Mathematical foundations of Machine Learning 2025 – lesson 8

Shai Dekel

TEL AVIV UNIVERSITY

Analysis of deep learning models: an approximation theoretical perspective

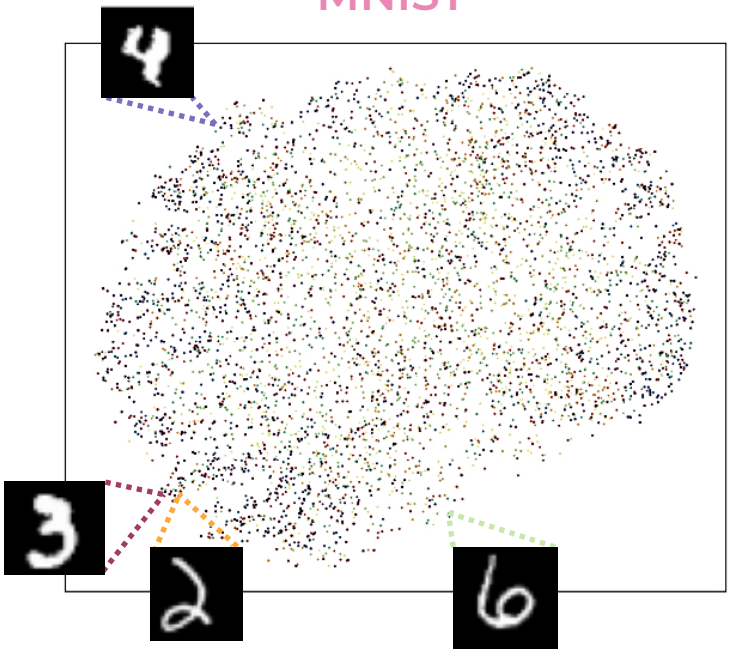
Our goal is to provide a holistic mathematical foundation for AI (classical ML and DL) through:

Function space theory, Approximation Theory, Geometric Harmonic Analysis

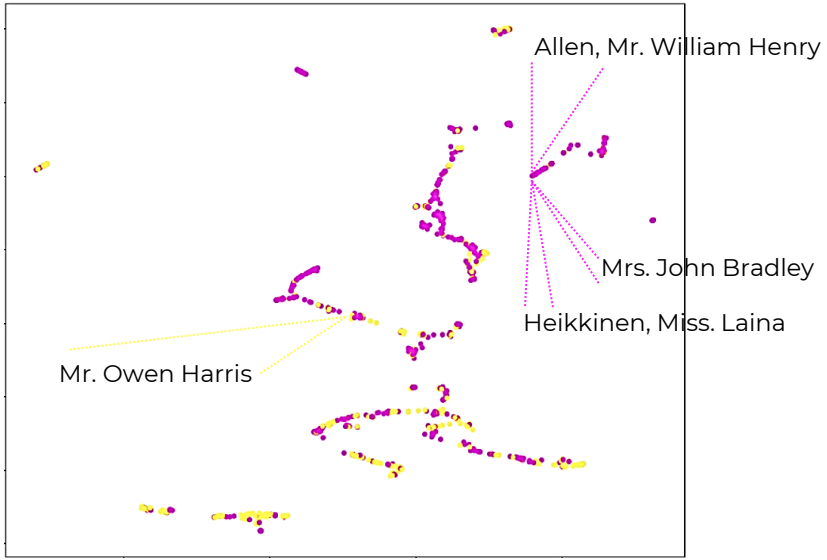
ML vs. DL

- When do we want to use classic ML vs. DL for classification?
- Is there ‘geometry’ in the feature space?

MNIST

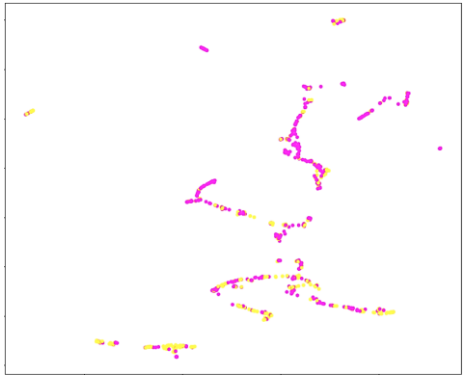


TITANIC



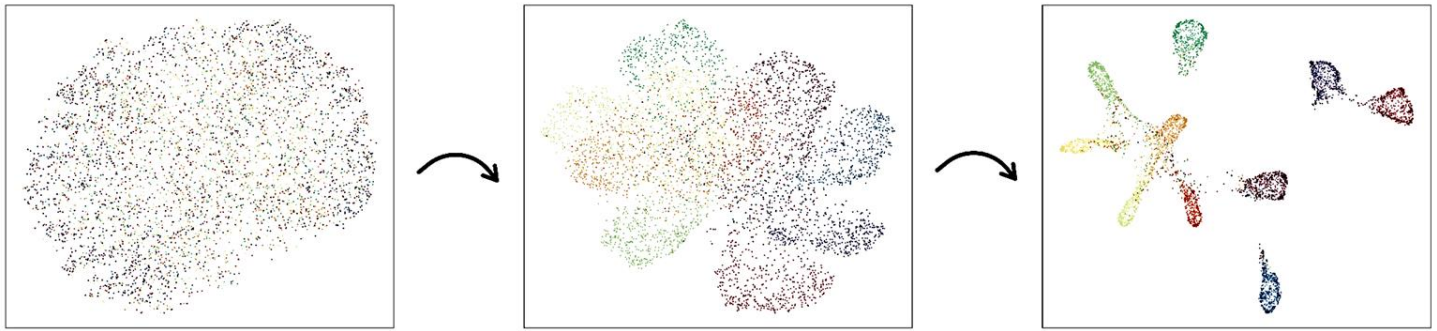
McInnes, L., Healy, J., and Melville, J., “UMAP: Uniform Manifold Approximation and Projection for Dimension Reduction”, arXiv e-prints, 2018.

ML vs. DL



- Good separability in input feature space → ML
- All successful Machine Learning algorithms look for this geometry:
    - Support Vector Machines, Random Forest, Gradient Boosting, etc.
- **If not, can we transform to a better feature space through feature engineering/deep learning (CNN, Resnets, Transformers etc)?**

MNIST DATASET



[1] UMAP of trained ConvNet on MNIST Dataset - Ben-Shaul, I. and Dekel, S., “Sparsity-Probe: Analysis tool for Deep Learning Models”, arXiv e-prints, 2021.
[2] F. Chollet, Deep Learning with Python, Manning, Nov. 2017.

Deep Learning: smoothness analysis across layers

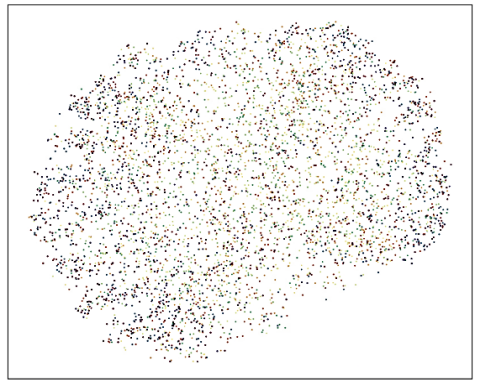
* I. Ben-Shaul, S. Dekel and O. Elisha, Sparse Besov space analysis of deep learning representation layers in high dimensions, Pure and Applied Functional Analysis, to appear.

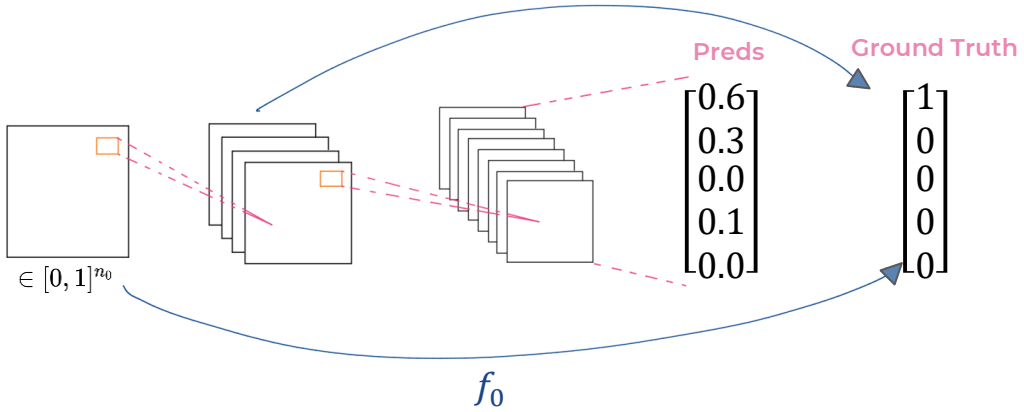
Function space representation

- Assume we have a dataset of $\sqrt{n_0} \times \sqrt{n_0}$ grayscale images.
- We concatenate the pixel values to vectors of size n_0
- We normalize the pixels values to $[0,1]$
- Each image is associated with one of L class labels.
- We map each label to its one-hot-encoding in $\mathbb{R}^L$
- Thus, each image is now a sample of a function

$$f_0 : [0,1]^{n_0} \to \mathbb{R}^L$$

In general f_0 will look like spaghetti...no separability!

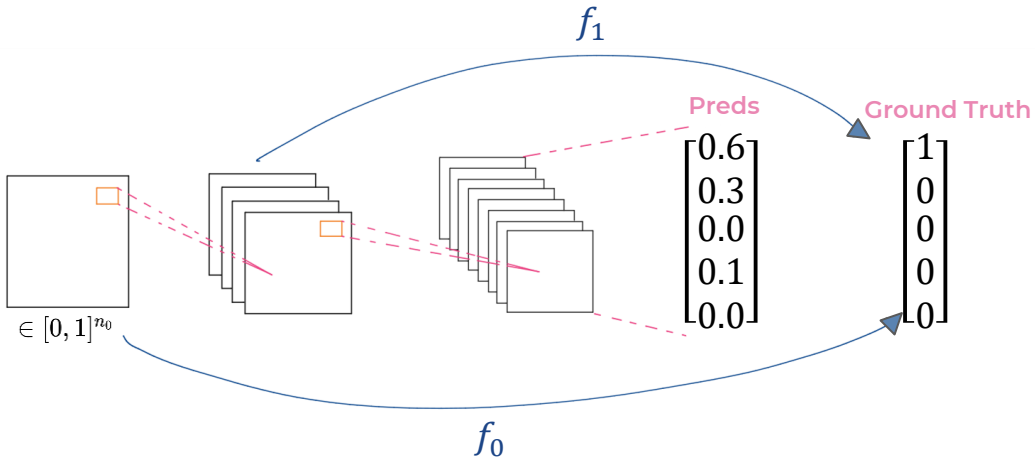




Function space representation: inner layers

- For each k-th inner layer we do something similar (during or after the training).
- We pass each image through the network until the k-th layer.
- We concatenate the features of the image into a vector of size n_k.
- The feature values are normalized to $[0,1]$.
- This implies we now have samples of a function

$$f_k : [0,1]^{n_k} \to \mathbb{R}^L$$



Forest wavelet Sparsity

For simplicity, assume $w_j = \frac{1}{J}, \quad 1 \le j \le J.$

$$\mathcal{N}_\tau(f, \mathcal{F}) := \frac{1}{J}\left(\sum_{\Omega \in \mathcal{F}} \|\psi_\Omega\|_p^\tau \right)^{1/\tau}$$

p- Norm for approximation , typically $p = 2$

Tree Besov Smoothness

$$|f|_{\mathcal{B}_\tau^{\alpha,r}(\mathcal{T})} := \left(\sum_{\Omega \in \mathcal{T}} \left(|\Omega|^{-\alpha} \, \omega_r(f, \Omega)_\tau \right)^\tau \right)^{1/\tau}.$$

$$\alpha > 0, \; r \geq 1, \quad 1/\tau := \alpha + 1/p.$$

Forest Besov Smoothness

$$|f|_{\mathcal{B}_\tau^{\alpha,r}(F)} := \frac{1}{J}\left(\sum_{j=1}^{J} |f|^\tau_{\mathcal{B}_\tau^{\alpha,r}(T_j)}\right)^{1/\tau}$$

$$\alpha > 0, \qquad 1/\tau := \alpha + 1/p.$$

Besov index (for fixed p)

$$\sup\left\{\alpha : |f|_{\mathcal{B}_\tau^{\alpha,r}(F)} < \infty\right\}$$

Theorem

$$|f|_{\mathcal{B}_\tau^{\alpha,r}(F)} \sim N_\tau(f, F)$$

A modern recipe from the old Approximation Theory cook-book

- Assume we have samples of a function $f:[0,1]^n \to \mathbb{R}^L$
- Construct a piecewise constant approximation using a Random Forest.
- Create a Wavelet decomposition of the Random Forest.
- Numeric algorithm to estimate minimal τ^* for which

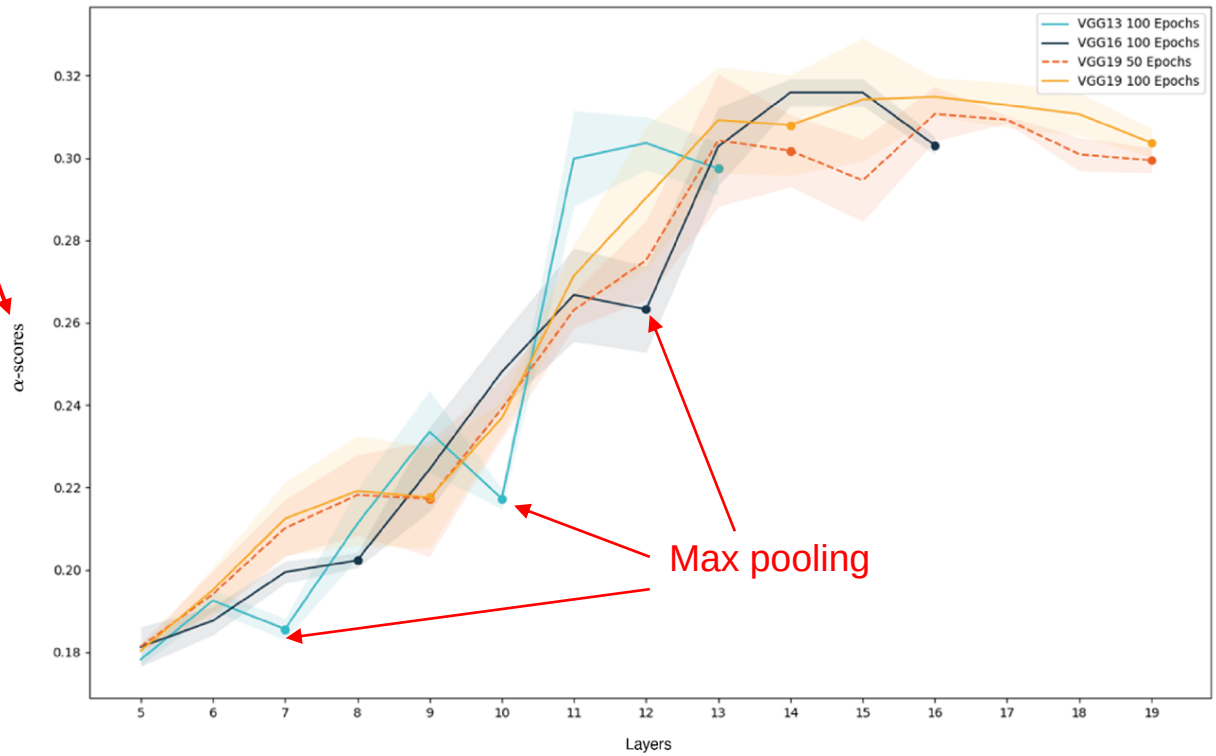
$$N_\tau(f,F) = \frac{1}{J}\left(\sum_{\Omega\in F}\|\psi_\Omega\|_2^\tau\right)^{1/\tau} < \infty$$

- The weak type smoothness of the function is estimated as

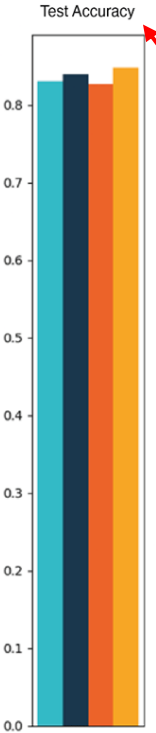
$$\alpha^* = \frac{1}{\tau^*} - \frac{1}{2}, \quad 0 < \tau^* < 2.$$

CIFAR10 / VGG: added value of added VGG layers?

Computed on training data

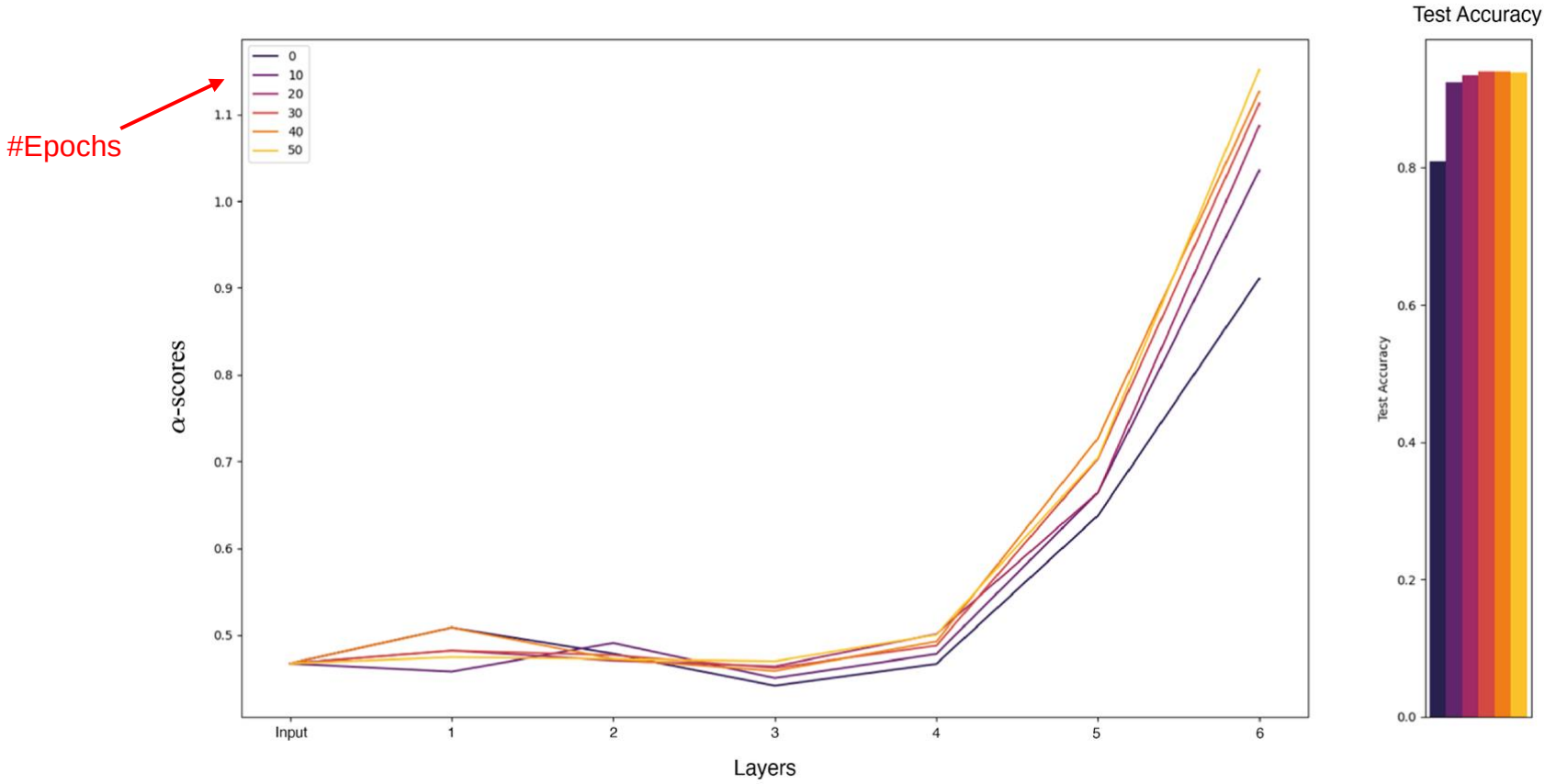


Max pooling



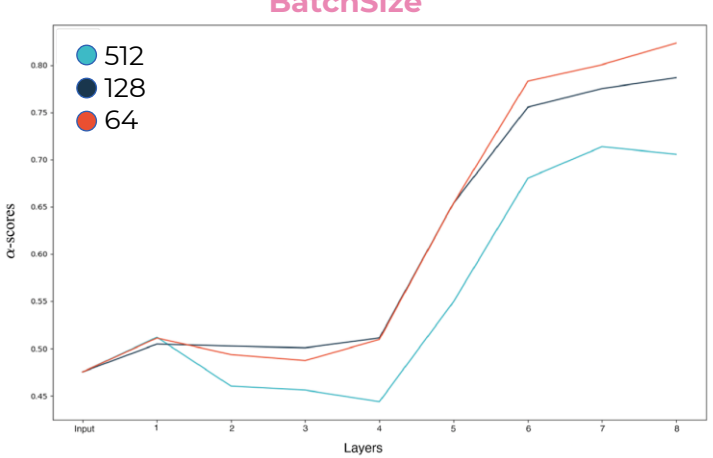
Computed on testing data

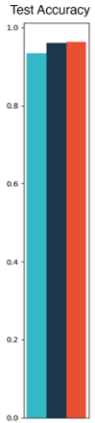
MNIST-1D/ ConvNet: geometry of the training process



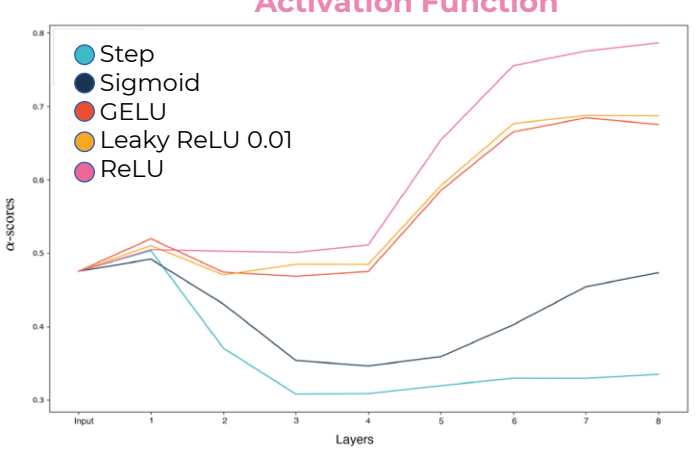
MNIST-1D/ ConvNet: HyperParameter tuning

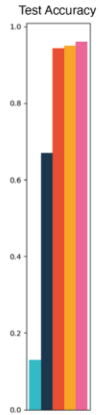
BatchSize



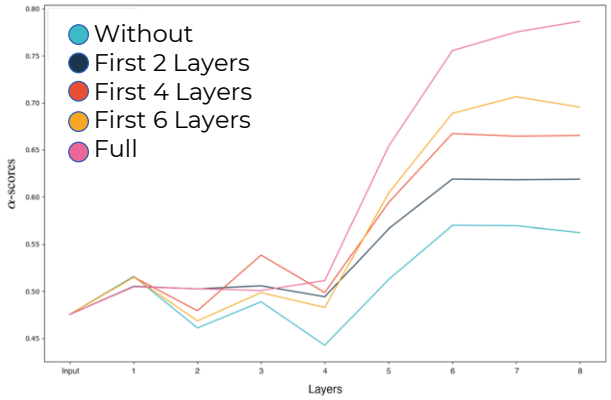


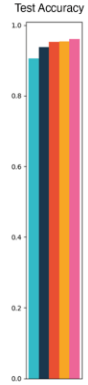
Activation Function



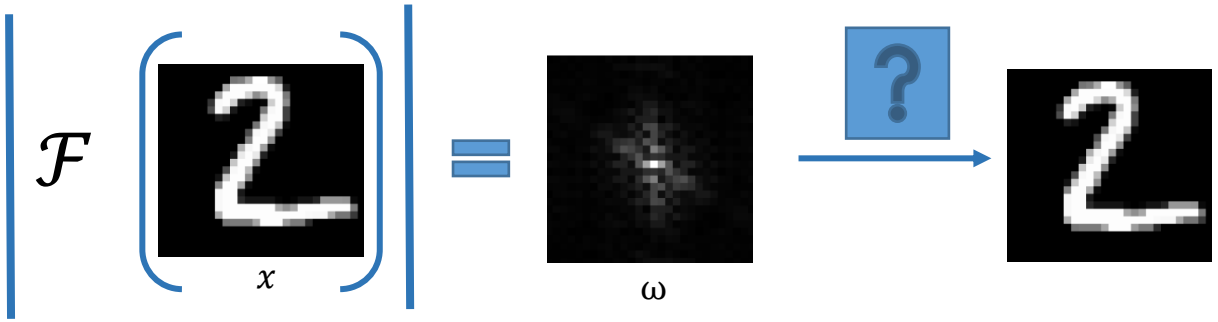


BatchNorm



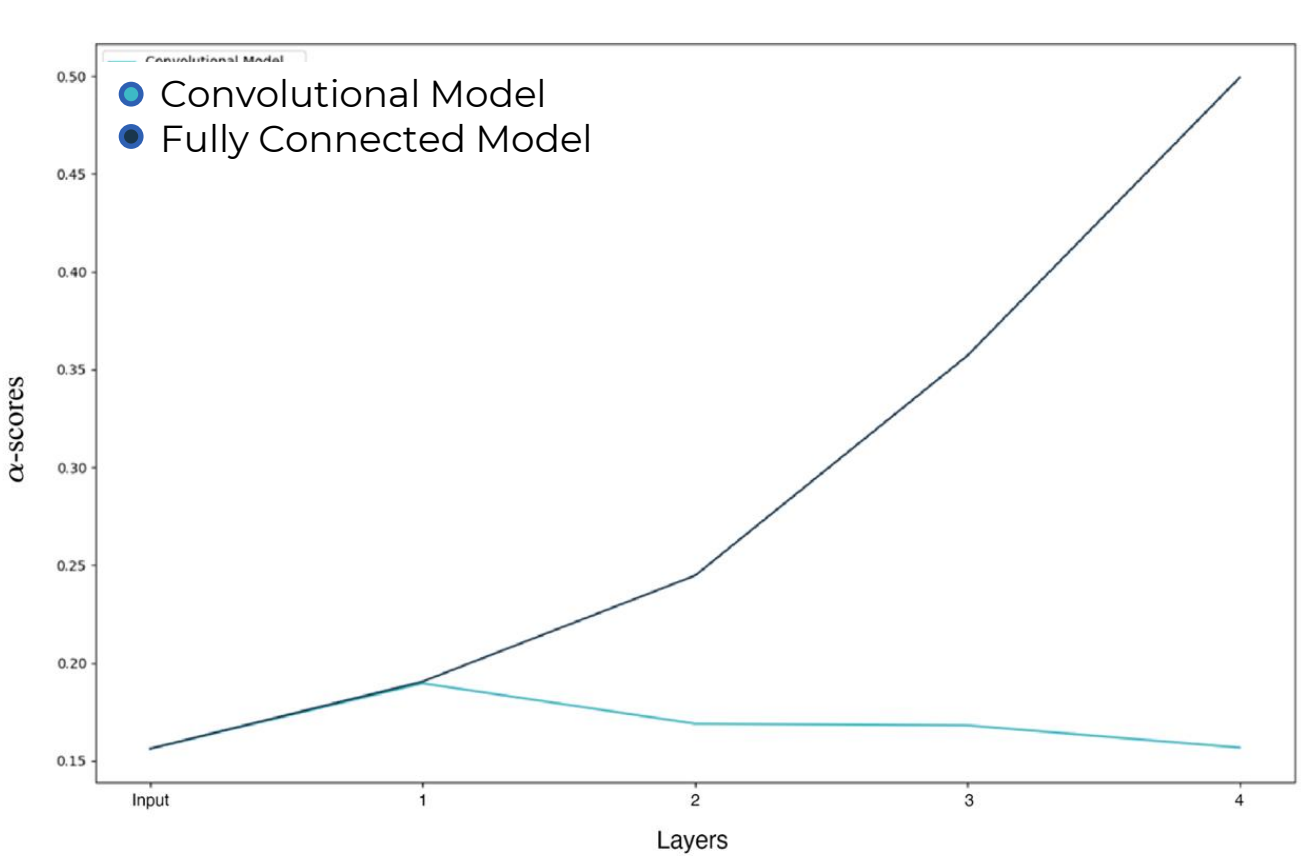


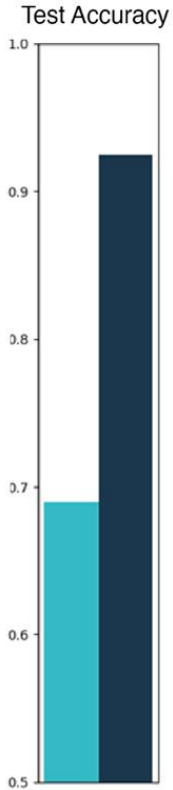
Phase Retrieval: problem formulation



- $x \in \mathbb{R}^{n \times n}$, where $\mathcal{F}(x) = \omega\, e^{-i\varphi}$, ω magnitude and φ phase of Fourier transform
- PR is an ill-posed inverse problem: recover x from ω

Unraveling bad architectural choices: Phase Retrieval example



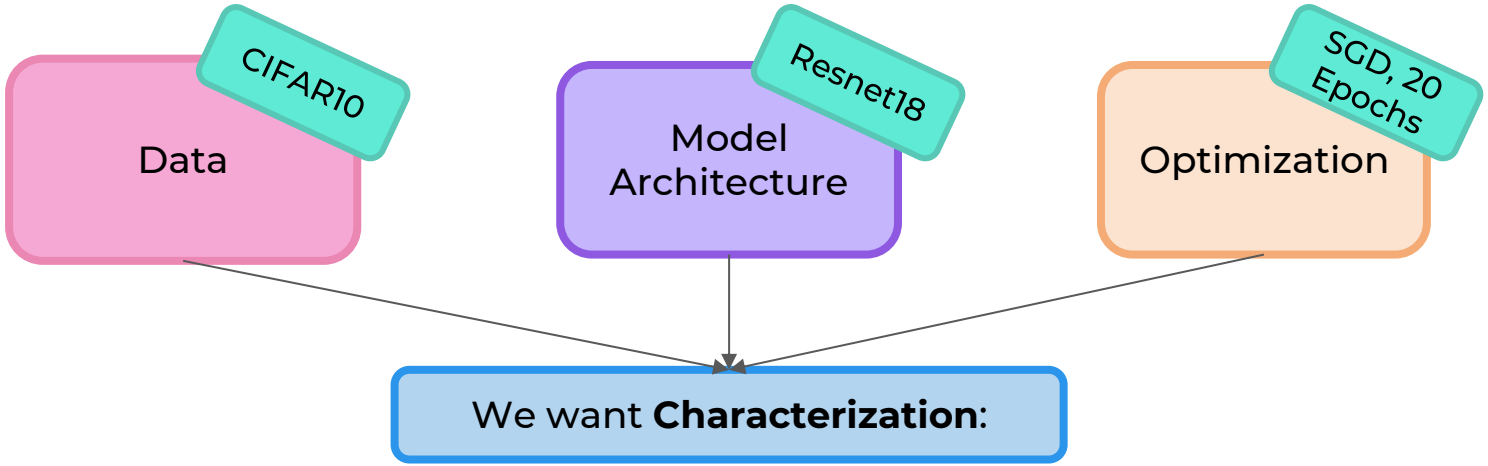


MNIST dataset : classification from Fourier magnitude data

Approximation theory of deep learning

* I. Ben-Shaul, S. Dekel and O. Elisha, Sparse Besov space analysis of deep learning representation layers in high dimensions, Pure and Applied Functional Analysis, to appear.

Theoretical Deep Learning



Approx. Rate - Approximation error with N added layers $\|f - f_N\|$

rate ← smoothness

rate → smoothness

Smoothness – Function weak type smoothness of layer representations of f

Statistical Dynamics

Information Theory

Approximation Theory

Interpretability Methods

. . .

Approximation Spaces

Let $\Phi := \{\Phi_N\}_{N\geq 0}$, each Φ_N is a set of functions in a (quasi) Banach space X, satisfying:

(i) $0 \in \Phi_N$, $\Phi_0 := \{0\}$,

(ii) $\Phi_N \subset \Phi_{N+1}$,

(iii) $a\Phi_N = \Phi_N$, $\forall a \neq 0$,

(iv) $\Phi_N + \Phi_N \subset \Phi_{cN}$, for some constant $c(\Phi)$,

(v) $\overline{\bigcup_N \Phi_N} = X$,

(vi) Each $f \in X$ has a near best approximation from Φ_N. That is, there exists a constant $C(\Phi)$, such that for any N, one has $\varphi_N \in \Phi_N$,

$$\|f - \varphi_N\|_X \leq C E_N(f)_X, \qquad E_N(f)_X := \inf_{\varphi\in\Phi_N} \|f - \varphi\|_X .$$

Examples for Φ_N

Linear

- Trigonometric polynomials of degree $\le N$, $X = L_p\left(\left[-\pi,\pi\right]^n\right)$.
- Algebraic polynomials of degree $\le N$, $X = L_p\left[-1,1\right]$.
- Uniform dyadic knot piecewise polynomials over pieces of length 2^{-N}, of fixed order r, $X = L_p\left[0,1\right]$.
- Shift invariant refinable spaces $\Phi_N := S(\phi)^{2^{-N}}$, $S(\phi) \subset S(\phi)^{1/2}$, $X = L_p\left(\mathbb{R}^n\right)$.

Nonlinear/Adaptive

- Rational functions of degree $\le N$, $X = L_p\left[-1,1\right]$,
- Free knot piecewise polynomials of fixed order r over N non-uniform intervals, $X = L_p\left[0,1\right]$.
- N-term wavelets $\Phi_N = \Sigma_N := \left\{ \sum_{\#I \le N} c_I \psi_I \right\}$, $X = L_2\left(\mathbb{R}^n\right)$.

Def Approximation spaces for $\alpha > 0$, $0 < q \le \infty$, $f \in X$,

$$|f|_{A_q^\alpha} := \begin{cases} \left(\sum_{N=1}^{\infty} \left[N^\alpha E_N(f) \right]^q \frac{1}{N} \right)^{1/q}, & 0 < q < \infty, \\ \sup_{N \ge 1} N^\alpha E_N(f), & q = \infty. \end{cases}$$

$$\|f\|_{A_q^\alpha} := \|f\|_X + |f|_{A_q^\alpha}.$$

One can show

$$|f|_{A_q^\alpha} \sim \begin{cases} \left(\sum_{m=0}^{\infty} \left[2^{m\alpha} E_{2^m}(f) \right]^q \right)^{1/q}, & 0 < q < \infty, \\ \sup_{m \ge 0} 2^{m\alpha} E_N(f), & q = \infty. \end{cases}$$

Goal: Fully characterize approximation spaces by smoothness spaces (iff)

Characterization of approximation spaces

1. Trigonometric polynomials

$X = L_p[-\pi, \pi]$, $1 \le p \le \infty$, Φ_N trigonometric polynomials of degree N

$$A_q^\alpha(L_p) \sim B_q^\alpha(L_p).$$

2. Dyadic univariate piecewise polynomials

$X = L_p[0,1]$, Φ_N piecewise polynomials of degree $d \ge 0$, over uniform subdivision of 2^N intervals.

For $1 \le p \le \infty$, $\alpha < r - 1 + 1/p$, $0 < q \le \infty$,

$$A_q^\alpha(L_p) \sim B_q^\alpha(L_p).$$

3. Adaptive non-uniform univariate piecewise polynomials

$$A_\tau^\alpha(L_p) \sim B_\tau^\alpha, \quad \frac{1}{\tau} = \alpha + \frac{1}{p}.$$

Characterization of Decision tree approximation

- Φ_N are tree-based wavelet approximations with $\leq N$ wavelets.
- Suitable for high dimensions
- Approximation spaces characterization? Complex, but exists!

THEOREM 3.3. *Let* $0 < \gamma < \alpha$, $0 < q \leq \infty$, *and* $0 < p < \infty$; *then*

$$A_{q,\tau}^{\gamma,r}(L_p) = (L_p, \mathcal{GB}_{\tau}^{\alpha,r})_{\frac{\gamma}{\alpha},q}, \tag{3.8}$$

where $1/\tau := \alpha + 1/p$.

- S. Dekel and D. Leviatan, Adaptive multivariate approximation using binary space partitions and geometric wavelets, SIAM Journal on Numerical Analysis 43 (2005), 707-732

Characterization of deep learning approximation?

- Φ_N can be networks with width $\leq W$ (can be comparable to input dimension $W \sim n$) and number of layers $\leq N$.
- We could assume nonlinearity is ReLU.
- Approximation spaces characterization? Very tricky!

* R. Gribonval, G. kutyniok, M. Nielsen and F. Voigtlaender, Approximation spaces of deep neural networks, Constructive Approximation 55 (2022), 259-367

Smoothness of input function → Approximation rate?

Still, one can certainly prove Jackson-type theorems that bound the degree of approximation using smoothness norms such as Sobolev or Besov. To this end, we first recall the Sobolev space $W_p^r(\Omega)$, $1 \leq p \leq \infty$, $\Omega \subseteq \mathbb{R}^n$, consisting of functions with distributional derivatives up to order r satisfying

$$\|f\|_{W_p^r(\mathbb{R}^n)} := \sum_{|\alpha| \leq r} \|\partial^\alpha f\|_{L_p(\Omega)} < \infty.$$

In this work we are focused on cases where the dimension n is large, since even the simplest classification network trained on a computer vision dataset consisting of very small images of size 32×32, produces inner layer dimensions of up to $n = 16,000 - 64,000$ neurons. However, when trying to approximate using a network functions from the unit ball of $W_p^r([0,1]^n)$, one encounters the 'curse of dimensionality'. That is, in the worst case, approximation with error $\varepsilon > 0$, requires a network of size $\sim \varepsilon^{-n/r}$ (e.g. there are lower bounds [34, 33]).

Smoothness of input function → Approximation rate?

sions of up to $n = 16,000 - 64,000$ neurons. However, when trying to approximate using a network functions from the unit ball of $W_p^r([0,1]^n)$, one encounters the 'curse of dimensionality'. That is, in the worst case, approximation with error $\varepsilon > 0$, requires a network of size $\sim \varepsilon^{-n/r}$ (e.g. there are lower bounds [34, 33]).

$$\varepsilon = 0.1, n = 1,000, r = 1 \Rightarrow \varepsilon^{-n/r} = 10^{1000}$$

Approximation of smooth functions without a ‘curse’

2.1). These results are useful in cases where $n >> r$, that is, the dimension is relatively higher than the given smoothness.

Theorem 2.1. [4] *Let* $n \geq 2$, $r \in \mathbb{N}$, $f \in W_2^r(\mathbb{R}^n)$ *such that* $\|\hat{f}\|_{L_1(\mathbb{R}^n)} < \infty$, *where* $\hat{f}$ *is the Fourier transform of* f *and* $0 < \varepsilon \leq 1$. *Then there exist:*

(i) *A deep MLP network* $\tilde{f}_{MLP}$ *with ReLU activations, of depth* $O(n^2\varepsilon^{-2/r})$ *and* $O(n^2\varepsilon^{-(2+2/r)})$ *neurons, such that*

$$\|f - \tilde{f}_{MLP}\|_{L_2([0,1]^n)} \leq c \max(|f|_{r,2}^2, \|\hat{f}\|_1)\varepsilon.$$

(ii) *A deep multiplicative network* $\tilde{f}_{MUL}$ *of depth* $O(n\varepsilon^{-1/r})$ *and* $O(n\varepsilon^{-(2+1/r)})$ *neurons, such that*

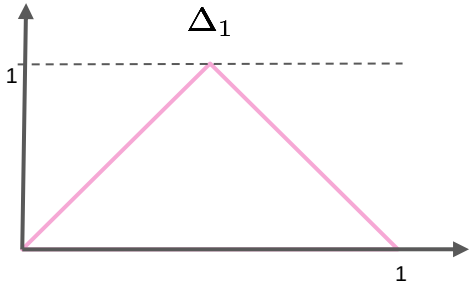
$$\|f - \tilde{f}_{MUL}\|_{L_2([0,1]^n)} \leq c \max(|f|_{r,2}^2, \|\hat{f}\|_1)\varepsilon.$$

*I. Ben-Shaul, T. Galanti. S. Dekel, Exploring the approximation capabilities of multiplicative neural networks for smooth functions, TMLR (2023).

The “Sawtooth” realization

Let $\Delta_1 : \mathbb{R} \to [0,1]$ be defined by

$$\Delta_1(x) := \begin{cases} 2x, & 0 \le x < 1/2, \\ -2x+2, & 1/2 \le x \le 1, \\ 0, & \text{else.} \end{cases} \tag{2.3}$$



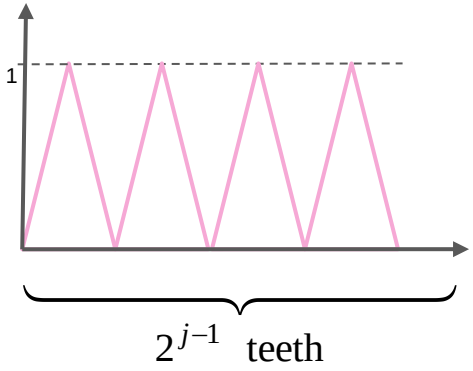
We then define $\Delta_j := \Delta_{j-1} \circ \Delta_1$, for $j \ge 2$. The sawtooth function Δ_j has 2^{j-1} ‘teeth’, as depicted in Figure 3. The function Δ_1 has the following representation

$$\begin{aligned} \Delta_1(x) &:= 2(x)_+ - 2(2x-1)_+ + 2(x-1)_+ \\ &= \sigma(2x) - 2\sigma(2x-1) + \sigma(2x-2), \end{aligned}$$

where σ is the ReLU nonlinear activation. Therefore, it can be realized using a network block composed of two MLP layers

$$x \to (2x, 2x, 2x) + (0, -1, -2) \to \sigma \to (x_1, x_2, x_3) \to x_1 - 2x_2 + x_3.$$

This implies that Δ_j, $j \ge 1$, can be realized as a composition of j blocks, which is a network of depth $2j$.



Approximation rate →Smoothness of input function?

It can be shown [19] that for any $\alpha < 1/\tau$

$$|\Delta_j|_{\mathcal{B}^\alpha_\tau} \sim 2^{j\alpha}. \quad (2.6)$$

(assignment III)

This is a pro-typical example of a function that, one the one hand, has a compositional structure, that allows it to be represented by a relatively shallow network of depth $\sim j$ and yet its Besov smoothness semi-norm is exponentially larger with j. That is, when initially inspecting the smoothness of the input function, it is not obvious that it can be approximated or, as in this special case, even realized, by a relatively shallow network.

In going further, let us denote by $E_N(f)_p$ the degree of approximation of a neural network with width of size $\sim n$ and depth of size N of a function $f \in L_p([0,1]^n)$. Let us recall the approximation space $\mathcal{A}^\gamma_\infty(L_p)$, defined by functions in L_p for which

$$|f|_{\mathcal{A}^\gamma_\infty(L_p)} := \sup_N N^\gamma E_N(f)_p < \infty.$$

Approximation rate →Smoothness of input function?

Since Δ_j can be realized by a network of width 3 and $2j$ layers, we have that

$$E_N\left(\Delta_j\right)_p \sim \begin{cases} 1, & N < 2j, \\ 0, & N \geq 2j. \end{cases}$$

Therefore

$$|\Delta_j|_{\mathcal{A}^\gamma_\infty(L_p)} \sim j^\gamma.$$

Combining this observation with (2.6), we see that we do not have a continuous embedding of $\mathcal{B}^\alpha_\tau$ in $\mathcal{A}^\gamma_\infty(L_p)$ for any values of $\alpha, \gamma > 0$.

$$\left|\Delta_j\right|_{B^\alpha_\tau} \underset{?}{\leq} c\left|\Delta_j\right|_{A^\gamma_\infty} \Rightarrow 2^{j\alpha} \leq cj^\gamma \Rightarrow j\alpha \leq c_1\gamma \log j \Rightarrow j \leq c_2 \log j$$

Back to smoothness analysis of layers

Yet, with the realization of $f_0 := \Delta_j$ as a neural network, we have that $|f_0|_{\mathcal{B}^\alpha_\tau} \sim 2^{j\alpha}$ and then, as we proceed trough the layers, the Besov semi-norm of the inner layer representations decreases exponentially, since $f_2 = \Delta_{j-1}$, $f_4 = \Delta_{j-2}$, etc. That is, $|f_2|_{\mathcal{B}^\alpha_\tau} \sim 2^{(j-1)\alpha}$, $|f_4|_{\mathcal{B}^\alpha_\tau} \sim 2^{(j-2)\alpha}$, etc.

Theorem 2.2. *Let $f : [0,1] \to \mathbb{R}$ be a continuous piecewise linear function with N knots and let $\alpha < 1/\tau$. Then, there exists a the neural network realization of f of width 5 and depth $N+3$, where the Besov semi-norms of the inner layer representations $f_k : \Omega \to \mathbb{R}$, $\Omega := [0,1]^4 \times [-\|f\|_\infty, \|f\|_\infty]$, $2 \le k \le N+2$, are bounded by*

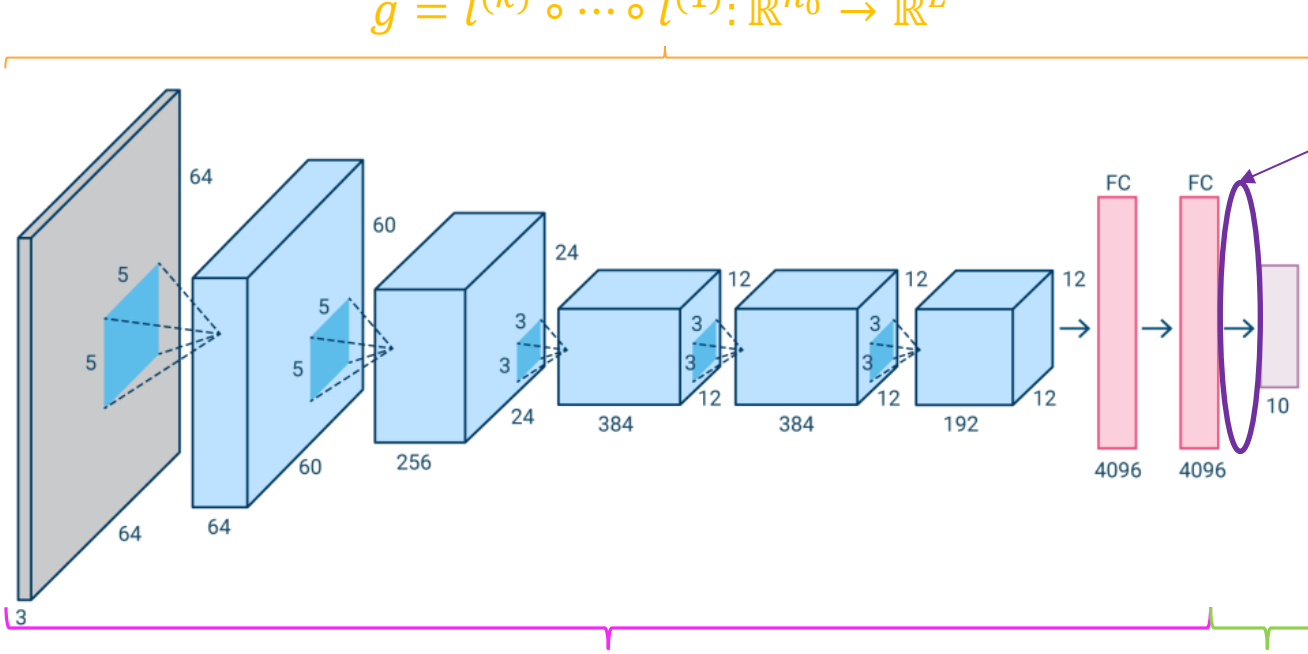
$$|f_k|_{\mathcal{B}^\alpha_\tau(\Omega)} \le c(\alpha, \tau, \|f\|_\infty)(N-k+2)^{1/\tau}. \tag{2.7}$$

Simple geometry of clusters in inner layers : Nearest Class Center Simplification

[1] V. Papyan, X. Han, & D. Donoho, Prevalence of neural collapse during the terminal phase of deep learning training. Proceedings of the National Academy of Sciences of the United States of America, 117 (2022), 24652 - 24663.
[2] I. Ben-Shaul & S. Dekel,Nearest Class-Center Simplification through Intermediate Layers, PMLR 196 (2022).

Setup

$g = l^{(k)} \circ \cdots \circ l^{(1)} : \mathbb{R}^{n_0} \to \mathbb{R}^L$



$h = l^{(k-1)} \circ \cdots \circ l^{(1)} : \mathbb{R}^{n_0} \to \mathbb{R}^{n_{k-1}}$

$l^{(k)} : \mathbb{R}^{n_{k-1}} \to \mathbb{R}^L$

- Let $g^{(j)}(\cdot)$ be the output of the j^{th} layer
- Classification Prediction for input x: $\arg \max_{1 \le \tilde{c} \le C} \langle W_{\tilde{c}}, h(x) \rangle + b_{\tilde{c}}$

Nearest Class Center Simplification

Let $\{(x_i, y_i)\}_{i \in I}$ be a training set. The **train class-means** for a class $1 \leq c \leq L$, at layer j

$$\mu_c^{(j)} \coloneqq \underset{i \in \mathcal{I}_{\text{Train}}, y_i = c}{\text{Avg}} \left\{ g_i^{(j)} \right\}$$

Train within-class covariance at layer j – mean of class covariance estimations (of Normal distributions)

$$\sum\nolimits_W^{(j)} \coloneqq \frac{1}{I} \sum_{c=1}^{L} \underset{i \in I_{\text{Train}}, y_i = c}{\text{Avg}} \left\{ \left(g_i^{(j)} - \mu_c^{(j)} \right) \left(g_i^{(j)} - \mu_c^{(j)} \right)^{\mathsf{T}} \right\}$$

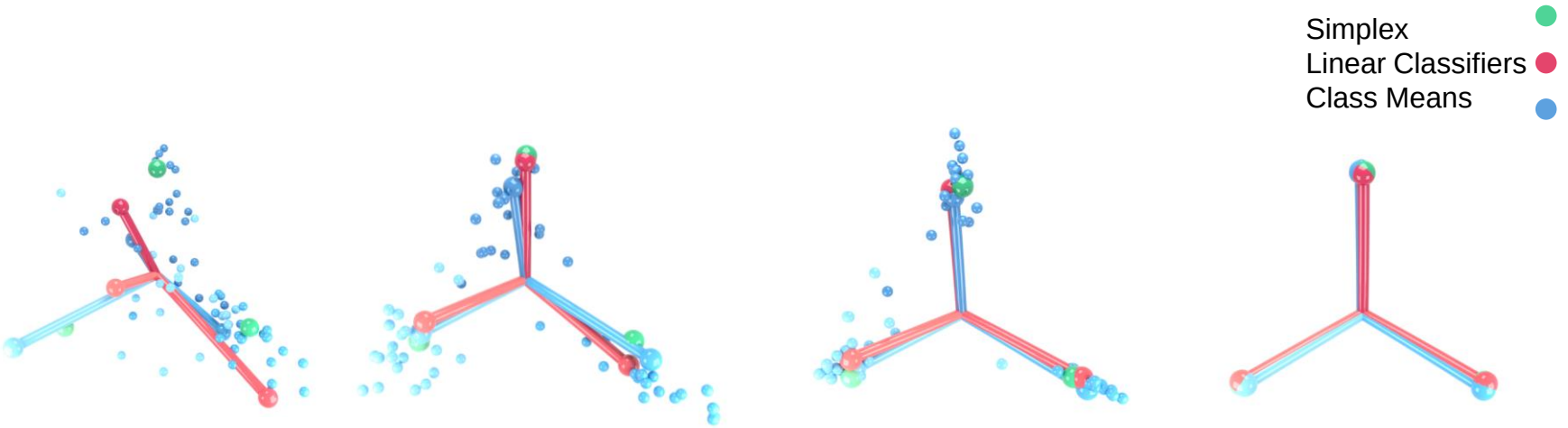
Neural Collapse phenomena

(NC1) Variability Collapse at last layer with training : $\sum_W^{(k-1)} \to 0$

(NC4) Simplification to Nearest Class Center (NCC): Let

$$S \coloneqq \left\{ i \in I_{\text{Train}} \;\middle|\; \arg\max_{1 \le \tilde{c} \le L} g(x_i)_{\tilde{c}} \neq \arg\min_{1 \le \tilde{c} \le L} \left\| g^{(k-1)}(x_i) - \mu_{\tilde{c}}^{(k-1)} \right\|_2 \right\}$$

then $|S| \to 0$, where $|X|$ is the number of elements in finite set X.



Nearest Class Center Simplification in inner layers

NCC Mismatch in intermediate layers:

$$\Lambda^{(j)}_{\text{Train}} \coloneqq \frac{1}{\text{N}_{\text{Train}}} \left| \left\{ \arg\max_{1 \le \tilde{c} \le C} g(x_i)_{\tilde{c}} \neq \arg\min_{1 \le \tilde{c} \le L} \left\| g^{(j)}(x_i) - \mu^{(j)}_{\tilde{c}} \right\|_2 \;\middle|\; i \in \mathcal{I}_{\text{Train}} \right\} \right|$$

$$\Lambda^{(j)}_{\text{Test}} \coloneqq \frac{1}{\text{N}_{\text{Test}}} \left| \left\{ \arg\max_{1 \le \tilde{c} \le C} g(x_i)_{\tilde{c}} \neq \arg\min_{1 \le \tilde{c} \le L} \left\| g^{(j)}(x_i) - \mu^{(j)}_{\tilde{c}} \right\|_2 \;\middle|\; i \in \mathcal{I}_{\text{Test}} \right\} \right|$$

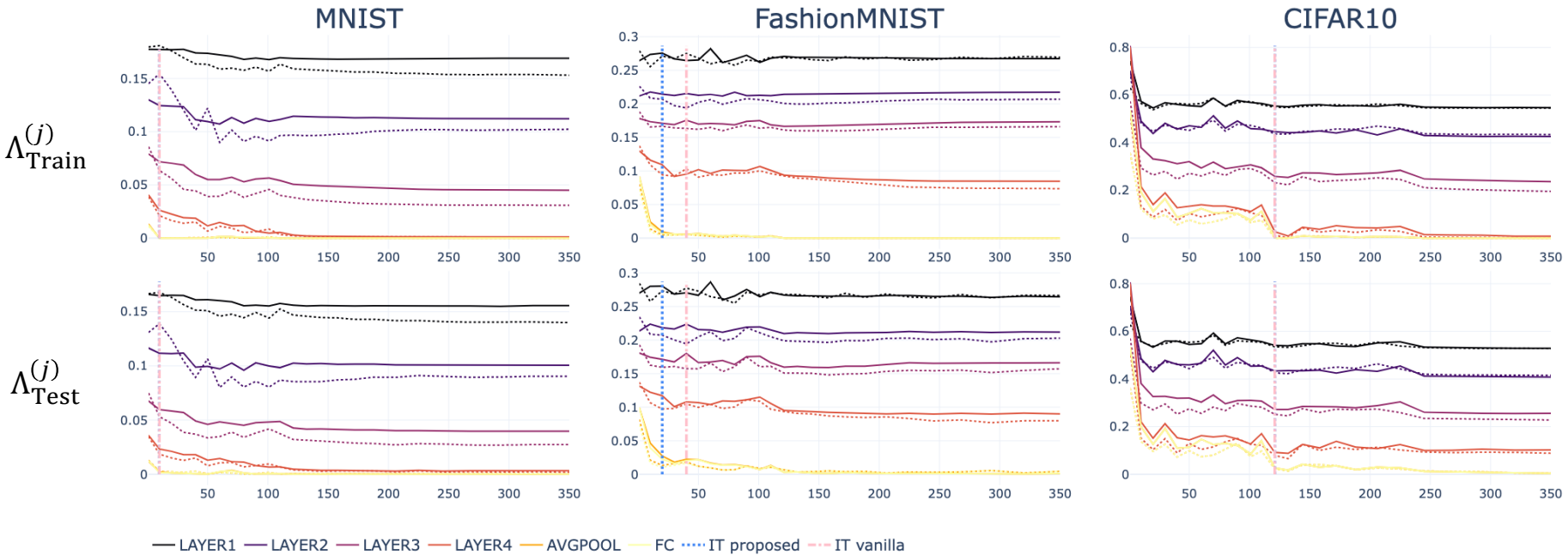
Nearest Class Center in Intermediate Layers

	Vision	Text
Datasets	MNIST, Fashion MNIST, CIFAR10, CIFAR100, STL10	GLUE benchmark: CoLA, RTE, MRPC, SST-2
Architectures	ResNet(18 or 50)	BERT(Uncased) - pretrained
Intermediate Layers	ResNet Blocks	Attention Blocks
Epochs	350	10

Nearest Class Center Simplification in inner layers

Layer Ordering: $\Lambda^{(j)}_{\text{Train}} \geq \Lambda^{(j+1)}_{\text{Train}}$ and $\Lambda^{(j)}_{\text{Test}} \geq \Lambda^{(j+1)}_{\text{Test}}$

NCC Improvement beyond IT: $\Lambda^{(j)}_{\text{Train,IT}} \geq \Lambda^{(j)}_{\text{Train,EOT}}$ and $\Lambda^{(j)}_{\text{Test,IT}} \geq \Lambda^{(j)}_{\text{Test,EOT}}$



NCC with Stochastic Variability-Simplification Loss (SVSL)

Can we encourage clustering in Intermediate Layers?

Stochastic Train class-means: for layer $l^{(j)}$, batch $\mathcal{B}$ and class c

$$\mu_{c,\mathcal{B}}^{(j)} \coloneqq \underset{i \in \mathcal{B}, y_i = c}{\text{Avg}} \left\{ g_i^{(j)} \right\}$$

Stochastic Variability-Simplification Loss (SVSL): Let g be a deepnet, (x_i, y_i), $i \in \mathcal{I}_{\text{Train}}$, $y_i = c$. We also let $\gamma \in \mathbb{N}, 1 \leq \gamma < k$. For batch $\mathcal{B}$ such that $(x_i, y_i) \in \mathcal{B}$, we define the SVSL as an additional loss function:

$$SVSL(i) \coloneqq \sum_{j=\gamma}^{k} \alpha_j \left\| g_i^{(j)}(x_i) - \mu_{c,\mathcal{B}}^{(j)} \right\|_2^2$$

NCC with Stochastic Variability-Simplification Loss (SVSL)

NCC with Stochastic Variability-Simplification Loss (SVSL)

	IT		EOT		Best Test Epoch			
Dataset	Vanilla	SVSL	Vanilla	SVSL	Vanilla	In TPT	SVSL	In TPT
MNIST	**99.37**	99.36	99.61	99.69	99.65	Yes	99.69	Yes
Fashion MNIST	91.78	93.13	93.82	93.88	93.93	Yes	94.03	Yes
STL10	53.41	55.95	54.11	56.65	54.19	Yes	56.94	Yes
CIFAR10	**80.64**	80.56	80.96	81.19	80.96	Yes	81.19	Yes
CIFAR100	52.77	53.28	53.31	54.29	53.79	Yes	54.29	Yes
CoLA	51.59	52.91	53.46	55.54	53.95	No	55.54	Yes
RTE	**58.84**	58.12	55.23	59.57	**61.01**	No	60.28	Yes
MRPC	70.83	74.26	74.26	75.25	75.00	No	76.71	No
SST-2	87.96	88.42	88.42	88.76	**89.22**	No	89.22	Yes