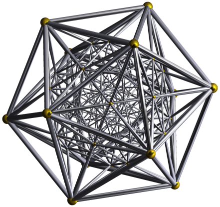
ICASSP 2023 Short Course

Learning Nonlinear and Deep Representations from High-Dimensional Data From Theory to Practice

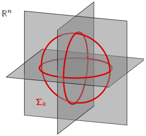
Lecture 7: Deep Representation Learning from the Ground Up

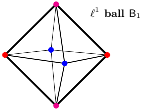
Sam Buchanan, Yi Ma, Qing Qu, Atlas Wang
John Wright, Yuqian Zhang, Zhihui Zhu

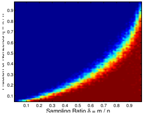
June 9, 2023



Recap: Sparse Recovery







Sparse approximation: **structured** signals, **linear** measurements

$$\boldsymbol{y} = \boldsymbol{A}\boldsymbol{x}_o, \quad \boldsymbol{x}_o \text{ sparse}, \quad \boldsymbol{A} \in \mathbb{R}^{m \times n} \text{ random}$$

with **convex** optimization

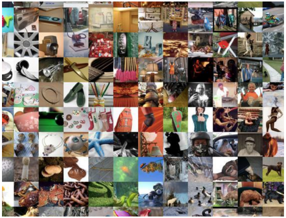
$$\boldsymbol{x}_\star = \arg\min_{\boldsymbol{x} \in \mathbb{R}^n} \frac{1}{2}\|\boldsymbol{y} - \boldsymbol{A}\boldsymbol{x}\|_2^2 + \lambda\|\boldsymbol{x}\|_1$$

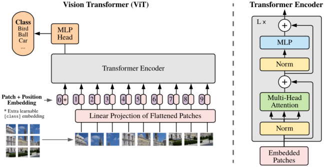
and provable (high probability) guarantees

$$\boldsymbol{x}_\star = \boldsymbol{x}_o \text{ when } \text{measurements} \gtrsim \text{sparsity} \times \log\left(\frac{\text{measurements}}{\text{sparsity}}\right)$$

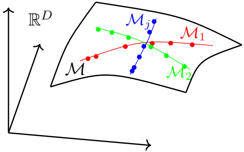
The Deep Learning Era

What role does **low-dimensional structure** play in the **practice** of deep learning? (*understand, improve, design*...)



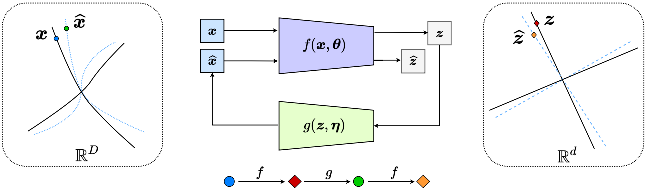


Focus of Today's Lecture: Representation Learning



Goal: seeking a low-dimensional representation $\boldsymbol{Z}$ in $\mathbb{R}^d$ $(d \ll D)$ for the data $\boldsymbol{X}$ on low-dimensional submanifolds such that:

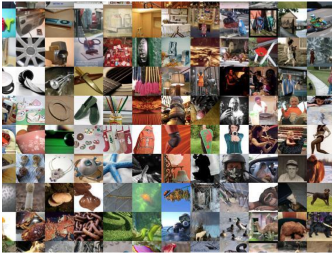
$$\boldsymbol{X} \subset \mathbb{R}^D \xrightarrow{f(\boldsymbol{x},\boldsymbol{\theta})} \boldsymbol{Z} \subset \mathbb{R}^d \xrightarrow{g(\boldsymbol{z},\boldsymbol{\eta})} \hat{\boldsymbol{X}} \approx \boldsymbol{X} \in \mathbb{R}^D.$$



Two subproblems: *identification* and *representation*.

Outline

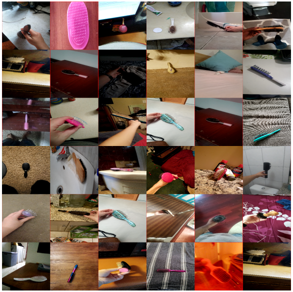
Low-Dimensional Structure in Deep Learning Problems



Appropriate mathematical model for data with low-dimensional structure in the deep learning era: **nonlinear manifolds**?

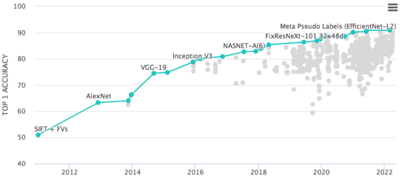
Vignette I: Large-Scale Image Classification

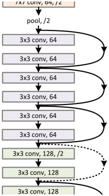
Task: Learn a deep network mapping images → object classes from data.



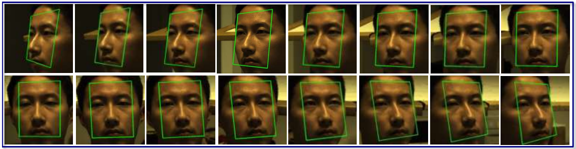
→ {hedgehog, hairbrush}

Massive driver of innovation in the last 10 years (ImageNet, ResNet, ViT...)





Nonlinear Variabilities in Natural Images

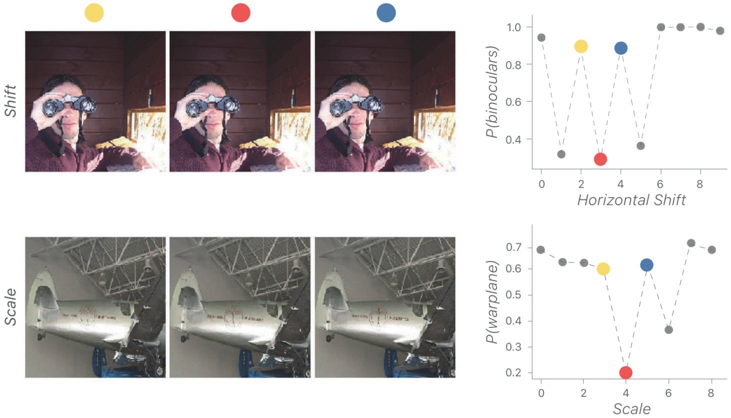


$\Longrightarrow$ **nonlinear, geometric** structure

- 6D for 3D rigid pose; 8D for perspective; 9D for certain illumination...

Limitations of a Purely Data-Driven Approach?

Can fail to learn even simple invariances in the data:

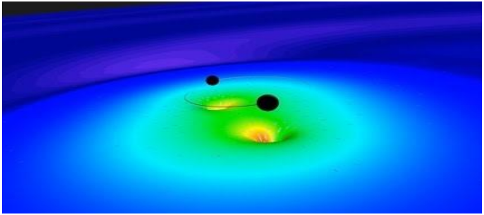


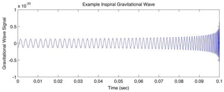
From [Azulay and Weiss, 2019]

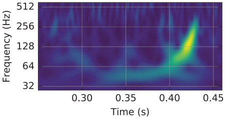
Vignette II: Deep Learning in Scientific Discovery

Gravitational Wave Astronomy

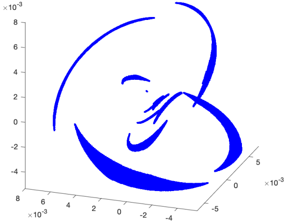
One binary black hole merger:



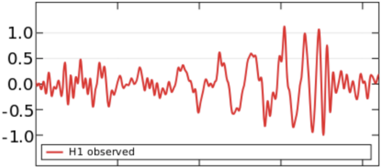


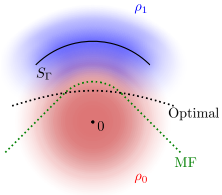


Many mergers
(varying mass M_1, M_2):
$\Longrightarrow$ **low-dim manifold**



Gravitational Wave Astronomy as Parametric Detection

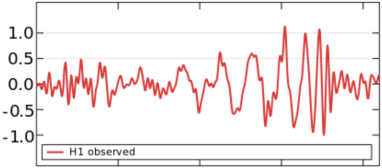


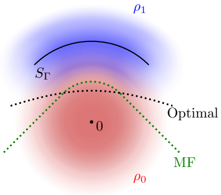


Is observation $\boldsymbol{x} = \boldsymbol{s}_\gamma + \boldsymbol{z}$ or $\boldsymbol{x} = \boldsymbol{z}$?

$\Longrightarrow$ **two (noisy) manifolds!**

Gravitational Wave Astronomy as Parametric Detection



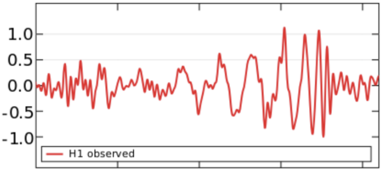


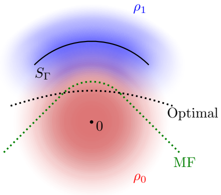
Is observation $\boldsymbol{x} = \boldsymbol{s}_{\boldsymbol{\gamma}} + \boldsymbol{z}$ or $\boldsymbol{x} = \boldsymbol{z}$?

$\implies$ **two (noisy) manifolds!**

Classical approach: template matching $\max_{\boldsymbol{\gamma}} \langle \boldsymbol{a}_{\boldsymbol{\gamma}}, \boldsymbol{x} \rangle > \tau$?

Gravitational Wave Astronomy as Parametric Detection



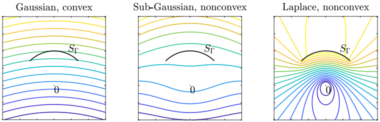


Is observation $\boldsymbol{x} = \boldsymbol{s}_\gamma + \boldsymbol{z}$ or $\boldsymbol{x} = \boldsymbol{z}$?

$\implies$ **two (noisy) manifolds!**

Classical approach: template matching $\max_\gamma \langle \boldsymbol{a}_\gamma, \boldsymbol{x} \rangle > \tau$?

Issues: Optimality? Complexity?
Unknown unknowns? Unknown noise?



Ideally: Combine low-dim structure of Γ with data-driven for statistical structure...

Takeaways from the Examples

Two key takeaways:

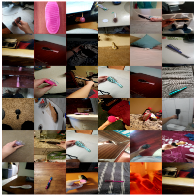
- Data with **nonlinear, geometric structure** pervade successful practical applications of deep learning
- Important practical issues (**robustness/invariance; resource efficiency; performance**) naturally linked to low-dim structure

Takeaways from the Examples

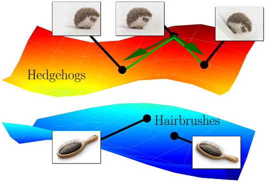
Two key takeaways:

- Data with **nonlinear, geometric structure** pervade successful practical applications of deep learning
- Important practical issues (**robustness/invariance; resource efficiency; performance**) naturally linked to low-dim structure

Next: Understanding mathematically when and why deep learning successfully classifies data with nonlinear geometric structure.



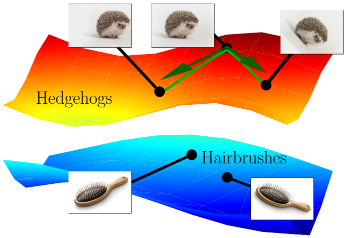
$\Longrightarrow$



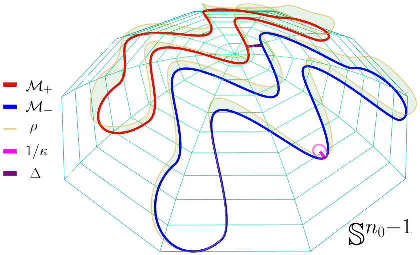
Outline

A Mathematical Model Problem for Deep Learning + Low-Dimensional Structure

Formalizing data with nonlinear geometric structure: Low-dimensional Riemannian submanifolds of high-dimensional space!

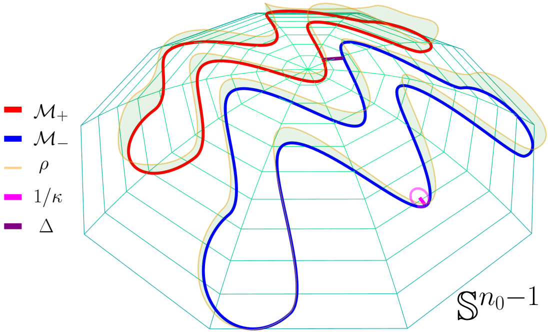


$\Longrightarrow$



The multiple manifold problem: K-way classification of data on d-dimensional Riemannian manifolds in $\mathbb{S}^{n_0-1}$.

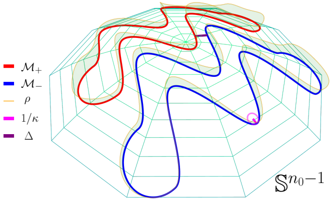
The Two Manifold Problem



Problem. Given N i.i.d. labeled samples $(\boldsymbol{x}_1, y(\boldsymbol{x}_1)), \dots, (\boldsymbol{x}_N, y(\boldsymbol{x}_N))$ from $\mathcal{M} = \mathcal{M}_+ \cup \mathcal{M}_-$, use gradient descent to train a deep network $f_{\boldsymbol{\theta}}$ that *perfectly labels the manifolds*:

$$\mathrm{sign}\left(f_{\boldsymbol{\theta}}(\boldsymbol{x})\right) = y(\boldsymbol{x}) \quad \text{for all} \quad \boldsymbol{x} \in \mathcal{M}.$$

The Two Manifold Problem: Key Aspects



Problem. Given N i.i.d. labeled samples $(\boldsymbol{x}_1, y(\boldsymbol{x}_1)), \dots, (\boldsymbol{x}_N, y(\boldsymbol{x}_N))$ from $\mathcal{M} = \mathcal{M}_+ \cup \mathcal{M}_-$, use gradient descent to train a deep network $f_{\boldsymbol{\theta}}$ that *perfectly labels the manifolds*:

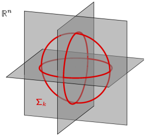
$$\operatorname{sign}(f_{\boldsymbol{\theta}}(\boldsymbol{x})) = y(\boldsymbol{x}) \quad \forall\, \boldsymbol{x} \in \mathcal{M}.$$

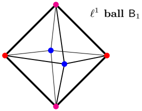
- Binary classification with a deep neural network
- High-dimensional data with (unknown!) low-dimensional structure
- Statistical structure, and asking for "strong" generalization

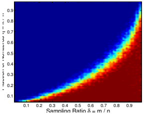
We will focus on the case of one-dimensional manifolds (curves)

What Can We Hope to Understand Here?

Our “barometer”: compressed sensing.





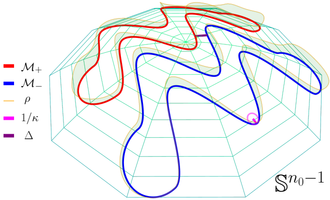


$$\boldsymbol{y} = \boldsymbol{A}\boldsymbol{x}_o; \qquad \boldsymbol{x}_\star = \operatorname*{arg\,min}_{\boldsymbol{x} \in \mathbb{R}^n} \frac{1}{2}\|\boldsymbol{y} - \boldsymbol{A}\boldsymbol{x}\|_2^2 + \lambda\|\boldsymbol{x}\|_1$$

$$\boldsymbol{x}_\star = \boldsymbol{x}_o \text{ when } \text{measurements} \gtrsim \text{sparsity} \times \log\left(\frac{\text{measurements}}{\text{sparsity}}\right)$$

Questions:
What are our ‘measurement resources’ in the two manifold problem?
What are intrinsic structural properties of nonlinear manifold data?

The Two Manifold Problem: Geometric Parameters



Problem. Given N i.i.d. labeled samples $(\boldsymbol{x}_1, y(\boldsymbol{x}_1)), \dots, (\boldsymbol{x}_N, y(\boldsymbol{x}_N))$ from $\mathcal{M} = \mathcal{M}_+ \cup \mathcal{M}_-$, use gradient descent to train a deep network $f_{\boldsymbol{\theta}}$ that *perfectly labels the manifolds*:

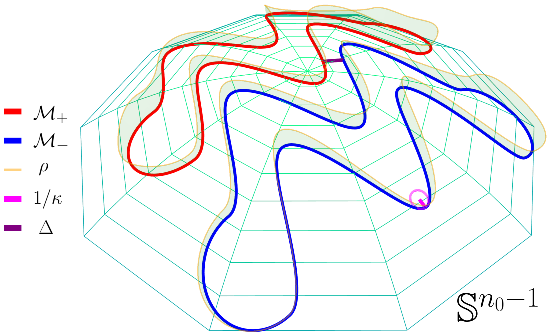
$$\mathrm{sign}\left(f_{\boldsymbol{\theta}}(\boldsymbol{x})\right) = y(\boldsymbol{x}) \quad \forall\, \boldsymbol{x} \in \mathcal{M}.$$

A set of 'sufficient' intrinsic problem difficulty parameters:

- Curvature κ;
- Separation Δ;
- Separation 'frequency' $\maltese$.

Intrinsic Structural Properties I: Separation

Intuitively: How close are the class manifolds?

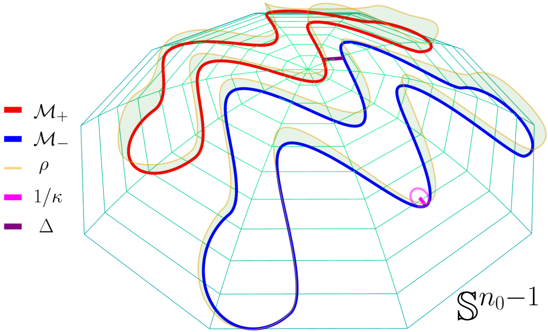


Mathematically:

$$\Delta = \inf_{\boldsymbol{x},\boldsymbol{x}' \in \mathcal{M}} \left\{ d_{\text{extrinsic}}(\boldsymbol{x}, \boldsymbol{x}') \right\}$$

Intrinsic Structural Properties II: Curvature

Intuitively: Local deviation from *flatness* of the manifold.

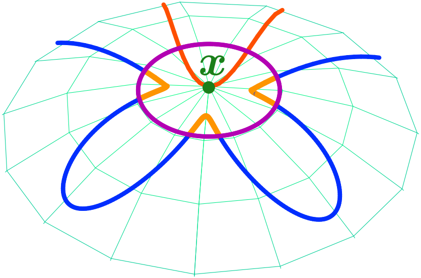


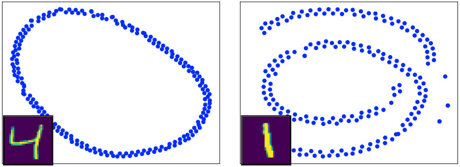
Mathematically:

$$\kappa = \sup_{\boldsymbol{x} \in \mathcal{M}} \left\| \left(\boldsymbol{I} - \frac{\boldsymbol{x}\boldsymbol{x}^*}{\|\boldsymbol{x}\|_2^2} \right) \ddot{\boldsymbol{x}} \right\|_2$$

Intrinsic Structural Properties III: ⌘-Number

Intuitively: How much do the class manifolds loop back on themselves?



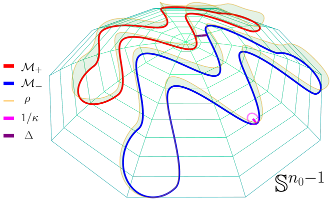


Mathematically:

$$\text{⌘}(\mathcal{M}) = \sup_{\boldsymbol{x} \in \mathcal{M}} N_{\mathcal{M}}\left(\left\{ \boldsymbol{x}' \;\middle|\; \begin{array}{l} d_{\text{intrinsic}}(\boldsymbol{x}, \boldsymbol{x}') > \tau_1 \\ d_{\text{extrinsic}}(\boldsymbol{x}, \boldsymbol{x}') < \tau_2 \end{array} \right\}, \frac{1}{\sqrt{1+\kappa^2}} \right)$$

Here, $N_{\mathcal{M}}(T, \delta)$ is the covering number of $T \subseteq \mathcal{M}$ by δ balls in $d_{\text{intrinsic}}$.

The Two Manifold Problem: Geometric Parameters



Problem. Given N i.i.d. labeled samples $(\boldsymbol{x}_1, y(\boldsymbol{x}_1)), \dots, (\boldsymbol{x}_N, y(\boldsymbol{x}_N))$ from $\mathcal{M} = \mathcal{M}_+ \cup \mathcal{M}_-$, use gradient descent to train a deep network $f_{\boldsymbol{\theta}}$ that *perfectly labels the manifolds*:

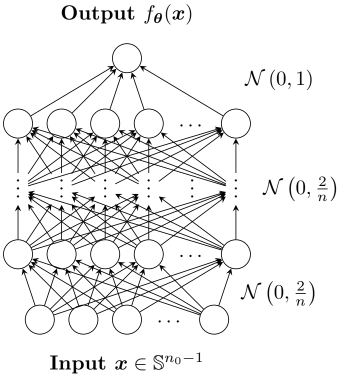
$$\operatorname{sign}(f_{\boldsymbol{\theta}}(\boldsymbol{x})) = y(\boldsymbol{x}) \quad \forall \boldsymbol{x} \in \mathcal{M}.$$

A set of 'sufficient' intrinsic problem difficulty parameters:

- Curvature κ;
- Separation Δ;
- Separation 'frequency' ✤.

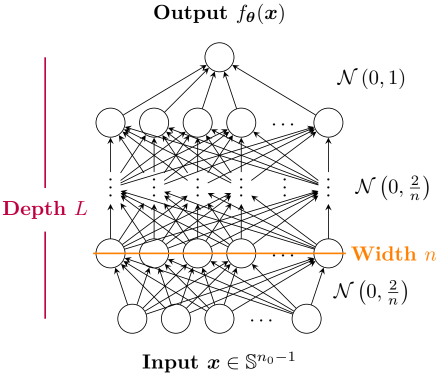
Network Architecture and Training Procedure

- Fully connected with ReLUs
- Gaussian initialization $\boldsymbol{\theta}_0$
- Trained with N i.i.d. samples from measure μ of density ρ



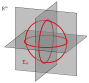
Network Architecture and Training Procedure

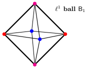
- Fully connected with ReLUs
- Gaussian initialization $\boldsymbol{\theta}_0$
- Trained with N i.i.d. samples from measure μ of density ρ

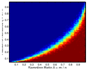


Resource Tradeoffs: From Linear to Nonlinear

The "linear" case (compressed sensing):



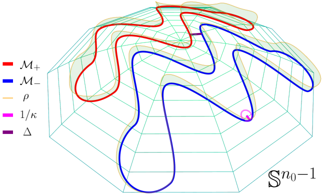




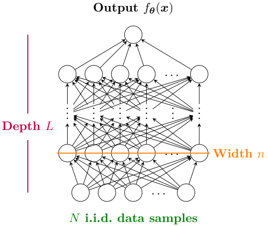
$$\boldsymbol{y} = \boldsymbol{A}\boldsymbol{x}_o; \qquad \boldsymbol{x}_\star = \arg\min_{\boldsymbol{x} \in \mathbb{R}^n} \frac{1}{2}\|\boldsymbol{y} - \boldsymbol{A}\boldsymbol{x}\|_2^2 + \lambda\|\boldsymbol{x}\|_1$$

$$\boldsymbol{x}_\star = \boldsymbol{x}_o \text{ when measurements} \gtrsim \text{sparsity} \times \log\left(\frac{\text{measurements}}{\text{sparsity}}\right)$$

Our current **nonlinear setting**:

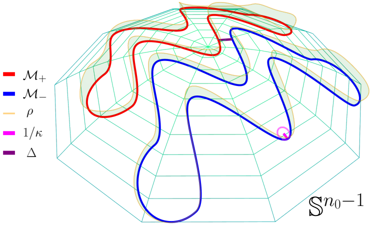


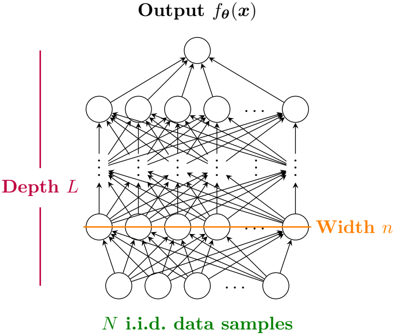
Data structure



Architectural resources

The Two Manifold Problem: Resource Tradeoffs





Theory question: How should we set resources (depth L, width n, samples N) relative to data structure (separation Δ, ⌘; curvature κ; density ρ) so that *gradient descent succeeds*?

Gradient Descent Training

Objective: Square Loss on Training Data

$$\min_{\boldsymbol{\theta}} \varphi(\boldsymbol{\theta}) \equiv \frac{1}{2} \int_{\mathcal{M}} \left(f_{\boldsymbol{\theta}}(\boldsymbol{x}) - y(\boldsymbol{x}) \right)^2 d\mu_N(\boldsymbol{x}).$$

Does gradient descent correctly label the manifolds?

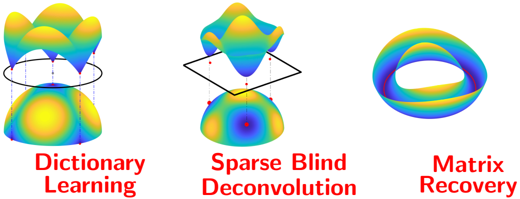
Gradient Descent Training

Objective: Square Loss on Training Data

$$\min_{\boldsymbol{\theta}} \varphi(\boldsymbol{\theta}) \equiv \frac{1}{2} \int_{\mathcal{M}} \left(f_{\boldsymbol{\theta}}(\boldsymbol{x}) - y(\boldsymbol{x})\right)^2 d\mu_N(\boldsymbol{x}).$$

Does gradient descent correctly label the manifolds?

One Approach: Geometry (from symmetry!) in **parameter space**:



See [Gilboa, B., Wright '18], survey [Zhang, Qu, Wright 20] (Lecture 4!)

Gradient Descent Training

Objective: Square Loss on Training Data

$$\min_{\boldsymbol{\theta}} \varphi(\boldsymbol{\theta}) \equiv \frac{1}{2} \int_{\mathcal{M}} \left(f_{\boldsymbol{\theta}}(\boldsymbol{x}) - y(\boldsymbol{x}) \right)^2 d\mu_N(\boldsymbol{x}).$$

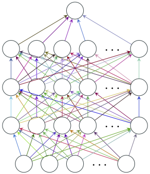
Does gradient descent correctly label the manifolds?

Today's talk: Dynamics in **input-output space**:

Neural Tangent Kernel

$$\Theta(\boldsymbol{x}, \boldsymbol{x}') = \left\langle \frac{\partial f_{\boldsymbol{\theta}}(\boldsymbol{x})}{\partial \boldsymbol{\theta}}, \frac{\partial f_{\boldsymbol{\theta}}(\boldsymbol{x}')}{\partial \boldsymbol{\theta}} \right\rangle$$

Measures ease of independently adjusting $f_{\boldsymbol{\theta}}(\boldsymbol{x})$, $f_{\boldsymbol{\theta}}(\boldsymbol{x}')$



Follows [Jacot et. al. 18], many recent works.

Dynamics of Gradient Descent

Objective: Square Loss on Training Data

$$\min_{\boldsymbol{\theta}} \varphi(\boldsymbol{\theta}) \equiv \frac{1}{2} \int_{\mathcal{M}} (f_{\boldsymbol{\theta}}(\boldsymbol{x}) - y(\boldsymbol{x}))^2 \, d\mu_N(\boldsymbol{x}).$$

Signed error: $\zeta(\boldsymbol{x}) = f_{\boldsymbol{\theta}}(\boldsymbol{x}) - y(\boldsymbol{x})$.

Gradient flow: $\dot{\boldsymbol{\theta}}_t = -\nabla_{\boldsymbol{\theta}} \varphi(\boldsymbol{\theta}_t) = -\int_{\mathcal{M}} \frac{\partial f_{\boldsymbol{\theta}}}{\partial \boldsymbol{\theta}}\Big|_{\boldsymbol{\theta}=\boldsymbol{\theta}_t}(\boldsymbol{x}) \zeta_t(\boldsymbol{x}) d\mu_N(\boldsymbol{x})$.

Dynamics of Gradient Descent

The error evolves according to the NTK:

$$\dot{\zeta}_t(\boldsymbol{x}) \quad = \quad \frac{\partial f_{\boldsymbol{\theta}}(\boldsymbol{x})}{\partial \boldsymbol{\theta}}\bigg|^{*}_{\boldsymbol{\theta}=\boldsymbol{\theta}_t} \dot{\boldsymbol{\theta}}_t$$

Dynamics of Gradient Descent

The error evolves according to the NTK:

$$
\begin{aligned}
\dot{\zeta}_t(\boldsymbol{x}) &= \left.\frac{\partial f_{\boldsymbol{\theta}}(\boldsymbol{x})}{\partial \boldsymbol{\theta}}\right|^*_{\boldsymbol{\theta}=\boldsymbol{\theta}_t} \dot{\boldsymbol{\theta}}_t \\
&= -\left.\frac{\partial f_{\boldsymbol{\theta}}(\boldsymbol{x})}{\partial \boldsymbol{\theta}}\right|^*_{\boldsymbol{\theta}=\boldsymbol{\theta}_t} \int_{\mathcal{M}} \left.\frac{\partial f_{\boldsymbol{\theta}}(\boldsymbol{x}')}{\partial \boldsymbol{\theta}}\right|_{\boldsymbol{\theta}=\boldsymbol{\theta}_t} \zeta_t(\boldsymbol{x}') d\mu_N(\boldsymbol{x}')
\end{aligned}
$$

Dynamics of Gradient Descent

The error evolves according to the NTK:

$$
\begin{aligned}
\dot{\zeta}_t(\boldsymbol{x}) &= \frac{\partial f_{\boldsymbol{\theta}}(\boldsymbol{x})}{\partial \boldsymbol{\theta}}\bigg|_{\boldsymbol{\theta}=\boldsymbol{\theta}_t}^{*} \dot{\boldsymbol{\theta}}_t \\
&= -\frac{\partial f_{\boldsymbol{\theta}}(\boldsymbol{x})}{\partial \boldsymbol{\theta}}\bigg|_{\boldsymbol{\theta}=\boldsymbol{\theta}_t}^{*} \int_{\mathcal{M}} \frac{\partial f_{\boldsymbol{\theta}}(\boldsymbol{x}')}{\partial \boldsymbol{\theta}}\bigg|_{\boldsymbol{\theta}=\boldsymbol{\theta}_t} \zeta_t(\boldsymbol{x}') d\mu_N(\boldsymbol{x}') \\
&= -\int_{\mathcal{M}} \left\langle \frac{\partial f_{\boldsymbol{\theta}}(\boldsymbol{x})}{\partial \boldsymbol{\theta}}\bigg|_{\boldsymbol{\theta}=\boldsymbol{\theta}_t}, \frac{\partial f_{\boldsymbol{\theta}}(\boldsymbol{x}')}{\partial \boldsymbol{\theta}}\bigg|_{\boldsymbol{\theta}=\boldsymbol{\theta}_t} \right\rangle \zeta_t(\boldsymbol{x}') d\mu_N(\boldsymbol{x}')
\end{aligned}
$$

Dynamics of Gradient Descent

The error evolves according to the NTK:

$$\begin{aligned}
\dot{\zeta}_t(\boldsymbol{x}) &= \frac{\partial f_{\boldsymbol{\theta}}(\boldsymbol{x})}{\partial \boldsymbol{\theta}}\bigg|_{\boldsymbol{\theta}=\boldsymbol{\theta}_t}^{*} \dot{\boldsymbol{\theta}}_t \\
&= -\frac{\partial f_{\boldsymbol{\theta}}(\boldsymbol{x})}{\partial \boldsymbol{\theta}}\bigg|_{\boldsymbol{\theta}=\boldsymbol{\theta}_t}^{*} \int_{\mathcal{M}} \frac{\partial f_{\boldsymbol{\theta}}(\boldsymbol{x}')}{\partial \boldsymbol{\theta}}\bigg|_{\boldsymbol{\theta}=\boldsymbol{\theta}_t} \zeta_t(\boldsymbol{x}') d\mu_N(\boldsymbol{x}') \\
&= -\int_{\mathcal{M}} \left\langle \frac{\partial f_{\boldsymbol{\theta}}(\boldsymbol{x})}{\partial \boldsymbol{\theta}}\bigg|_{\boldsymbol{\theta}=\boldsymbol{\theta}_t}, \frac{\partial f_{\boldsymbol{\theta}}(\boldsymbol{x}')}{\partial \boldsymbol{\theta}}\bigg|_{\boldsymbol{\theta}=\boldsymbol{\theta}_t} \right\rangle \zeta_t(\boldsymbol{x}') d\mu_N(\boldsymbol{x}')
\end{aligned}$$

Dynamics of Gradient Descent

The error evolves according to the NTK:

$$\begin{aligned}
\dot{\zeta}_t(\boldsymbol{x}) &= \left.\frac{\partial f_{\boldsymbol{\theta}}(\boldsymbol{x})}{\partial \boldsymbol{\theta}}\right|^{*}_{\boldsymbol{\theta}=\boldsymbol{\theta}_t} \dot{\boldsymbol{\theta}}_t \\
&= -\left.\frac{\partial f_{\boldsymbol{\theta}}(\boldsymbol{x})}{\partial \boldsymbol{\theta}}\right|^{*}_{\boldsymbol{\theta}=\boldsymbol{\theta}_t} \int_{\mathcal{M}} \left.\frac{\partial f_{\boldsymbol{\theta}}(\boldsymbol{x}')}{\partial \boldsymbol{\theta}}\right|_{\boldsymbol{\theta}=\boldsymbol{\theta}_t} \zeta_t(\boldsymbol{x}') d\mu_N(\boldsymbol{x}') \\
&= -\int_{\mathcal{M}} \left\langle \left.\frac{\partial f_{\boldsymbol{\theta}}(\boldsymbol{x})}{\partial \boldsymbol{\theta}}\right|_{\boldsymbol{\theta}=\boldsymbol{\theta}_t}, \left.\frac{\partial f_{\boldsymbol{\theta}}(\boldsymbol{x}')}{\partial \boldsymbol{\theta}}\right|_{\boldsymbol{\theta}=\boldsymbol{\theta}_t} \right\rangle \zeta_t(\boldsymbol{x}') d\mu_N(\boldsymbol{x}') \\
&= -\int_{\mathcal{M}} \Theta_t(\boldsymbol{x}, \boldsymbol{x}') \zeta_t(\boldsymbol{x}') d\mu_N(\boldsymbol{x}') \\
&= -\boldsymbol{\Theta}_t[\zeta_t](\boldsymbol{x}).
\end{aligned}$$

Dynamics of Gradient Descent ("NTK Regime")

When width and number of data samples are large, we have (whp)

$$\sup_t \|\boldsymbol{\Theta}_t - \boldsymbol{\Theta}\|_{L^2 \to L^2} = o_{\text{width}}(1)$$

throughout training.

$\implies$ *LTI dynamics*

$$\dot{\zeta}_t = -\boldsymbol{\Theta}[\zeta_t]$$

$\implies$ **Fast decay** if ζ_t is aligned with lead eigenvectors of $\boldsymbol{\Theta}$!

Implicit Error-NTK Alignment with Certificates

Challenge: For nonlinear $\mathcal{M}$, eigenvectors of $\boldsymbol{\Theta}$ are intractable!

Definition. $g : \mathcal{M} \to \mathbb{R}$ is called a *certificate* if for all $\boldsymbol{x} \in \mathcal{M}$

$$f_{\boldsymbol{\theta}_0}(\boldsymbol{x}) - y(\boldsymbol{x}) \underset{\text{square}}{\overset{\text{mean}}{\approx}} \int_{\mathcal{M}} \Theta(\boldsymbol{x}, \boldsymbol{x}') g(\boldsymbol{x}') \, \mathrm{d}\mu(\boldsymbol{x}')$$

and $\int_{\mathcal{M}} (g(\boldsymbol{x}'))^2 \, \mathrm{d}\mu(\boldsymbol{x}')$ is small.

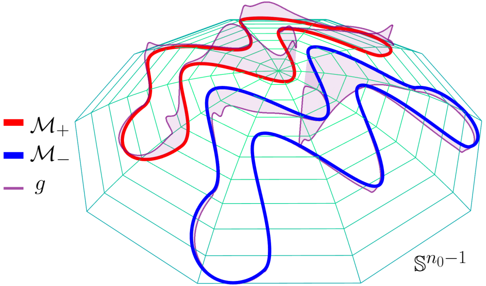
Implicit Error-NTK Alignment with Certificates

Challenge: For nonlinear $\mathcal{M}$, eigenvectors of $\boldsymbol{\Theta}$ are intractable!

Definition. $g : \mathcal{M} \to \mathbb{R}$ is called a *certificate* if for all $\boldsymbol{x} \in \mathcal{M}$

$$f_{\boldsymbol{\theta}_0}(\boldsymbol{x}) - y(\boldsymbol{x}) \underset{\text{square}}{\overset{\text{mean}}{\approx}} \int_{\mathcal{M}} \Theta(\boldsymbol{x}, \boldsymbol{x}') g(\boldsymbol{x}') \, \mathrm{d}\mu(\boldsymbol{x}')$$

and $\int_{\mathcal{M}} (g(\boldsymbol{x}'))^2 \, \mathrm{d}\mu(\boldsymbol{x}')$ is small.



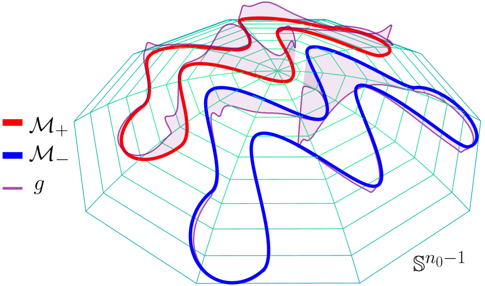
Implicit Error-NTK Alignment with Certificates

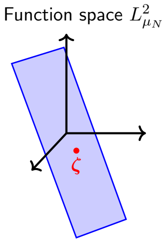
Challenge: For nonlinear $\mathcal{M}$, eigenvectors of $\boldsymbol{\Theta}$ are intractable!

Definition. $g : \mathcal{M} \to \mathbb{R}$ is called a *certificate* if for all $\boldsymbol{x} \in \mathcal{M}$

$$f_{\boldsymbol{\theta}_0}(\boldsymbol{x}) - y(\boldsymbol{x}) \overset{\text{mean}}{\underset{\text{square}}{\approx}} \int_{\mathcal{M}} \Theta(\boldsymbol{x}, \boldsymbol{x}') g(\boldsymbol{x}') \, \mathrm{d}\mu(\boldsymbol{x}')$$

and $\int_{\mathcal{M}} (g(\boldsymbol{x}'))^2 \, \mathrm{d}\mu(\boldsymbol{x}')$ is small.





Error ζ near stable range of *random operator* $\boldsymbol{\Theta}$

Implicit Error-NTK Alignment with Certificates

Challenge: For nonlinear $\mathcal{M}$, eigenvectors of $\boldsymbol{\Theta}$ are intractable!

Definition. $g : \mathcal{M} \to \mathbb{R}$ is called a *certificate* if for all $\boldsymbol{x} \in \mathcal{M}$

$$f_{\boldsymbol{\theta}_0}(\boldsymbol{x}) - y(\boldsymbol{x}) \underset{\text{square}}{\overset{\text{mean}}{\approx}} \int_{\mathcal{M}} \Theta(\boldsymbol{x}, \boldsymbol{x}')g(\boldsymbol{x}')\,\mathrm{d}\mu(\boldsymbol{x}')$$

and $\int_{\mathcal{M}} (g(\boldsymbol{x}'))^2\,\mathrm{d}\mu(\boldsymbol{x}')$ is small.

Lemma. (informal) If a certificate g exists for $\mathcal{M}$, then

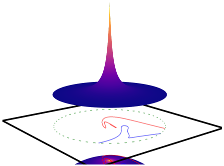
$$\|\zeta_t\|_{L^2_\mu} \lesssim \frac{L \log L}{t}.$$

Roles of Width, Depth, and Data

$$\dot{\zeta}_t = -\boldsymbol{\Theta}[\zeta_t]$$

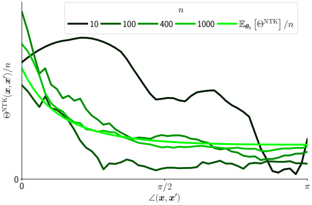
Questions:
How do width, depth, and samples affect $\boldsymbol{\Theta}$?
How does $\boldsymbol{\Theta}$ depend on the geometry of the data?

Depth L: **fitting resource**



$\frac{1}{L}\Theta(\boldsymbol{e}_1, \boldsymbol{x}')$, $L = 125$

Width n: **statistical resource**

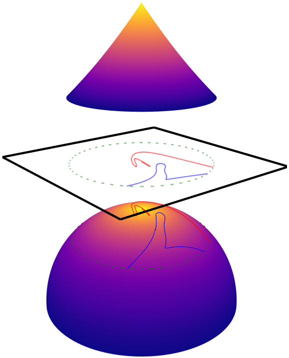


Resource Tradeoffs I: Depth as a Fitting Resource

Key insights:

1. Θ decays with angle.
2. Faster decay as depth increases.

$\implies$ Set depth based on geometry!



$\frac{1}{L}\Theta(\boldsymbol{e}_1, \boldsymbol{x}')$, $L = 5$

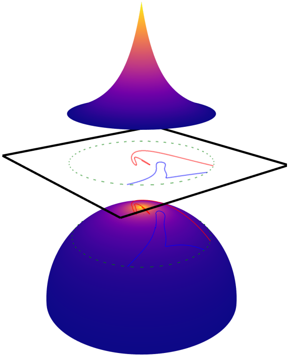
Deeper networks fit more complicated geometries.

Resource Tradeoffs I: Depth as a Fitting Resource

Key insights:

1. Θ decays with angle.
2. Faster decay as depth increases.

$\implies$ Set depth based on geometry!



$\frac{1}{L}\Theta(\boldsymbol{e}_1, \boldsymbol{x}')$, $L = 25$

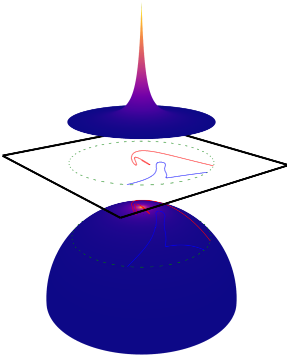
Deeper networks fit more complicated geometries.

Resource Tradeoffs I: Depth as a Fitting Resource

Key insights:

1. Θ decays with angle.
2. Faster decay as depth increases.

$\Longrightarrow$ Set depth based on geometry!



$\frac{1}{L}\Theta(\boldsymbol{e}_1, \boldsymbol{x}')$, $L = 125$

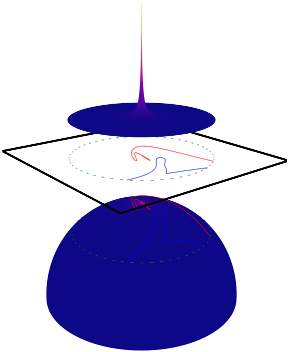
Deeper networks fit more complicated geometries.

Resource Tradeoffs I: Depth as a Fitting Resource

Key insights:

1. Θ decays with angle.
2. Faster decay as depth increases.

$\implies$ Set depth based on geometry!

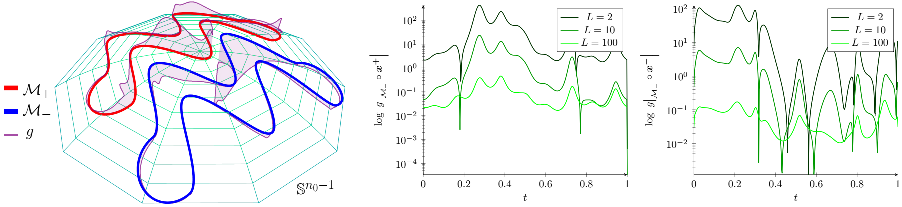


$\frac{1}{L}\Theta(\boldsymbol{e}_1, \boldsymbol{x}')$, $L = 625$

Deeper networks fit more complicated geometries.

Resource Tradeoffs I: Certificates from Depth

Numerical experiment:

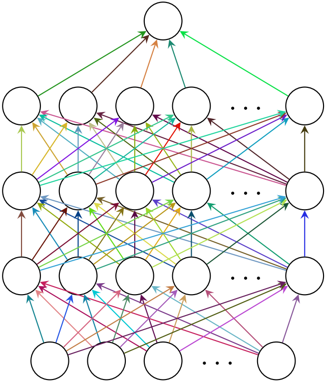


Depth as a fitting resource: Larger depth L leads to a sharper kernel Θ and a smaller certificate g

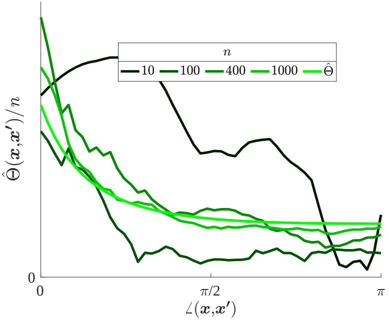
$\implies$ Easier fitting!

Resource Tradeoffs II: Width as a Statistical Resource

Output $f_{\boldsymbol{\theta}}(\boldsymbol{x})$



Input $\boldsymbol{x} \in \mathbb{S}^{n_0 - 1}$



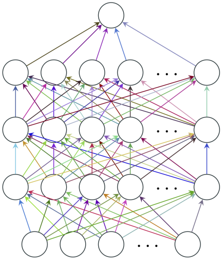
As width increases, $\Theta(\boldsymbol{x}, \boldsymbol{x}')$ concentrates about $\mathbb{E}_{\text{init weights}}[\Theta(\boldsymbol{x}, \boldsymbol{x}')]$

Resource Tradeoffs II: Width as a Statistical Resource

Proposition. Suppose that $n > L\mathrm{polylog}(Ln_0)$. Then (whp)

$$\left| \Theta(\boldsymbol{x}, \boldsymbol{x}') - \frac{n}{2} \sum_{\ell} \cos(\varphi^{\ell} \nu) \prod_{\ell'=\ell}^{L-1} \left(1 - \frac{\varphi^{\ell'} \nu}{\pi}\right) \right|$$

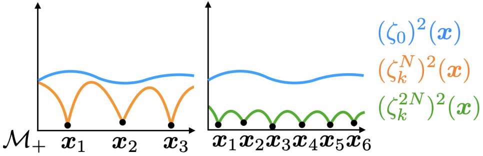
is small (simultaneously) for all $(\boldsymbol{x}, \boldsymbol{x}') \in \mathcal{M} \times \mathcal{M}$.

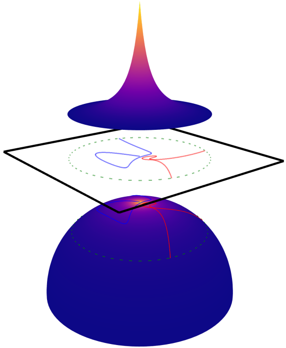


$\Rightarrow$ **set width n based on depth L**

and implicitly based on κ, Δ

Resource Tradeoffs III: Data as a Statistical Resource





Depth $L = 50$

⇒ Sample complexity N is dictated by kernel "aperture", which depends on geometry (κ, Δ) via L

End-to-End Generalization Guarantee

Theorem (very informal): For sufficiently regular one-dimensional manifolds and ReLU networks, when

$$\text{depth} \geq \text{geometry}, \ \text{width} \geq \mathrm{poly}(\text{depth}), \ \text{data} \geq \mathrm{poly}(\text{depth}),$$

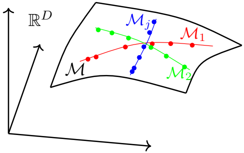
randomly-initialized small-stepping gradient descent perfectly classifies the two manifolds!

Upshot:

- We understand the role each resource plays in solving the classification problem.
- We understand how intrinsic geometric properties of the data drive these resource requirements.

Outline

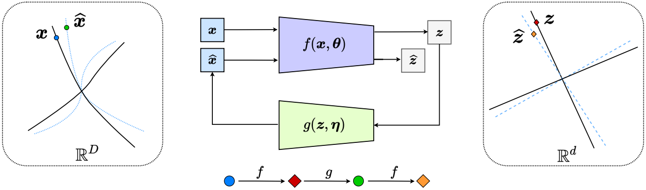
Ideal Representation as Autoencoding + Linearization



Goal: seeking a low-dimensional representation $\boldsymbol{Z}$ in $\mathbb{R}^d$ $(d \ll D)$ for the data $\boldsymbol{X}$ on low-dimensional submanifolds such that:

$$\boldsymbol{X} \subset \mathbb{R}^D \xrightarrow{f(\boldsymbol{x},\boldsymbol{\theta})} \boldsymbol{Z} \subset \mathbb{R}^d \xrightarrow{g(\boldsymbol{z},\boldsymbol{\eta})} \hat{\boldsymbol{X}} \approx \boldsymbol{X} \in \mathbb{R}^D.$$

We moreover want the representation $\boldsymbol{Z}$ to consist of **certain canonical geometric configurations**, say **subspaces**:



Focus here on $\mathcal{M}$ = one manifold (we understand identification!)

Standard Approaches to Linearize a Manifold, and Pitfalls

1. Embed training data in $\mathbb{R}^d$ by gluing local isometries (*manifold learning*)

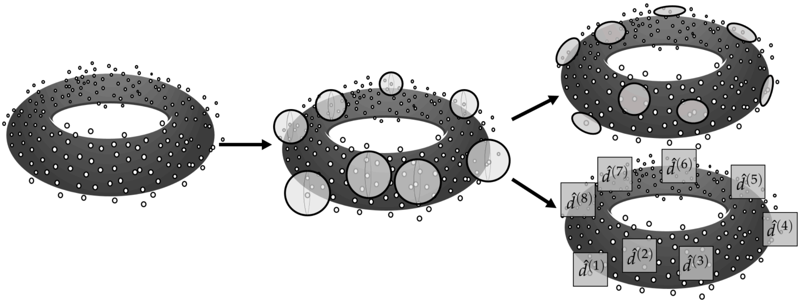


Figure credit: Lim, Oberhauser, and Nanda 2022

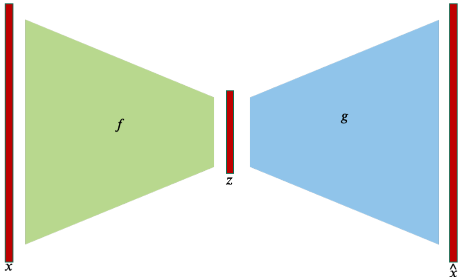
+ Provably correct with enough data [Lim et al. 2022], one-one mapping
− No standard generalization to test data without retraining, difficult to scale to high-dimensional datasets

Standard Approaches to Linearize a Manifold, and Pitfalls

2. Parameterize f, g with deep networks, regularized reconstruction training:

$$\min_{f,g} \mathbb{E}_{\boldsymbol{X}} \left[\| \boldsymbol{X} - g\left(f(\boldsymbol{X})\right) \|_{\mathrm{F}}^2 \right] + R(f, g)$$

Encompasses most deep net autoencoders (variational, denoising, VQGAN-type)

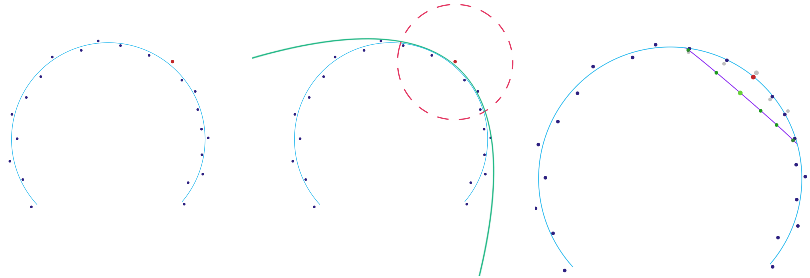


+ Truly learns a representation of the distribution, one-one mapping with proper regularization
− Black-box, no mathematical guarantees in regimes of interest

Manifold Flattening with Second-Order Information

Recent approach to "have it all": [Psenka, Pai, Raman, Sastry, Ma 2023]

- Ask for **flattening**, rather than *isometry*
- Use second-order local information (better **efficiency**)
- Gluing as a **multi-layer**, **invertible** process!

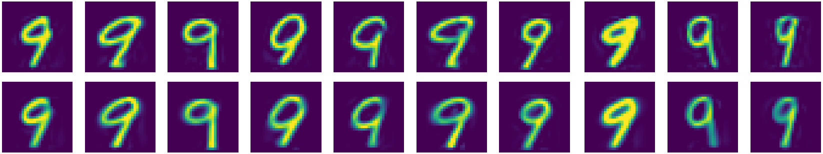


Visualization of Psenka et al.'s Method

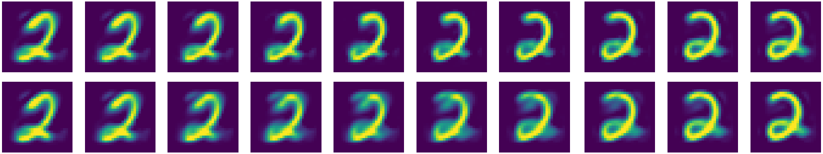
figures/flatnet-music-video.mp4

Scaling Psenka et al.'s Method to MNIST

$$D = 784,\ d \approx 12$$



Reconstruction of 9s

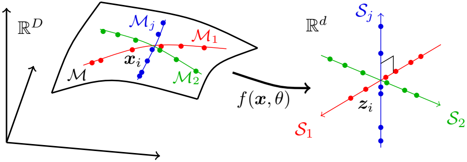


Latent interpolation of two 2s

Limitations of Perfect Manifold Linearization (+ Relaxation)

Still hard to scale this to modern high-dim datasets (ImageNet, LAION-5B)

Practically-motivated solution: give up on one-one representation
$\implies$ distribution learning



one-one: $\boldsymbol{X} \subset \mathbb{R}^D \xrightarrow{f(\boldsymbol{x},\boldsymbol{\theta})} \boldsymbol{Z} \subset \mathbb{R}^d \xrightarrow{g(\boldsymbol{z},\boldsymbol{\eta})} \hat{\boldsymbol{X}} \approx \boldsymbol{X}$

distributional: $\boldsymbol{X} \subset \mathbb{R}^D \xrightarrow{f(\boldsymbol{x},\boldsymbol{\theta})} \boldsymbol{Z} \subset \mathbb{R}^d \xrightarrow{g(\boldsymbol{z},\boldsymbol{\eta})} \mathrm{Law}(\hat{\boldsymbol{X}}) \approx \mathrm{Law}(\boldsymbol{X})$

Spectacular Success of Distribution Learning: Diffusion Models

Diffusion models let us *generate new samples of our data* $\boldsymbol{X}$...

figures/diffusion-iterations-lastlong.m

...by *incrementally* transforming $\mathrm{Law}(\boldsymbol{X})$ to $\mathrm{Law}(\boldsymbol{Z}) = \mathcal{N}(\mathbf{0}, \boldsymbol{I}_D)$ and back

Diffusion Models: Conceptual Idea

Conceptual idea: Transform data into noise, and back!

figures/curve-diffusion-sin.r figures/curve-diffusion-circl

Outline for understanding diffusion models: (next slides)

- *How do we transform data into noise?*
- *How do we transform noise back into data?*
- *How do we actually implement it?* (finite samples and efficient computation)

Math of Diffusion Models: Data to Noise (SDEs)

Transform data into noise with the "Ornstein-Uhlenbeck process":

$$\begin{aligned} \mathrm{d}\boldsymbol{x}_t &= -\boldsymbol{x}_t \,\mathrm{d}t + \sqrt{2}\,\mathrm{d}\boldsymbol{w}_t \\ \boldsymbol{x}_0 &= \boldsymbol{x} \end{aligned}$$

This is a "stochastic differential equation".

???

Math of Diffusion Models: Data to Noise (SDEs)

Transform data into noise with the "Ornstein-Uhlenbeck process":

$$\mathrm{d}\boldsymbol{x}_t = -\boldsymbol{x}_t \,\mathrm{d}t + \sqrt{2}\,\mathrm{d}\boldsymbol{w}_t$$
$$\boldsymbol{x}_0 = \boldsymbol{x}$$

This is a "stochastic differential equation".

Formal intuition: this notation means

$$\boldsymbol{x}_t = -\int_0^t \boldsymbol{x}_s \,\mathrm{d}s + \sqrt{2}\int_0^t \mathrm{d}\boldsymbol{w}_s, \quad t \geq 0.$$

The last integral is like a sum of gaussians, and $\int_0^t \mathrm{d}\boldsymbol{w}_s = \boldsymbol{w}_t$. Thus

$$\boldsymbol{x}_t = e^{-t}\boldsymbol{x}_0 + \sqrt{2}e^{-t}\int_0^t e^s \,\mathrm{d}\boldsymbol{w}_s.$$

Now term two is like a *weighted* sum of gaussians! In particular

$$\mathrm{Law}(\boldsymbol{x}_t) = \mathcal{N}\left(e^{-t}\boldsymbol{x}, (1 - e^{-2t})\boldsymbol{I}\right).$$

Closed-Form OU Evolution

For the OU process:

$$\mathrm{Law}(\boldsymbol{x}_t) = \mathcal{N}\left(e^{-t}\boldsymbol{x}, (1 - e^{-2t})\boldsymbol{I}\right)$$

If $\boldsymbol{x}$ is a random variable, then

$$\mathrm{Law}(\boldsymbol{x}_t) = \underbrace{\varphi_{1-e^{-2t}}}_{\text{gaussian density}} * \mathrm{Law}(e^{-t}\boldsymbol{x})$$

figures/curve-diffusion-sin.r

figures/curve-diffusion-circl

$\implies$ x_t **has a density** ρ_t**! Linear convergence to normality!**

Math of Diffusion Models: Noise to Data

If we stop the process at time $T > 0$, $\boldsymbol{x}_t^{\leftarrow} = \boldsymbol{x}_{T-t}$ also satisfies a SDE:

$$\mathrm{d}\boldsymbol{x}_t^{\leftarrow} = (\boldsymbol{x}_t^{\leftarrow} + 2\nabla \log \rho_{T-t}(\boldsymbol{x}_t^{\leftarrow}))\,\mathrm{d}t + \sqrt{2}\,\mathrm{d}\boldsymbol{w}_t$$

figures/curve-diffusion-sin-r

figures/curve-diffusion-sin-r

$\Longrightarrow$ **discretize, and generate new samples from data!**

Math of Diffusion Models: Actually Implementing It

One (big) problem: **We don't know $\mathrm{Law}(\boldsymbol{x})$!**

figures/diffusion-iterations-lastlong.m

E.g. $\mathrm{Law}(\boldsymbol{x}) = \{$distribution of natural images$\}$...

Math of Diffusion Models: Sampling with Score Matching

Idea: sampling follows the process

$$\mathrm{d}\boldsymbol{x}_t^{\leftarrow} = (\boldsymbol{x}_t^{\leftarrow} + 2\nabla \log \rho_{T-t}(\boldsymbol{x}_t^{\leftarrow}))\,\mathrm{d}t + \sqrt{2}\,\mathrm{d}\boldsymbol{w}_t \tag{1}$$

Tweedie's formula (1956): *Let* $\boldsymbol{y} = e^{-t}\boldsymbol{x} + \mathcal{N}(\boldsymbol{0}, (1 - e^{-2t})\boldsymbol{I})$. *Then*

$$e^{-t}\mathbb{E}[\boldsymbol{x} \mid \boldsymbol{y}] = \boldsymbol{y} + (1 - e^{-2t})\nabla \log \rho_t(\boldsymbol{y}).$$

$\implies$ **equivalence between estimation (denoising) and score matching!**

Many authors ([Hyvärinen 2005], [Vincent 2011], [Song & Ermon 2019], [Ho, Jain, & Abbeel 2020]):

Train a neural network to perform estimation

$$\min_{F:\mathbb{R}^D \times \mathbb{R} \to \mathbb{R}^D} \mathop{\mathbb{E}}_{\boldsymbol{x},\boldsymbol{g}\sim\mathcal{N}(\boldsymbol{0},\boldsymbol{I})} \left[\left\| F\left(e^{-t}\boldsymbol{x} + (1 - e^{-2t})^{1/2}\boldsymbol{g}; t\right) + \frac{1}{(1 - e^{-2t})^{1/2}}\boldsymbol{g} \right\|_2^2 \right]$$

then plug F into Eq. (1) to sample!

Conceptual Pipeline for Diffusion Models

- Train score estimation network F with i.i.d. samples $\boldsymbol{x}_i$, $\boldsymbol{g}_{ij}$:

$$\min_F \sum_{i,j,t} \left\| F\left(e^{-t}\boldsymbol{x}_i + (1-e^{-2t})^{1/2}\boldsymbol{g}_{ij}; t\right) + \frac{1}{(1-e^{-2t})^{1/2}}\boldsymbol{g}_{ij} \right\|_2^2$$

- Sample as though F is the true score:

$$\mathrm{d}\boldsymbol{x}_t^{\leftarrow} = (\boldsymbol{x}_t^{\leftarrow} + 2F(\boldsymbol{x}_t^{\leftarrow}; T-t))\,\mathrm{d}t + \sqrt{2}\,\mathrm{d}\boldsymbol{w}_t$$

figures/curve-diffusion-circl

figures/curve-diffusion-circl

Pitfalls of Diffusion Models

Despite impressive performance and excitement, critical issues remain

figures/diffusion-iterations-lastlong.m

1. Good learning of $\nabla \log \rho_t \iff$ **network** F **has proper architecture**

Pitfalls of Diffusion Models

Despite impressive performance and excitement, critical issues remain

figures/diffusion-iterations-lastlong.m

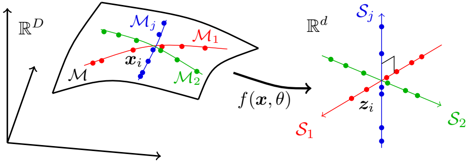
2. **Black box learned representation (no identification/control)**

Outline

Identification/Representation of High-Dim Structured Data

Focus on one half of our goal:

Given samples
$\boldsymbol{X} = [\boldsymbol{x}_1, \ldots, \boldsymbol{x}_m] \subset \cup_{j=1}^{k} \mathcal{M}_j$,
seek a good representation
$\boldsymbol{Z} = [\boldsymbol{z}_1, \ldots, \boldsymbol{z}_m] \subset \mathbb{R}^d$
through a continuous mapping:
$f(\boldsymbol{x}, \boldsymbol{\theta}) : \boldsymbol{x} \in \mathbb{R}^D \mapsto \boldsymbol{z} \in \mathbb{R}^d$.



So far:

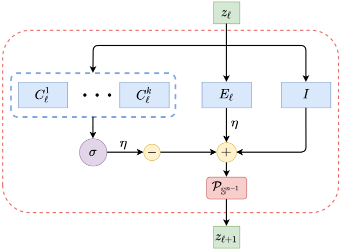
- **Resource requirements** to *identify* nonlinear manifolds with deep nets
- **Challenges with popular approaches** to *representation*

How to obtain a white-box architecture f that simultaneously identifies and represents large-scale datasets?

Recap: White-Box Deep Networks

A promising approach: signal models $\Longrightarrow$ deep architectures

- Convolutional sparse coding networks [Papyan et al. 2018]
- Scattering networks [Bruna & Mallat 2013]
- ReduNets [Chan, Yu et al. 2022]



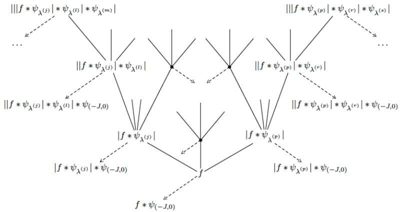


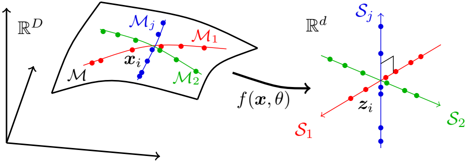
Fig. 2: Scattering network architecture based on wavelet filters and the modulus non-linearity. The elements of the feature vector $\Phi_W(f)$ in (1) are indicated at the tips of the arrows.

Figure: Left: **ReduNet** layer. Right: **Scattering Network** [Bruna & Mallat 2013] [Wiatowski & Bölcskei 2018] (**only 2-3 layers**).

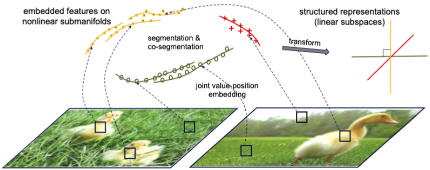
Pitfall of existing methods: Challenging to scale to massive datasets with strong performance

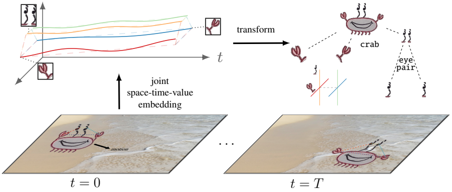
Improved White-Box Scaling by Improved Signal Modeling?

So far: *Each* **sample** *is drawn from a mixture of manifolds*



Better? *Each sample* $\supset$ **correlated tokens**—*mixture of manifold marginals!*

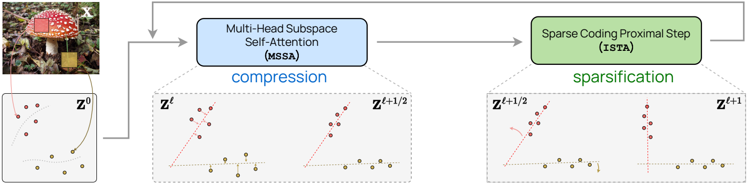




CRATE: A White-Box Transformer via Sparse MCR^2

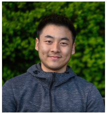
A white-box, mathematically interpretable, transformer-like deep network architecture from **iterative unrolling** optimization schemes to incrementally optimize the sparse rate reduction objective:

$$\max_{f \in \mathcal{F}} \mathbb{E}_{\boldsymbol{Z}} \left[\Delta R(\boldsymbol{Z}; \boldsymbol{U}_{[K]}) - \|\boldsymbol{Z}\|_0 \right], \quad \boldsymbol{Z} = f(\boldsymbol{X}).$$



CRATE: White-Box Transformers via Sparse Rate Reduction

https://arxiv.org/abs/2306.01129



Yaodong Yu (UCB)

Druv Pai (UCB)

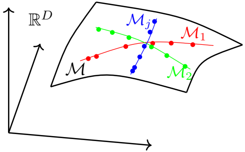
Sparse MCR^2 Objective and Incremental Representation

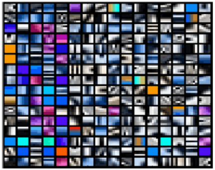
The sparse rate reduction (Sparse MCR^2) objective is defined as

$$\arg\max_{f \in \mathcal{F}} \mathbb{E}_{\boldsymbol{Z}} \left[\Delta R(\boldsymbol{Z}; \boldsymbol{U}_{[K]}) - \|\boldsymbol{Z}\|_0 \right]$$

$$= \arg\min_{f \in \mathcal{F}} \mathbb{E}_{\boldsymbol{Z}} \Big[\underbrace{R^c(\boldsymbol{Z}; \boldsymbol{U}_{[K]})}_{\text{compression}} + \underbrace{\|\boldsymbol{Z}\|_0 - R(\boldsymbol{Z})}_{\text{sparsification}} \Big].$$

$\boldsymbol{U}_{[K]} = (\boldsymbol{U}_1, \dots, \boldsymbol{U}_K)$, $\boldsymbol{U}_k \in \mathbb{R}^{d \times p}$ are *subspaces parameterizing the marginal distribution of tokens* $(\boldsymbol{z}_i)_{i=1}^N$





Sparse MCR^2 Objective and Incremental Representation

The sparse rate reduction (Sparse MCR^2) objective is defined as

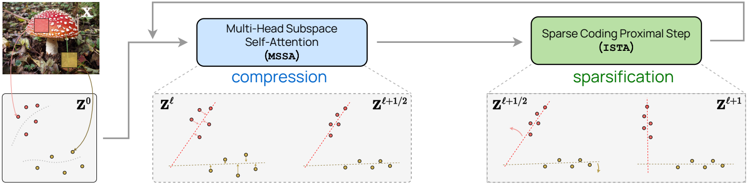
$$\arg\max_{f \in \mathcal{F}} \mathbb{E}_{\boldsymbol{Z}} \left[\Delta R(\boldsymbol{Z}; \boldsymbol{U}_{[K]}) - \|\boldsymbol{Z}\|_0 \right]$$

$$= \arg\min_{f \in \mathcal{F}} \mathbb{E}_{\boldsymbol{Z}} \Big[\underbrace{R^c(\boldsymbol{Z}; \boldsymbol{U}_{[K]})}_{\text{compression}} + \underbrace{\|\boldsymbol{Z}\|_0 - R(\boldsymbol{Z})}_{\text{sparsification}} \Big].$$

The global transformation f is realized through **local transformations**:

$$f\colon \boldsymbol{X} \xrightarrow{f^0} \boldsymbol{Z}^0 \to \cdots \to \boldsymbol{Z}^\ell \xrightarrow{f^\ell} \boldsymbol{Z}^{\ell+1} \to \cdots \to \boldsymbol{Z}^L = \boldsymbol{Z}.$$

Each f^ℓ deforms $\boldsymbol{Z}^\ell$ according to its own **local signal model** $\boldsymbol{U}^\ell_{[K]}$.



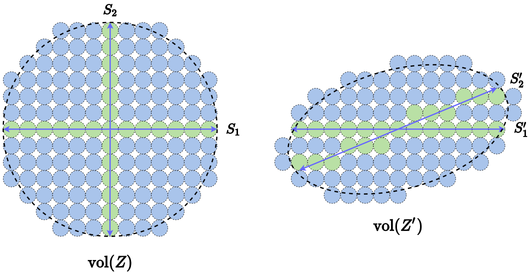
Recap: Compression and Expansion in MCR^2

Compression:

$$R^c(\boldsymbol{Z};\boldsymbol{U}_{[K]}) = \frac{1}{2}\sum_{k=1}^{K} \operatorname{logdet}\left(\boldsymbol{I} + \frac{p}{N\epsilon^2}(\boldsymbol{U}_k^*\boldsymbol{Z})^*(\boldsymbol{U}_k^*\boldsymbol{Z})\right)$$

Expansion:

$$R(\boldsymbol{Z}) = \frac{1}{2}\sum_{k=1}^{K} \operatorname{logdet}\left(\boldsymbol{I} + \frac{d}{N\epsilon^2}\boldsymbol{Z}^*\boldsymbol{Z}\right)$$

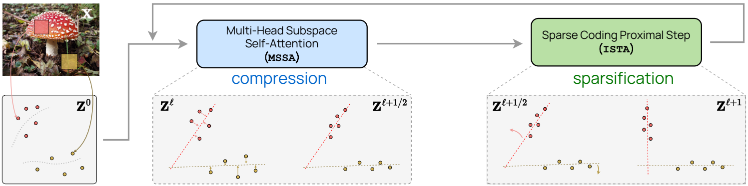


Sparse MCR^2 Objective and Incremental Representation

The sparse rate reduction (Sparse MCR^2) objective is defined as

$$\arg\max_{f \in \mathcal{F}} \mathbb{E}_{\boldsymbol{Z}} \left[\Delta R(\boldsymbol{Z}; \boldsymbol{U}_{[K]}) - \|\boldsymbol{Z}\|_0\right]$$

$$= \arg\min_{f \in \mathcal{F}} \mathbb{E}_{\boldsymbol{Z}} \Big[\underbrace{R^c(\boldsymbol{Z}; \boldsymbol{U}_{[K]})}_{\text{compression}} + \underbrace{\|\boldsymbol{Z}\|_0 - R(\boldsymbol{Z})}_{\text{sparsification}}\Big].$$



How to construct a representation f to incrementally optimize the compression term and the sparsification term?

Compression in Sparse MCR^2

To optimize the compression term $R^c(\boldsymbol{Z}; \boldsymbol{U}_{[K]})$, we propose to compress the set of tokens against the subspaces $(\boldsymbol{U}_k)_{k=1}^K$ by minimizing the coding rate via "approximate" gradient descent

$$\text{(Gradient Descent):} \quad \boldsymbol{Z}^\ell - \kappa \nabla_{\boldsymbol{Z}} R^c(\boldsymbol{Z}^\ell; \boldsymbol{U}_{[K]}) \approx \left(1 - \kappa \cdot \frac{p}{N\epsilon^2}\right) \boldsymbol{Z}^\ell + \kappa \cdot \frac{p}{N\epsilon^2} \cdot \texttt{MSSA}(\boldsymbol{Z}^\ell | \boldsymbol{U}_{[K]}),$$

where MSSA is defined through an SSA operator as:

$$\texttt{SSA}(\boldsymbol{Z}|\boldsymbol{U}_k) = (\boldsymbol{U}_k^* \boldsymbol{Z})\,\text{softmax}((\boldsymbol{U}_k^* \boldsymbol{Z})^* (\boldsymbol{U}_k^* \boldsymbol{Z})),$$

$$\texttt{MSSA}(\boldsymbol{Z}|\boldsymbol{U}_{[K]}) = \frac{p}{N\epsilon^2} \cdot [\boldsymbol{U}_1, \ldots, \boldsymbol{U}_K] \begin{bmatrix} \texttt{SSA}(\boldsymbol{Z}|\boldsymbol{U}_1) \\ \vdots \\ \texttt{SSA}(\boldsymbol{Z}|\boldsymbol{U}_K) \end{bmatrix}.$$

No need for separate query-Q, key-K, value-V in transformer attention block.

Compression in Sparse MCR^2

To optimize the compression term $R^c(\boldsymbol{Z}; \boldsymbol{U}_{[K]})$, we propose to compress the set of tokens against the subspaces $(\boldsymbol{U}_k)_{k=1}^{K}$ by minimizing the coding rate via "approximate" gradient descent

$$\boldsymbol{Z}^{\ell+1/2} = \boldsymbol{Z}^{\ell} + \texttt{MSSA}(\boldsymbol{Z}^{\ell} | \boldsymbol{U}_{[K]}).$$

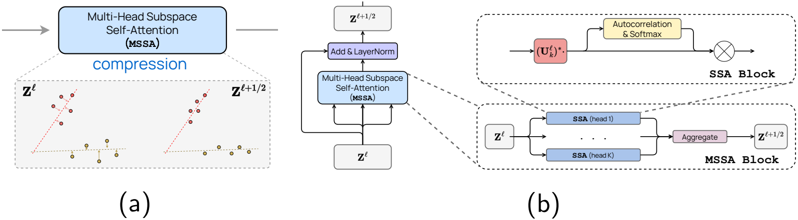


Figure: (a). Visualization of MSSA block; (b). Architecture of MSSA block.

Sparsification in Sparse MCR^2

To optimize the sparsification term $\|\boldsymbol{Z}\|_0 - R(\boldsymbol{Z})$, we posit a incoherent or orthogonal dictionary $\boldsymbol{D} \in \mathbb{R}^{d \times d}$ and sparsify $\boldsymbol{Z}^{\ell+1/2}$ with respect to $\boldsymbol{D}$, that is

$$\boldsymbol{Z}^{\ell+1/2} = \boldsymbol{D}\boldsymbol{Z}^{\ell+1}.$$

By the incoherence assumption, we have $\boldsymbol{D}^*\boldsymbol{D} \approx \boldsymbol{I}_d$; thus

$$R(\boldsymbol{Z}^{\ell+1}) \approx R(\boldsymbol{D}\boldsymbol{Z}^{\ell+1}) = R(\boldsymbol{Z}^{\ell+1/2}).$$

Thus we approximately optimize the sparsification objective with the following program:

$$\boldsymbol{Z}^{\ell+1} = \text{argmin}_{\boldsymbol{Z}}\|\boldsymbol{Z}\|_0 \quad \text{subject to} \quad \boldsymbol{Z}^{\ell+1/2} = \boldsymbol{D}\boldsymbol{Z}.$$

Sparsification in Sparse MCR^2

Given the sparse representation program

$$Z^{\ell+1} = \text{argmin}_{Z} \|Z\|_0 \quad \text{subject to} \quad Z^{\ell+1/2} = DZ.$$

we can relax it to an convex program, i.e., positive sparse coding:

$$Z^{\ell+1} = \arg\min_{Z \geq 0} \left[\lambda \|Z\|_1 + \|Z^{\ell+1/2} - DZ\|_F^2\right].$$

We can incrementally optimize the above objective by performing an unrolled proximal gradient descent step, known as an ISTA step:

$$\begin{aligned} Z^{\ell+1} &= \mathrm{ReLU}(Z^{\ell+1/2} + \eta D^*(Z^{\ell+1/2} - DZ^{\ell+1/2}) - \eta\lambda \mathbf{1}) \\ &:= \texttt{ISTA}(Z^{\ell+1/2} \mid D^{\ell}). \end{aligned}$$

The ISTA block uses much fewer parameters than transformer MLP block, and provides more interpretable representations.

Sparsification in Sparse MCR^2

To optimize the sparsification term $\|\boldsymbol{Z}\|_0 - R(\boldsymbol{Z})$, we propose to apply an unrolled proximal gradient descent step, known as an ISTA step:

$$\begin{aligned}\boldsymbol{Z}^{\ell+1} &= \mathrm{ReLU}(\boldsymbol{Z}^{\ell+1/2} + \eta \boldsymbol{D}^{*}(\boldsymbol{Z}^{\ell+1/2} - \boldsymbol{D}\boldsymbol{Z}^{\ell+1/2}) - \eta\lambda\mathbf{1}) \\ &:= \mathtt{ISTA}(\boldsymbol{Z}^{\ell+1/2} \mid \boldsymbol{D}^{\ell}).\end{aligned}$$

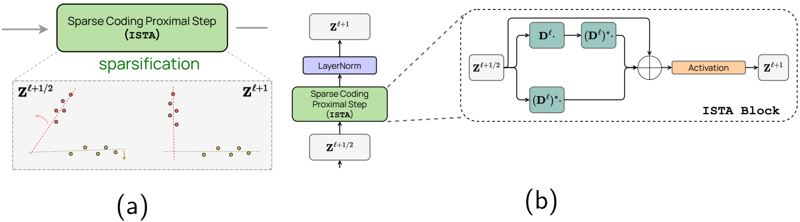


Figure: (a). Visualization of ISTA block; (b). Architecture of ISTA block.

One Layer of CRATE

Each layer of **CRATE** thus incrementally optimizes the compression term $R^c(\boldsymbol{Z}; \boldsymbol{U}_{[K]})$ and sparsification term $\|\boldsymbol{Z}\|_0 - R(\boldsymbol{Z})$,

$$\boldsymbol{Z}^{\ell+1} = f^\ell(\boldsymbol{Z}^\ell) = \texttt{ISTA}\Big(\underbrace{(\texttt{Id} + \texttt{MSSA})(\boldsymbol{Z}^\ell)}_{\boldsymbol{Z}^{\ell+1/2}}\Big).$$

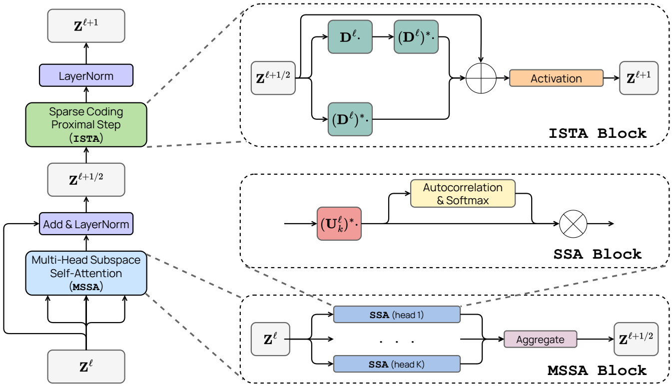
More specifically,

$$\begin{aligned} \boldsymbol{Z}^{\ell+1/2} &= \boldsymbol{Z}^\ell + \texttt{MSSA}(\boldsymbol{Z}^\ell \mid \boldsymbol{U}^\ell_{[K]}), && \text{[Compression step]} \\ \boldsymbol{Z}^{\ell+1} &= \texttt{ISTA}(\boldsymbol{Z}^{\ell+1/2} \mid \boldsymbol{D}^\ell), && \text{[Sparsification step]} \end{aligned}$$

so the ℓ-th layer of the global representation f is

$$f^\ell \colon \boldsymbol{Z}^\ell \xrightarrow{\texttt{Id+MSSA}} \boldsymbol{Z}^{\ell+1/2} \xrightarrow{\texttt{ISTA}} \boldsymbol{Z}^{\ell+1}.$$

Overall White-Box CRATE Architecture



- Forward optimization: perform compression and sparsification.
- Learning from data: apply SGD to learn $(U_{[K]}^{\ell}, D^{\ell})_{\ell=1}^{L}$ from data.

Experiment I: Supervised Learning on ImageNet-1K

Experimental setup: let the CLS token of Z^L (i.e., the output token set of the last layer), and then apply a linear linear to perform supervised learning on ImageNet-1K using our proposed CRATE architecture.

Table 1: Top 1 accuracy of CRATE on various datasets with different model scales when pre-trained on ImageNet. For ImageNet/ImageNetReaL, we directly evaluate the top-1 accuracy. For other datasets, we use models that are pre-trained on ImageNet as initialization and the evaluate the transfer learning performance via fine-tuning.

Datasets	CRATE-T	CRATE-S	CRATE-B	CRATE-L	ViT-T	ViT-S
# parameters	6.09M	13.12M	22.80M	77.64M	5.72M	22.05M
ImageNet	66.7	69.2	70.8	71.3	71.5	72.4
ImageNet ReaL	74.0	76.0	76.5	77.4	78.3	78.4

- CRATE demonstrates promising performance on the ImageNet-1K dataset, indicating its potential for further advancement.

Experiment I: Supervised Learning on ImageNet-1K

Experimental setup: apply the CRATE model pre-trained on ImageNet-1K as initialization, and then evaluate transfer learning performance via fine-tuning.

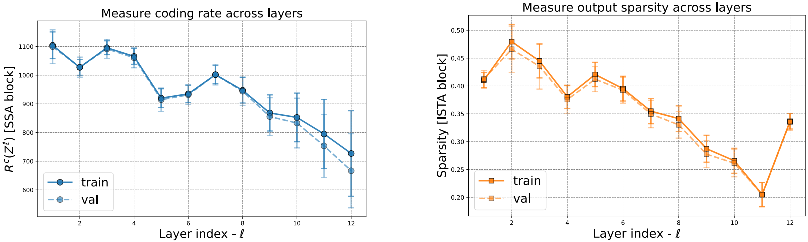
Table 1: Top 1 accuracy of CRATE on various datasets with different model scales when pre-trained on ImageNet. For ImageNet/ImageNetReaL, we directly evaluate the top-1 accuracy. For other datasets, we use models that are pre-trained on ImageNet as initialization and the evaluate the transfer learning performance via fine-tuning.

Datasets	CRATE-T	CRATE-S	CRATE-B	CRATE-L	ViT-T	ViT-S
# parameters	6.09M	13.12M	22.80M	77.64M	5.72M	22.05M
ImageNet	66.7	69.2	70.8	71.3	71.5	72.4
ImageNet ReaL	74.0	76.0	76.5	77.4	78.3	78.4
CIFAR10	95.5	96.0	96.8	97.2	96.6	97.2
CIFAR100	78.9	81.0	82.7	83.6	81.8	83.2
Oxford Flowers-102	84.6	87.1	88.7	88.3	85.1	88.5
Oxford-IIIT-Pets	81.4	84.9	85.3	87.4	88.5	88.6

- CRATE achieves performance close to thoroughly engineered vision transformers.
- Promising scaling behavior in CRATE.

Experiment II: Layer-wise Analysis of CRATE

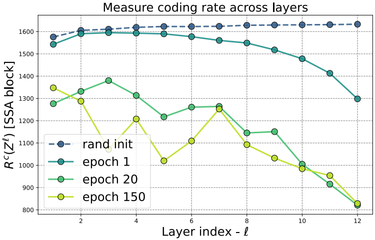
Given a learned CRATE model, we measure the compression term of $Z^{\ell+1/2}$ (*left*, $R^c(Z^{\ell+1/2})$) and the sparsification term of $Z^{\ell+1}$ (*right*, $\|Z^{\ell+1}\|_0$) on train/validation samples at **each layer**.

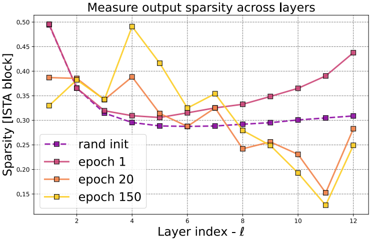


- The learned CRATE model indeed performs its design objective – each layer incrementally optimizes the compression term and the sparsification term.

Experiment II: Layer-wise Analysis of CRATE

For comparison, we measure the compression/sparsification term of randomly initialized CRATE model and models at different epochs.

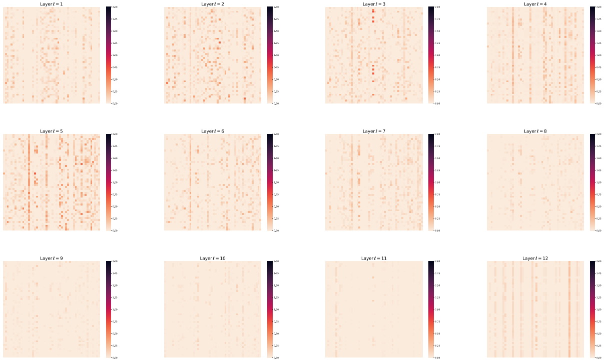




- Without learning from data, the random initialized CRATE model does not perform its design objective effectively.

Experiment III: Visualize Layer-wise Output of CRATE

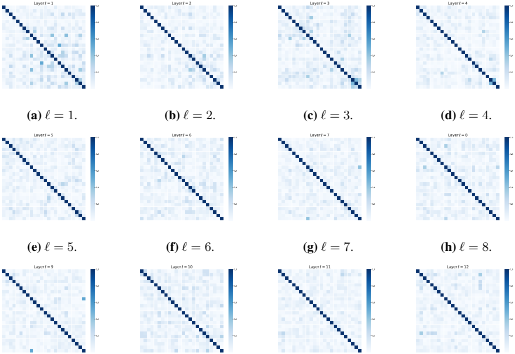
We use heatmaps to visualize the output of each layer in CRATE ($Z^{\ell+1}$).



- We observe clear sparse and low-rank patterns of intermediate outputs of CRATE.

Experiment IV: Visualize Learned Subspaces of CRATE

We use heatmaps to visualize the correlations between different subspaces $(\boldsymbol{U}_k)_{k=1}^K$ of each `MSSA` layer in CRATE, i.e., $[\boldsymbol{U}_1^\ell, \ldots, \boldsymbol{U}_K^\ell]^*[\boldsymbol{U}_1^\ell, \ldots, \boldsymbol{U}_K^\ell]$.



(a) $\ell = 1$. **(b)** $\ell = 2$. **(c)** $\ell = 3$. **(d)** $\ell = 4$.
(e) $\ell = 5$. **(f)** $\ell = 6$. **(g)** $\ell = 7$. **(h)** $\ell = 8$.
(i) $\ell = 9$. **(j)** $\ell = 10$. **(k)** $\ell = 11$. **(l)** $\ell = 12$.

- The learned subspaces in `MSSA` blocks are incoherent.

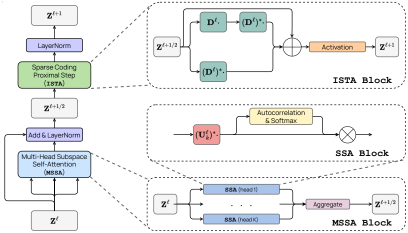
Outline

A Parting Message

We've seen today

- What **structures in modern data** are we learning?
- **Resource requirements** for identifying nonlinear manifolds
- **Manifold representation** with manifold learning and diffusion
- **Joint identification/representation** via white-box transformers

For white-box deep networks, the future is bright!



figures/diffusion-iteratic

Thank You! Questions?

Call for Papers

- IEEE JSTSP Special Issue on Seeking Low-dimensionality in Deep Neural Networks (SLowDNN) Manuscript Due: **Nov. 30, 2023**.
- Conference on Parsimony and Learning (CPAL) January 2024, Hongkong, Manuscript Due: **Aug. 28, 2023**.

SCAN ME

SCAN ME

CEU/PDH Certificates

You can receive an CEU/PDH certificate by completing the course and pass the quiz. Here is the quiz/evaluation form:

`https://bit.ly/ICASSP23_QuizSC2`