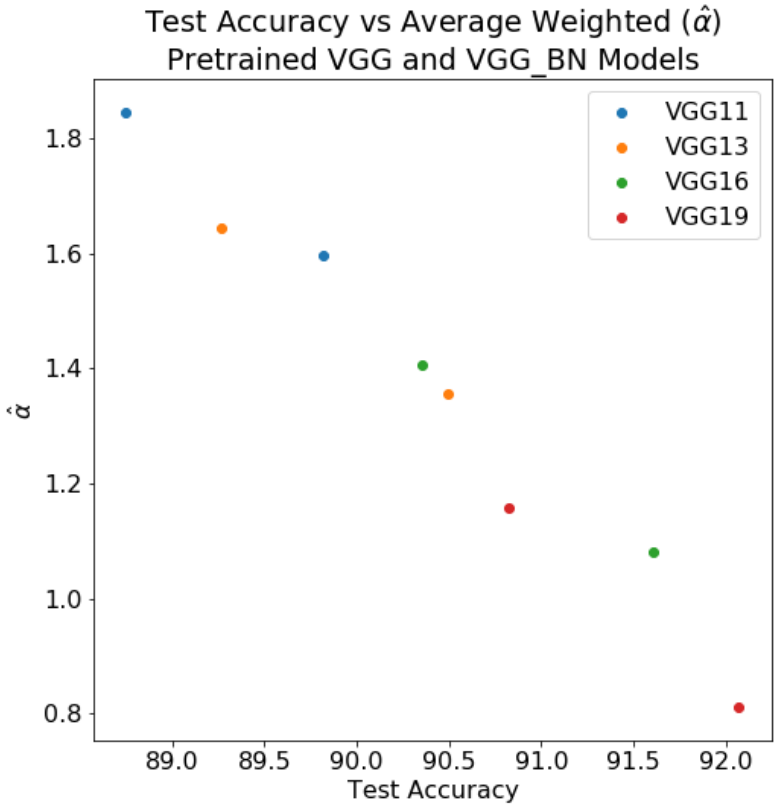
PL–Norm Relation: $\alpha \log \lambda^{max} \approx \log \|\mathbf{W}\|_F^2$.

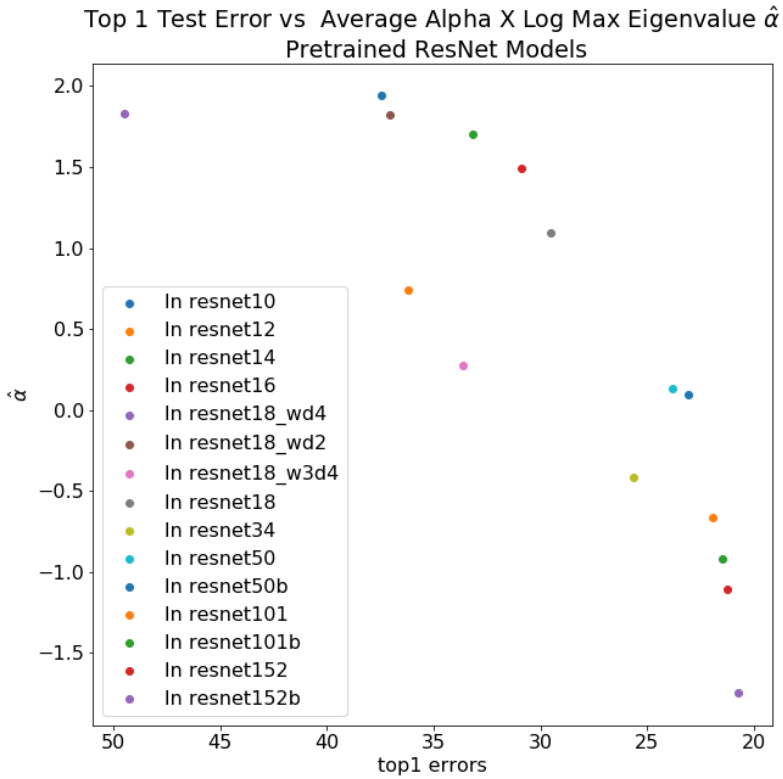
[††] Open problem: make the "heuristic" argument more "rigorous."

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.





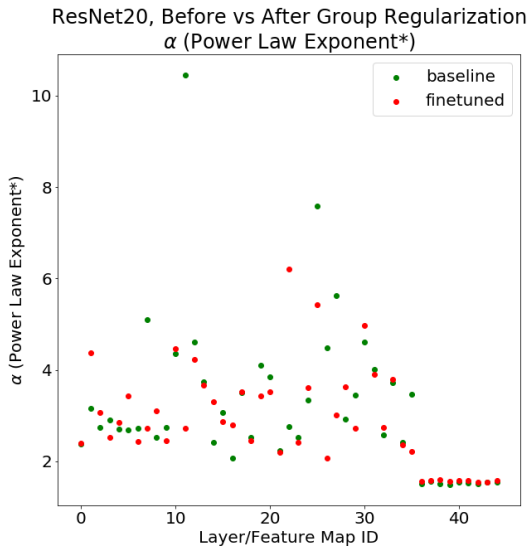
We can predict trends in the test accuracy—*without peeking at the test data!*

"pip install weightwatcher"

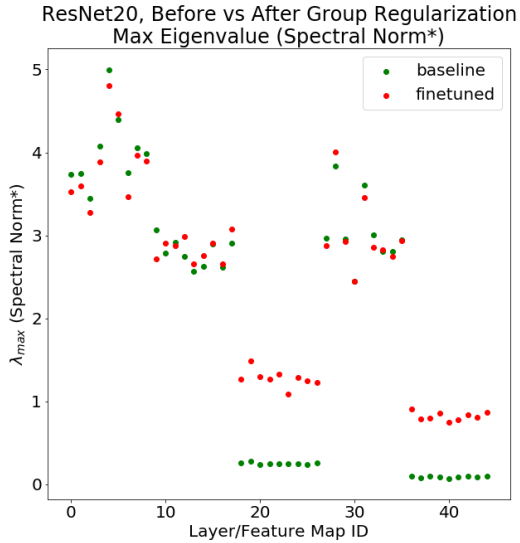
Predicting test accuracies better: Distilled Models

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 noticibly



(a) PL Exponents (α)



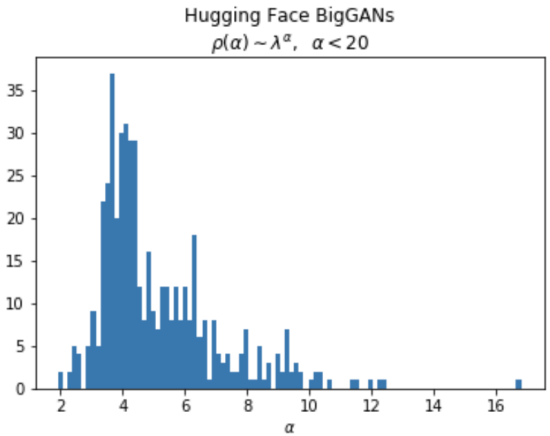
(b) Spectral Norm (λ_{max})

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).

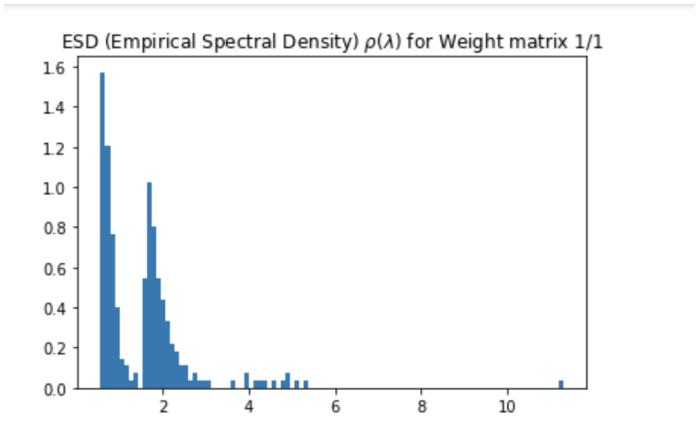
WeighWatcher: ESDs of GANs

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 αs



(b) Anamolous ESD.

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

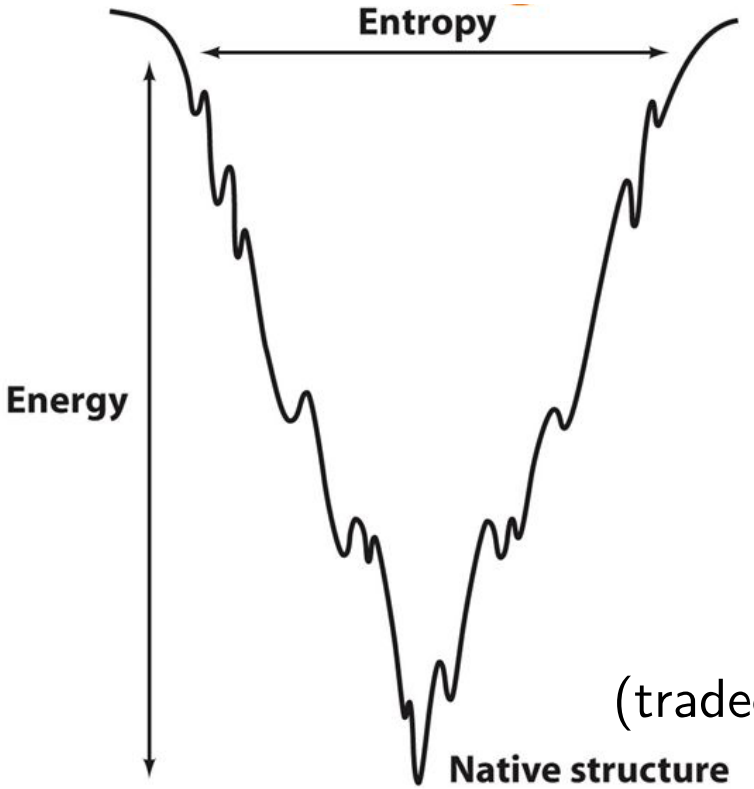
Join us:

- "pip install weightwatcher"—contribute to the repo.[‡‡]

[‡‡] *Don't do everything from scratch in a non-reproducible way. Make it reproducible!*

Outline

Implications: RMT and Deep Learning



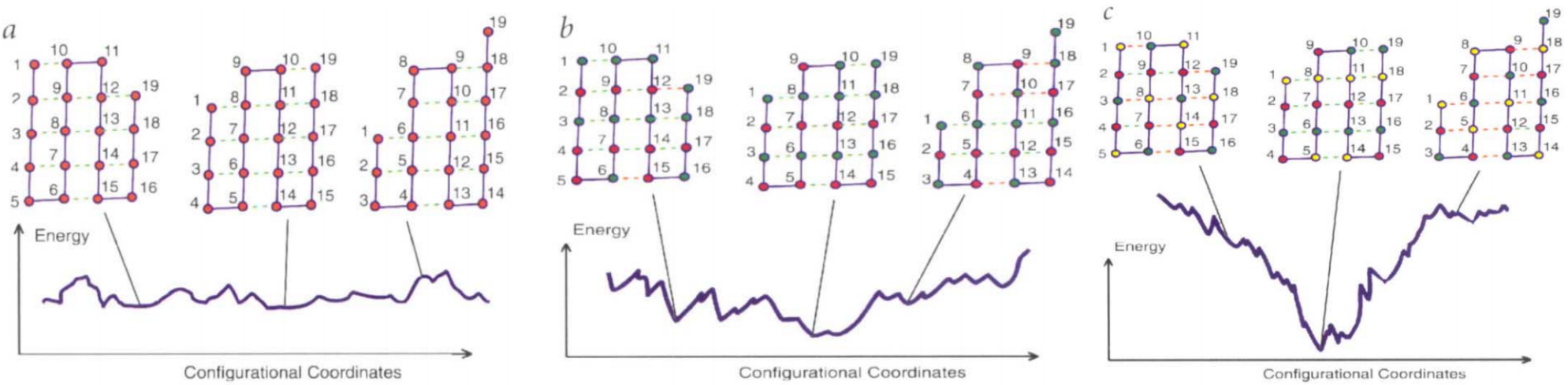
- Where are the local minima?
- How is the Hessian behaved?
- Are simpler models misleading?
- Can we design better learning strategies?

(tradeoff between Energy and Entropy minimization)

How can RMT be used to understand the Energy Landscape?

Implications: Minimizing Frustration and Energy Funnels

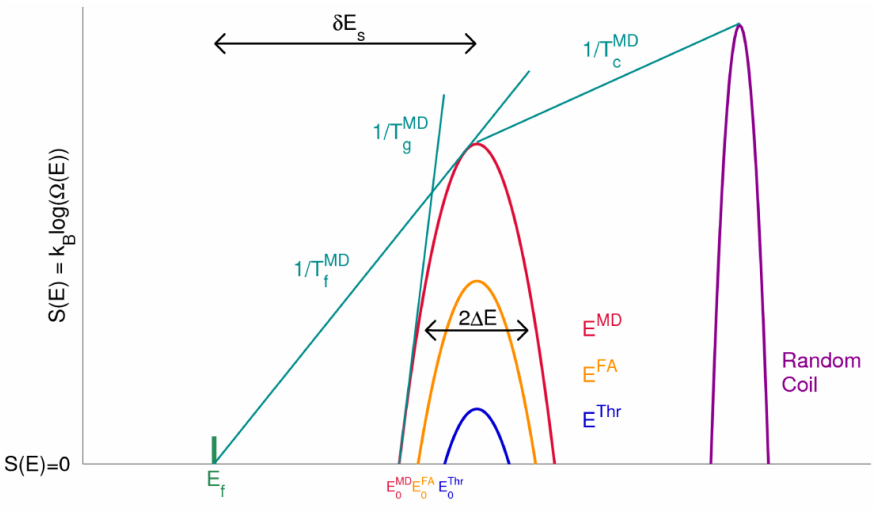
As simple as can be?, Wolynes, 1997

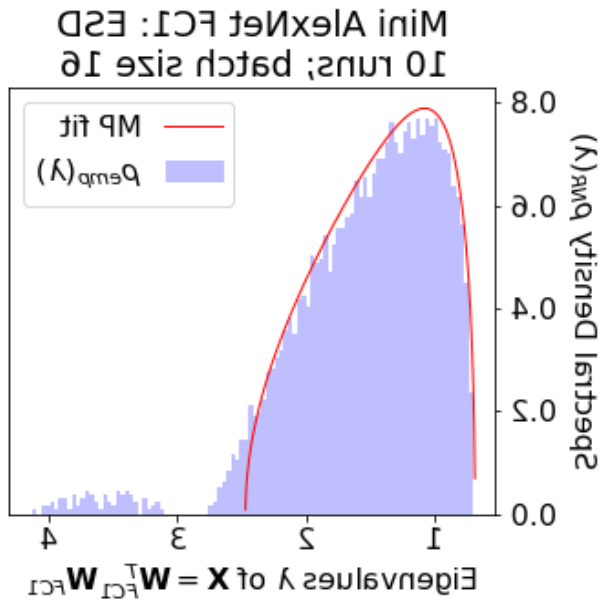


Energy Landscape Theory: "random heteropolymer" versus "natural protein" folding

Implications: The Spin Glass of Minimal Frustration

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

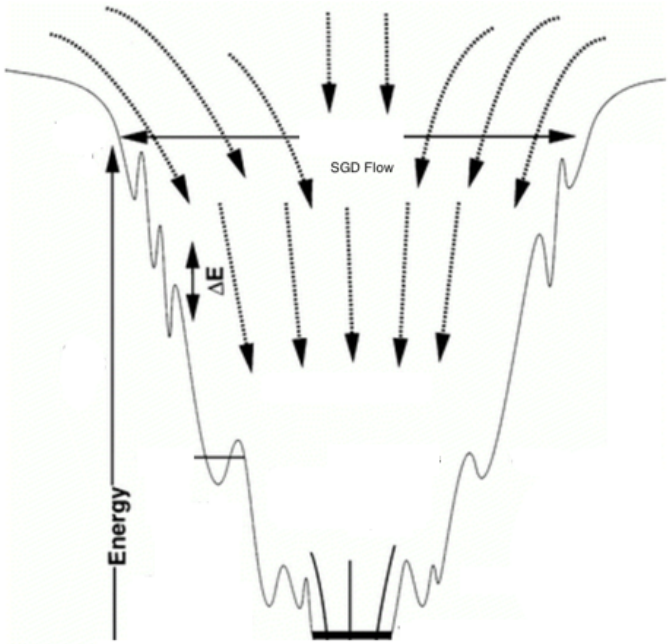




low lying Energy state in Spin Glass $\sim$ spikes in RMT

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 $\rightarrow$ Hypotheses $\rightarrow$ Build a Theory $\rightarrow$ 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"