Noncommutative probability for deep learning

Statistics Seminar
Department of Mathematics
The University of Mississippi

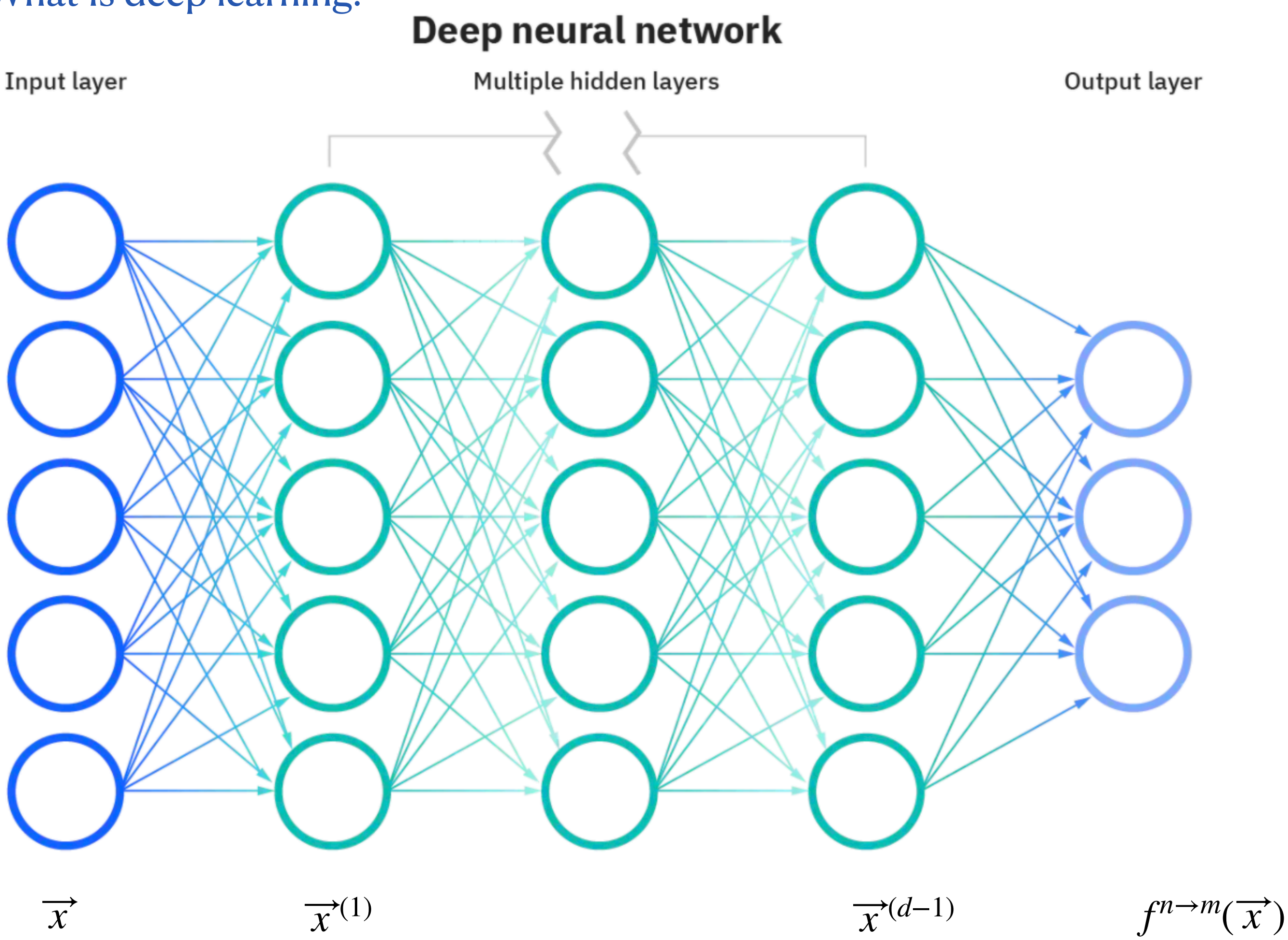
Cong Zhou October 14, 2021

Outline

1. **Background and motivation**
2. **Connection to random matrices**
3. **Essentials of random matrix and free probability theory**
4. **Condition for fast training**
5. **Conclusions**

Background and motivation

What is deep learning?



$$
\begin{aligned}
\vec{x}^{(0)} &= \vec{x} \\
Z^{(1)} &= W^{(1)}\vec{x}^{(0)} + b^{(1)} \\
\vec{x}^{(1)} &= \phi(Z^{(1)}) \\
&\vdots \\
\vec{x}^{(d)} &= \phi(Z^{(d)}) \\
Z^{(d+1)} &= W^{(d+1)}\vec{x}^{(d)} + b^{(d+1)} \\
f^{n\to m}(\vec{x}) &= Z^{(d+1)}
\end{aligned}
$$

Weights	$W^{(i)}$
Bias	$b^{(i)}$
Parameter vector	$\theta = (W^{(i)}, b^{(i)})$
Activation function	ϕ
Pre-activation	$Z^{(i)}$
Post-activation	$\overrightarrow{x}^{(i)}$
Input data	$\overrightarrow{x} = \overrightarrow{x}^{(0)}$
Output data	$f^{n\to m}(\overrightarrow{x}) = Z^{(d+1)}$
Targets	$\overrightarrow{y}$
Loss function (MSE)	$\text{Error}(W, b) = \dfrac{1}{2m}\|f^{n\to m}(\overrightarrow{x}) - \overrightarrow{y}\|^2$

$$\overrightarrow{x}^{(0)} = \overrightarrow{x}$$
$$Z^{(1)} = W^{(1)}\overrightarrow{x}^{(0)} + b^{(1)}$$
$$\overrightarrow{x}^{(1)} = \phi(Z^{(1)})$$
$$\vdots$$
$$\overrightarrow{x}^{(d)} = \phi(Z^{(d)})$$
$$Z^{(d+1)} = W^{(d+1)}\overrightarrow{x}^{(d)} + b^{(d+1)}$$
$$f^{n\to m}(\overrightarrow{x}) = Z^{(d+1)}$$

How to find the parameters?

Supervised learning Problem

Given examples $\{\overrightarrow{x}_\varepsilon\}_{\varepsilon \in \text{Examples}} \subset \mathbb{R}^n$, and labels $\{\overrightarrow{y}_\varepsilon\}_{\varepsilon \in \text{Examples}} \subset \mathbb{R}^m$. How to find parameters $\underline{W}^{(i)}$ and $\overrightarrow{b}^{(i)}$ so that $f^{n \to m}$ minimizes the prediction error:

$$\text{Error}(W, b) = \sum_{\varepsilon \in \text{Exmaples}} \|f^{n \to m}(\overrightarrow{x}_\varepsilon) - \overrightarrow{y}_\varepsilon\|^2$$

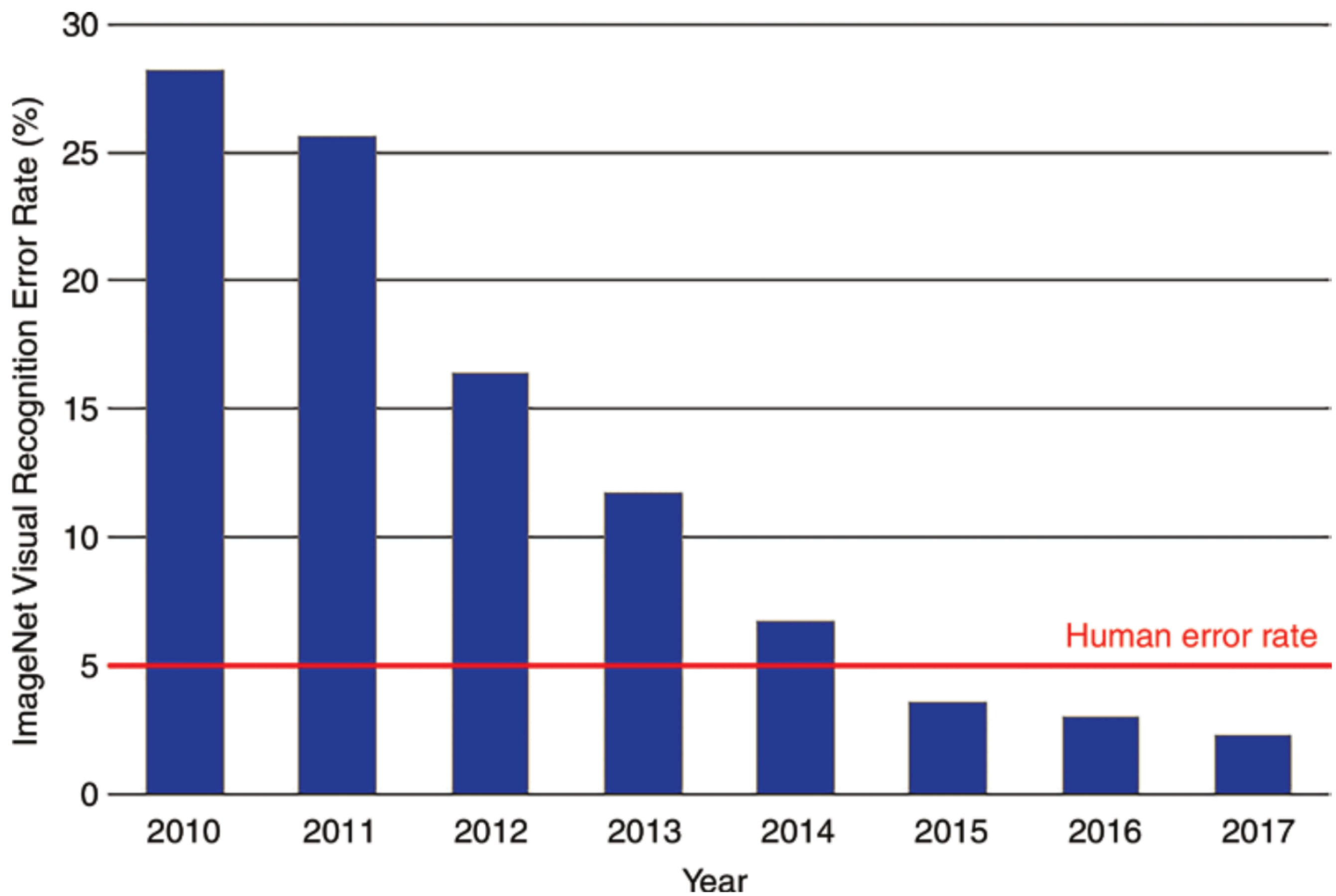
Supervised Learning: Solution Idea

0. **Invent** the architecture: depth d and layer width $n_1, \ldots, n_{d-1}$. Set $n_0 = n,\ n_d = m$.
1. **Initialization**: Pick parameters $\underline{W}, \overrightarrow{b}$ at **random**.
2. **Modify** parameters to **shrink** Error(W, b) by **gradient descent**.

$$\text{new}W^{(i)}_{j,k} := \text{old}W^{(i)}_{j,k} - \partial_{W^{(i)}_{j,k}}\text{Error}(W, b)$$

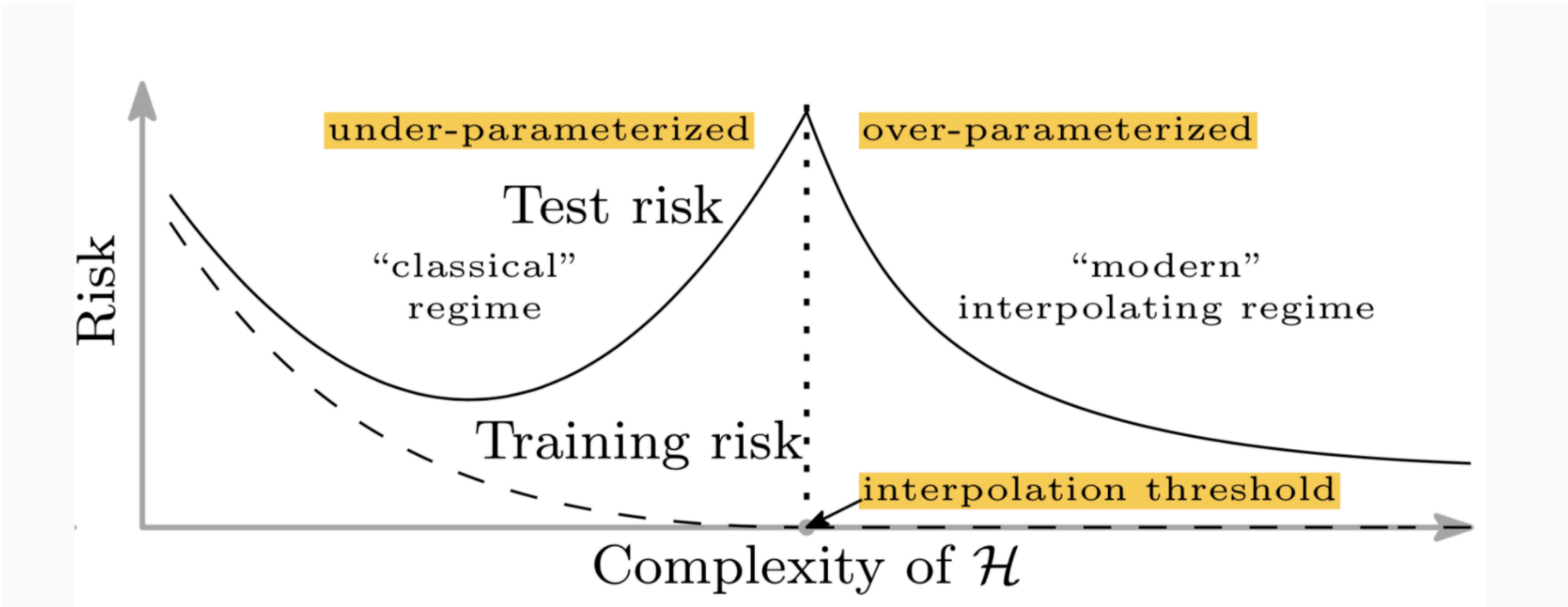
3. Repeat step 2 many time. **Hope** that the error is now small.

ImageNet Large Scale Visual Recognition Challenge Results



d=2. d=2. d=8. d=8. d=22. d=152. ————————-

Double descent phenomenon



Empirical Issue: d large has **vanishing and exploding gradients.**

Random initialization: $\partial_{W^{(i)}_{j,k}} \mathrm{Error}(W, b)$ is very large or small.

Today's story is about what happens on the initialization.

Connection to random matrices

Part 2: Products of random matrices

- Connect to neural network
- Limit theories for products of random matrices.

Definition. The **Input-Output Jacobian** matrix $\underline{J} = \mathrm{Jac}\{f^{n_0 \to n_d}\}$ is the $n_d \times n_0$ matrix

$$J_{ij}(\overrightarrow{x}) := \partial_j f_i^{n_0 \to n_d}(\overrightarrow{x})$$

Remark: $\partial_{W_{j,k}^{(i)}} \mathrm{Error}(W, b)$ can be written in terms of $\underline{J}$.

$\underline{J}$ when $\phi(x) = \max\{x,0\}$

Recall $f^{n_{i-1}\to n_i}(\overrightarrow{x}) := \phi\left(\underline{W}^{(i)}\overrightarrow{x} + \overrightarrow{b}^{(i)}\right)$ and want to compute

$$\underline{J} = \text{Jac}\left(f^{n_{d-1}\to n_d} \circ f^{n_{d-2}\to n_{d-1}} \circ \ldots \circ f^{n_1\to n_0}\right)$$

Since $\phi(x) = \max\{x,0\}$, $\phi'(x) = 1\{x > 0\}$, then the gradient of each layer is

$$\text{Jac}\left(f^{n_{i-1}\to n_i}\right) = \text{Diag}\left(1\{\underline{W}^{(i)}\overrightarrow{x} + \overrightarrow{b}^{(i)} > 0\}\right)\underline{W}^{(i)}.$$

Assume all **random matrices** are **symmetric**:

$$\text{Diag}\left(1\{\underline{W}^{(i)}\overrightarrow{x} + \overrightarrow{b}^{(i)} > 0\}\right) \stackrel{\text{d}}{=} \text{Diag}\left(\overrightarrow{X}^{(i)}\right)$$

where $\overrightarrow{X}^{(i)} \in \mathbb{R}^n$ has i.i.d entries $X_j^{(i)} \sim$ Bernoulli $\left(\frac{1}{2}\right)$.

By the chain rule, we shall expect

$$\underline{J} \stackrel{?}{=} \underbrace{\text{Diag}\left(\overrightarrow{X}^{(d)}\right) W^{(d)} \ldots \text{Diag}\left(\overrightarrow{X}^{(1)}\right) W^{(1)}}_{\underline{M}}$$

Could show $\underline{M} \stackrel{d}{=} \underline{J}$ up to conjugation by random ± 1Bernoulli's.

Essentials of random matrix and free probability theory

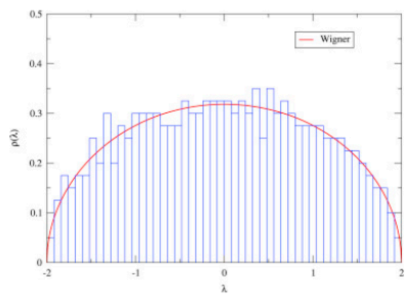
Dan-Virgil Voiculescu

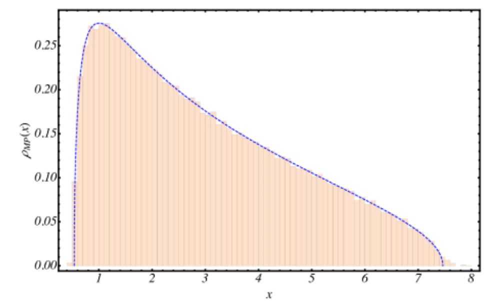
Hari Bercovici

Spectral density

Let $(A_n)_{n=1}^{\infty}$ be a sequence of $n \times n$ matrix.
The empirical spectral density of $\frac{1}{\sqrt{n}} A_n$ is:

$$\mu_{\frac{1}{\sqrt{n}} A_n}(\lambda) = \frac{1}{n} \sum_{j=1}^{n} \delta(\lambda - \lambda_j(A_n)).$$





For a sequence of matrices with increasing size, $(A_n)_{n=1}^{\infty}$, the limiting spectral density is:

$$\mu(\lambda) = \lim_{n \to \infty} \mu_{\frac{1}{\sqrt{n}} A_n}(\lambda).$$

For $z \in \mathbb{C} \backslash \text{supp}(\mu)$ the Cauchy transform is defined as:

$$G(z) = \sum_{n=1}^{\infty} \frac{m_n}{z^{n+1}}, \qquad m_n = \int_{\text{supp}(\mu)} t^n d\mu(t)$$

The moment generating function, $M(z)$, defined by a functional equation,

$$M(z) = \sum_{n=1}^{\infty} m_n z^n$$

The spectral density can be recovered from G using the Stieltjes inversion formula,

$$\mu(\lambda) = -\frac{1}{\pi} \lim_{\varepsilon \to 0^+} \Im G(\lambda + i\varepsilon).$$

The Cauchy transform G relates to the R-transform, R, defined by the functional equation,

$$1 + R(G(z)) = zG(z).$$

The moment series M and R-transform has the following relationship:

$$R(z(1 + M(z))) = M(z).$$

and the S-transform,

$$S(zG(z) - 1) = \frac{G(z)}{zG(z) - 1}$$

If a and b are **free independent**, then the distribution of sum $a+b$ can be computed using the R-transform:

$$
\begin{tikzcd}
\mu_a(\lambda) \arrow[r] & G_a(z) \arrow[r] & R_a \arrow[dr] & \\
 & & & R_a + R_b = R_{a+b} \rightarrow G_{a+b}(z) \rightarrow \mu_{a+b}(\lambda) \\
\mu_b(\lambda) \arrow[r] & G_b(z) \arrow[r] & R_b \arrow[ur] &
\end{tikzcd}
$$

S-transfrom:

$$
\begin{tikzcd}
\mu_a(\lambda) \arrow[r] & G_a(z) \arrow[r] & S_a \arrow[dr] & \\
 & & & S_a S_b = S_{ab} \rightarrow G_{ab}(z) \rightarrow \mu_{ab}(\lambda) \\
\mu_b(\lambda) \arrow[r] & G_b(z) \arrow[r] & S_b \arrow[ur] &
\end{tikzcd}
$$

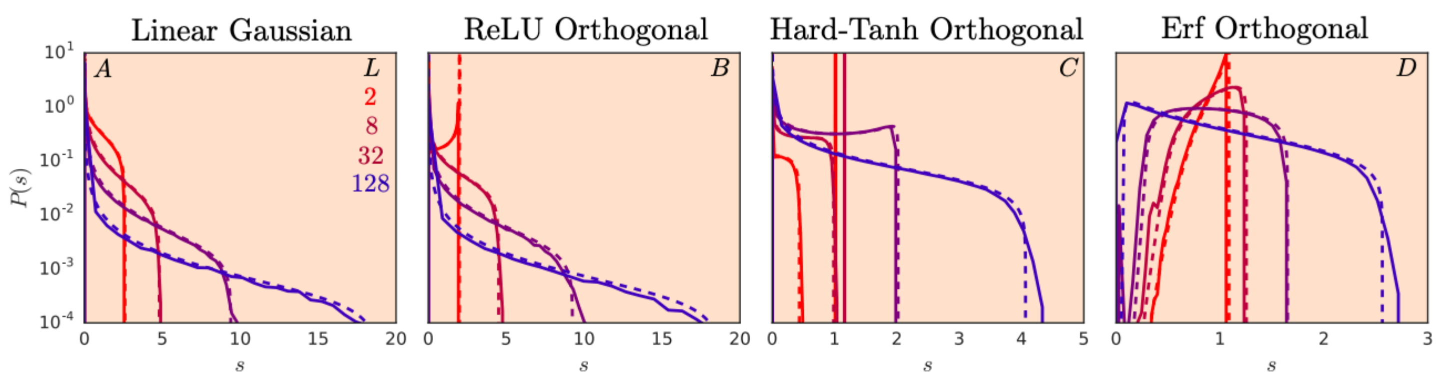
Classical independence	Free independence
X, Y are independent if one has $\mathbf{E} f(X) g(Y) = 0$ whenever f and g are such that $\mathbf{E} f(X) = 0$ $\mathbf{E} g(Y) = 0$	X, Y are freely independent if $\tau(f_1(X) g_1(Y) \dots f_k(X) g_k(Y)) = 0$ whenever f_i and g_i are such that $\tau(f_i(X)) = 0$ $\tau(g_i(Y)) = 0$
	$*$-free is $\{X, X^*\}$ and $\{Y, Y^*\}$ are free.

Define the $n_d \times n_0$ product random matrix $\underline{M}$

$$\underline{M} = \text{Diag}\left(\overrightarrow{X}^{(d)}\right) W^{(d)} \ldots \text{Diag}\left(\overrightarrow{X}^{(1)}\right) W^{(1)}$$

where $\overrightarrow{X}^{(i)} \in \mathbb{R}^n$ has i.i.d entries $X_j^{(i)} \sim \text{Bernoulli}\left(\frac{1}{2}\right)$.

W and $\text{Diag}\left(\overrightarrow{X}\right)$ are asymptotically free [Yang '19].



Condition for fast training

Conditions for trainability

The back-propagated signals (error signals) should not explode/vanish with depth.

Along the critical line, the gradient norms are well-behaved on average:

$$\delta^1 = \epsilon^T J, \qquad \epsilon = \delta^d \text{ (error signal)}$$

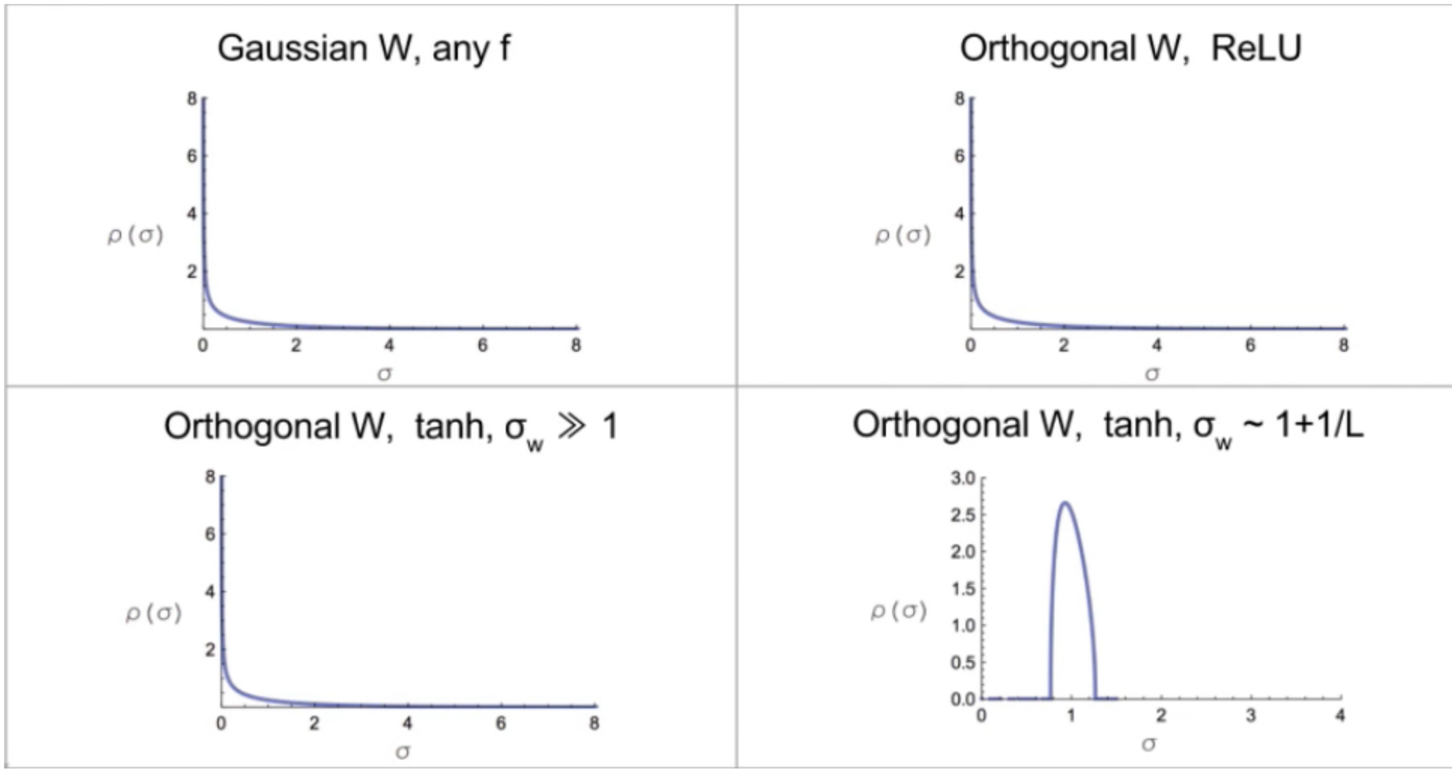
$$\underline{J} = \mathrm{Diag}\left(\overrightarrow{X}^{(d)}\right) W^{(d)} \ldots \mathrm{Diag}\left(\overrightarrow{X}^{(1)}\right) W^{(1)}$$

$$\mathbb{E}_\epsilon[\|\delta^1\|^2] = \mathbb{E}_\epsilon[\epsilon^T J J^T \epsilon] = tr[JJ^T] = \sum_{i=1}^{n} \sigma_i^2$$

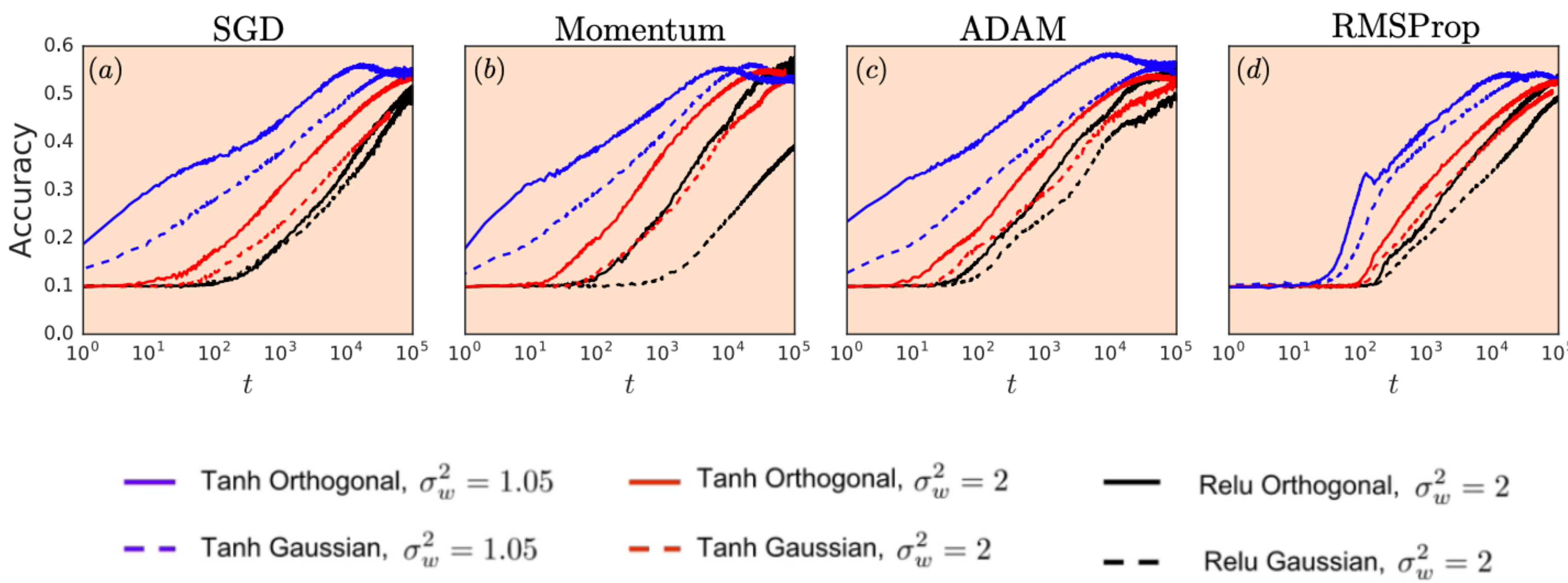
$$\frac{1}{n}\sum_{i=1}^{n} \sigma_i^2 = 1 \Leftarrow \text{trainable}$$

$$\underline{J} = \mathrm{Diag}\left(\overrightarrow{X}^{(d)}\right) W^{(d)} \ldots \mathrm{Diag}\left(\overrightarrow{X}^{(1)}\right) W^{(1)}$$

Jacobian spectra for large depth



Implications for training speed



Conclusions

Conclusion:

Free probability and random matrix theory provides powerful tools for studying deep learning:

- Jacobian (showed how to equilibrate the distribution of singular values of the input-output Jacobian for faster training).
- Hessian

Thank you!