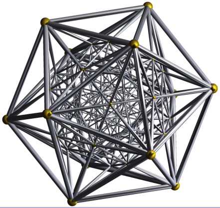
ICASSP22 Short Course One on
Low-Dimensional Models for High-Dimensional Data

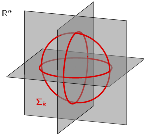
Lecture 4:
Learning Low-Dimensional Structure via Deep Networks

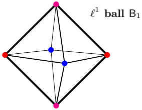
Sam Buchanan, Yi Ma, Qing Qu
John Wright, Yuqian Zhang, Zhihui Zhu

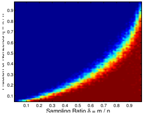
May 26, 2022



Recap: Sparse Approximation (Linear, Convex)







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

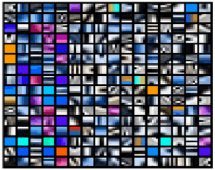
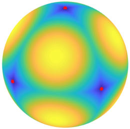
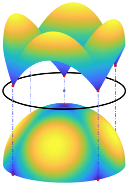
$$\boldsymbol{x}_\star = \arg\min_{\boldsymbol{x} \in \mathbb{R}^n} \frac{1}{2}\|\boldsymbol{y} - \boldsymbol{A}\boldsymbol{x}\|_2^2 + \lambda\|\boldsymbol{x}\|_1$$

and provable (high probability) guarantees

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

Recap: Dictionary Learning (Bilinear, Nonconvex)



Dictionary Learning: **structured** signals, **bilinear** measurements

$$\boldsymbol{Y} = \boldsymbol{A}_o \boldsymbol{X}_o \in \mathbb{R}^{n \times p}, \quad \boldsymbol{X}_o \text{ sparse and random}, \quad \boldsymbol{A}_o^* \boldsymbol{A}_o \approx \boldsymbol{I}$$

with (efficient) **nonconvex** optimization

$$\boldsymbol{a}_\star = \underset{\|\boldsymbol{a}\|_2 = 1}{\arg\min} \|\boldsymbol{Y}^* \boldsymbol{a}\|_1$$

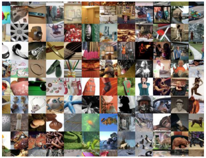
and provable (high probability) guarantees

$$\boldsymbol{a}_\star \approx (\boldsymbol{A}_o)_j \ \text{ when } \ \text{observations} \geq \mathrm{poly}(\text{expected sparsity})$$

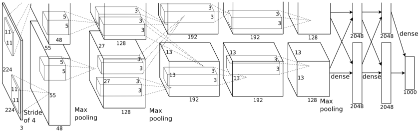
Today: Deep Learning (Very Nonlinear, Extra Nonconvex)

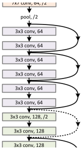
Supervised Deep Learning: Given labeled data

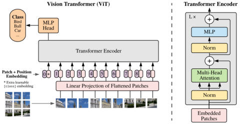
$$\left\{ \left(\underbrace{\boldsymbol{x}_i}_{\text{data (images, text, ...)}}, \underbrace{\boldsymbol{y}_i}_{\text{labels (classes, values, ...)}} \right) \right\}_{i=1}^{N}$$



fit a mapping f : parameters $\times$ data $\to$ labels



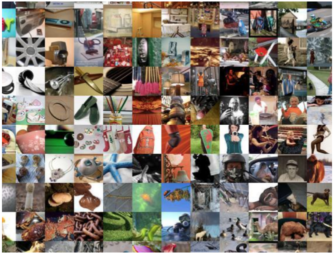


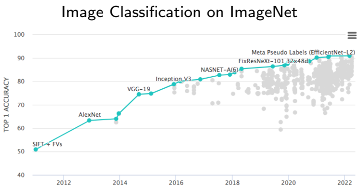


using stochastic gradient descent on a task-appropriate loss

$$\underbrace{\boldsymbol{\theta}_\star = \mathrm{SGD}_{\boldsymbol{\theta}} \left(\frac{1}{2} \sum_{i=1}^{N} \| f_{\boldsymbol{\theta}}(\boldsymbol{x}_i) - \boldsymbol{y}_i \|_2^2 \right)}_{\text{regression}}$$

Today's Lectures





Big question: What role does **low-dimensional structure** play in the **practice** of deep learning?

TEXT PROMPT

an illustration of a baby daikon radish in a tutu walking a dog

AI-GENERATED IMAGES

TEXT DESCRIPTION

An astronaut Teddy bears A bowl of soup

mixing sparkling chemicals as mad scientists shopping for groceries working on new AI research

as kids' crayon art on the moon in the 1980s underwater with 1990s technology

→

DALL·E 2

Outlook for Today's Lectures

Answer: A huge role!

Today:

- **Nonlinear** low-dimensional structures in practical data necessitate the use of **deep networks** over classical models;

Outlook for Today's Lectures

Answer: A huge role!

Today:

- **Nonlinear** low-dimensional structures in practical data necessitate the use of **deep networks** over classical models;
- **A mathematical model problem** helps understand **resource tradeoffs** between data geometry and network architecture (a **nonlinear generalization** of the sparse approximation analysis!);

Outlook for Today's Lectures

Answer: A huge role!

Today:

- **Nonlinear** low-dimensional structures in practical data necessitate the use of **deep networks** over classical models;
- **A mathematical model problem** helps understand **resource tradeoffs** between data geometry and network architecture (a **nonlinear generalization** of the sparse approximation analysis!);
- **For classification problems**, understand the features learned by deep neural networks, and improve training robustness using insights from low-dimensional structure;

Outlook for Today's Lectures

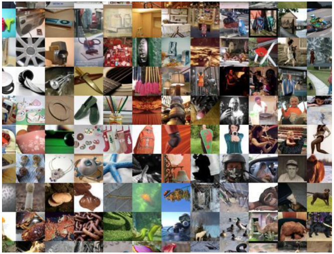
Answer: A huge role!

Today:

- **Nonlinear** low-dimensional structures in practical data necessitate the use of **deep networks** over classical models;
- **A mathematical model problem** helps understand **resource tradeoffs** between data geometry and network architecture (a **nonlinear generalization** of the sparse approximation analysis!);
- **For classification problems**, understand the features learned by deep neural networks, and improve training robustness using insights from low-dimensional structure;
- **Whitebox design of deep networks** for pursuing nonlinear low-dim structures. (Lecture 5)

Outline

Low-Dimensional Structure in Deep Learning Problems



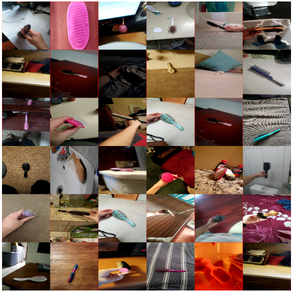
Questions:

What is an appropriate mathematical model for data with low-dimensional structure in deep learning applications?

What insights into practical deep learning can we get by studying low-dimensional structure?

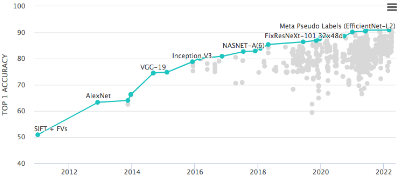
Vignette I: Large-Scale Image Classification

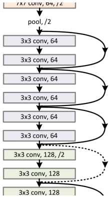
Task: Learn a deep network mapping images → object classes from data.



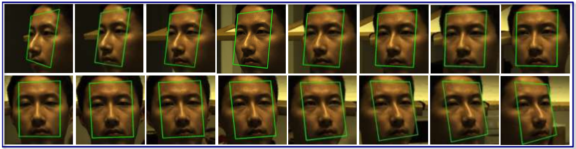
→ {hedgehog, hairbrush}

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





Nonlinear Variabilities in Natural Images

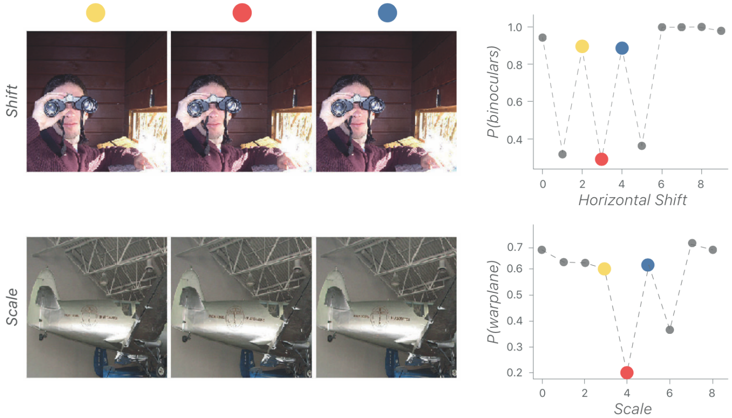


$\Longrightarrow$ **nonlinear, geometric** structure

- 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:

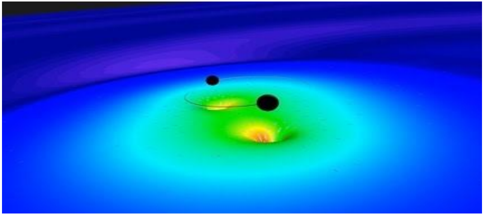


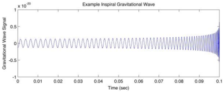
From [Azulay and Weiss, 2019]

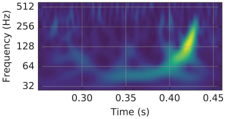
Vignette II: Deep Learning in Scientific Discovery

Gravitational Wave Astronomy

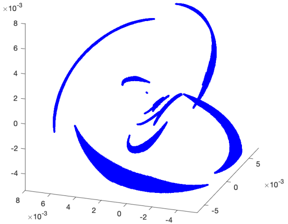
One binary black hole merger:



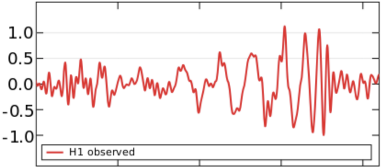


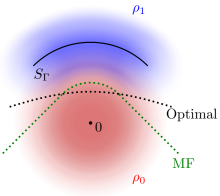


Many mergers
(varying mass M_1, M_2):
$\Longrightarrow$ **low-dim manifold**



Gravitational Wave Astronomy as Parametric Detection

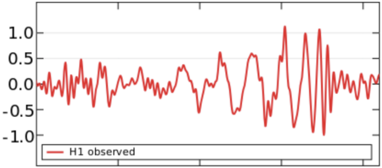


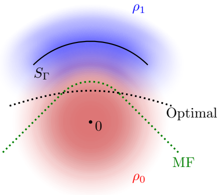


Is observation $\boldsymbol{x} = \boldsymbol{s}_\gamma + \boldsymbol{z}$ or $\boldsymbol{x} = \boldsymbol{z}$?

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

Gravitational Wave Astronomy as Parametric Detection



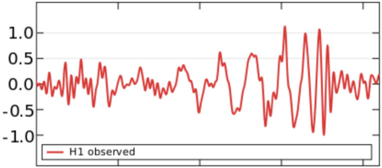


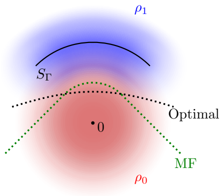
Is observation $\boldsymbol{x} = \boldsymbol{s}_\gamma + \boldsymbol{z}$ or $\boldsymbol{x} = \boldsymbol{z}$?

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

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

Gravitational Wave Astronomy as Parametric Detection





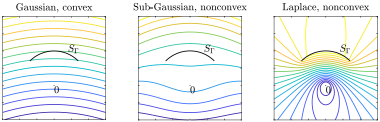
Is observation $\boldsymbol{x} = \boldsymbol{s}_\gamma + \boldsymbol{z}$ or $\boldsymbol{x} = \boldsymbol{z}$?

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

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

Issues: Optimality? Complexity?

Unknown unknowns? Unknown noise?

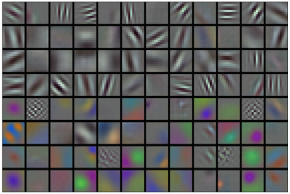


Ideally: Combine low-dim structure of Γ with data-driven for statistical structure...

Vignette III: Learning Features with Deep Learning for Downstream Tasks

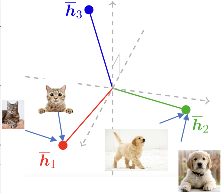
Ubiquitous deep learning workflow (science/engineering/industry):

1. Data-driven pretraining to learn good features (ImageNet pretraining; masked prediction)
2. Fine-tuning on specific downstream tasks for performance (tracking; segmentation; object detection; ...)



First-layer filters from AlexNet

Inception v4 activations



Neural collapse visualization

Issues: What features are learned? Robustness to imperfect labeling? How to incorporate prior knowledge about data/task?

Takeaways from the Examples

Two key takeaways:

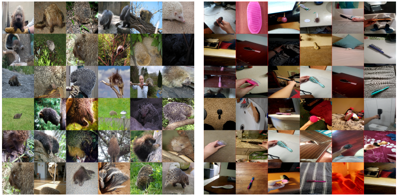
- Data with **nonlinear, geometric structure** pervade successful practical applications of deep learning
- Important practical issues (**robustness/invariance; resource efficiency; performance**) naturally linked to low-dim structure

Takeaways from the Examples

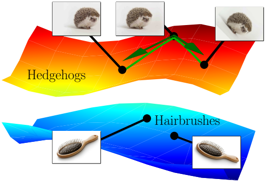
Two key takeaways:

- 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.



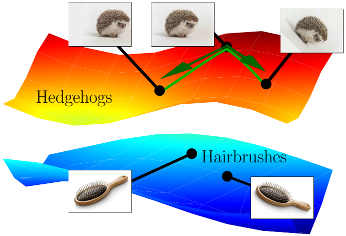
$\Longrightarrow$



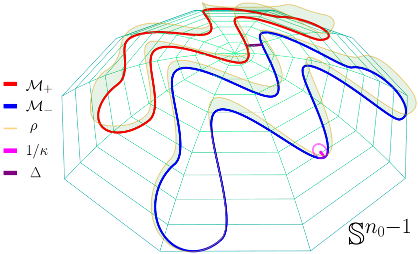
Outline

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

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

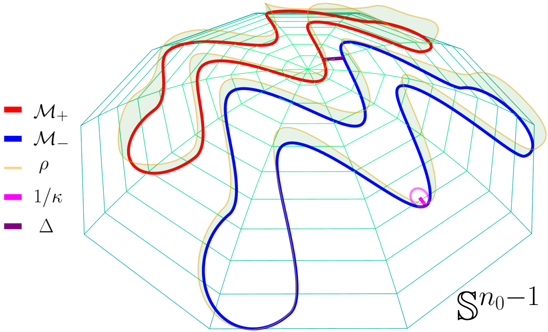


$\Longrightarrow$



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

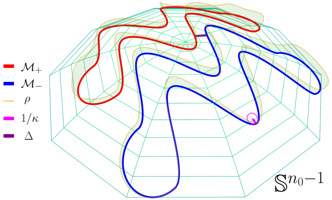
The Two Manifold Problem



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

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

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*:

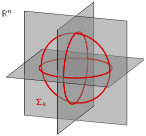
$$\operatorname{sign}(f_{\boldsymbol{\theta}}(\boldsymbol{x})) = y(\boldsymbol{x}) \quad \forall\, \boldsymbol{x} \in \mathcal{M}.$$

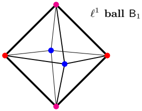
- Binary classification with a deep neural network
- High-dimensional data with (unknown!) low-dimensional structure
- Statistical structure, and asking for "strong" generalization

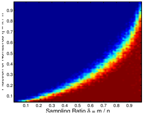
We will focus on the case of one-dimensional manifolds (curves)

What Can We Hope to Understand Here?

Our “barometer”: compressed sensing.





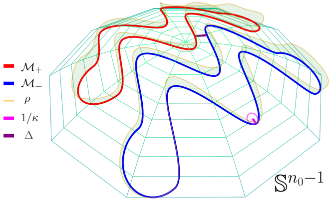


$$\boldsymbol{y} = \boldsymbol{A}\boldsymbol{x}_o; \qquad \boldsymbol{x}_\star = \arg\min_{\boldsymbol{x}\in\mathbb{R}^n} \frac{1}{2}\|\boldsymbol{y} - \boldsymbol{A}\boldsymbol{x}\|_2^2 + \lambda\|\boldsymbol{x}\|_1$$

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

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 $(\boldsymbol{x}_1, y(\boldsymbol{x}_1)), \ldots, (\boldsymbol{x}_N, y(\boldsymbol{x}_N))$ from $\mathcal{M} = \mathcal{M}_+ \cup \mathcal{M}_-$, use gradient descent to train a deep network $f_{\boldsymbol{\theta}}$ that *perfectly labels the manifolds*:

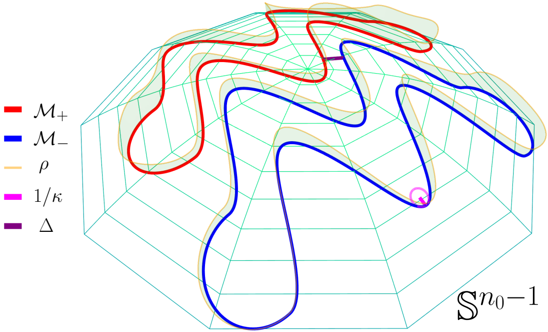
$$\operatorname{sign}\left(f_{\boldsymbol{\theta}}(\boldsymbol{x})\right) = y(\boldsymbol{x}) \quad \forall\, \boldsymbol{x} \in \mathcal{M}.$$

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?

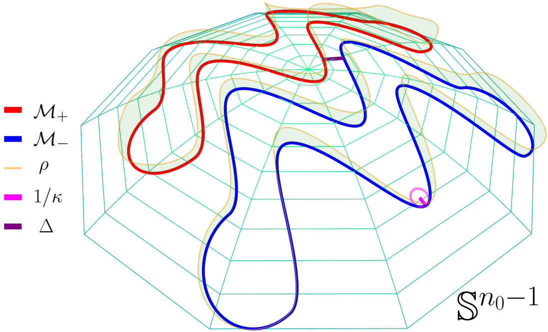


Mathematically:

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

Intrinsic Structural Properties II: Curvature

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

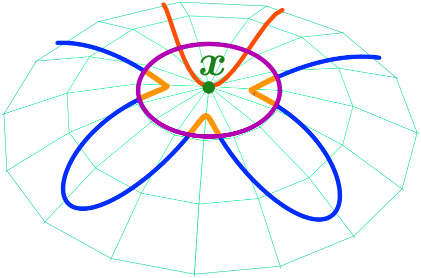


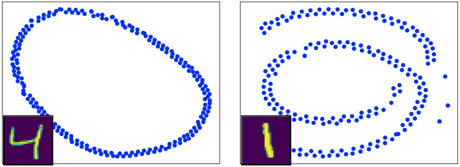
Mathematically:

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

Intrinsic Structural Properties III: ⌘-Number

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



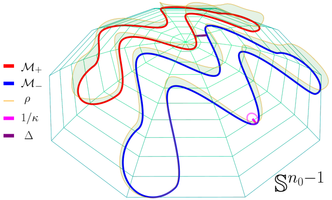


Mathematically:

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

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 $(\boldsymbol{x}_1, y(\boldsymbol{x}_1)), \ldots, (\boldsymbol{x}_N, y(\boldsymbol{x}_N))$ from $\mathcal{M} = \mathcal{M}_+ \cup \mathcal{M}_-$, use gradient descent to train a deep network $f_{\boldsymbol{\theta}}$ that *perfectly labels the manifolds*:

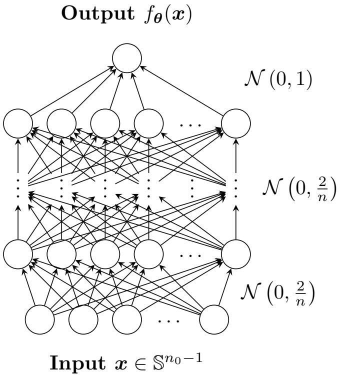
$$\operatorname{sign}(f_{\boldsymbol{\theta}}(\boldsymbol{x})) = y(\boldsymbol{x}) \quad \forall\, \boldsymbol{x} \in \mathcal{M}.$$

A set of 'sufficient' intrinsic problem difficulty parameters:

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

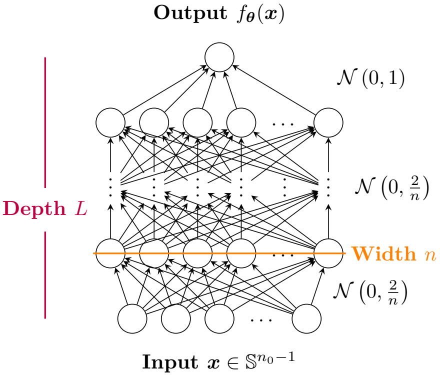
Network Architecture and Training Procedure

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



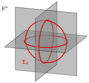
Network Architecture and Training Procedure

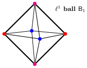
- Fully connected with ReLUs
- Gaussian initialization $\boldsymbol{\theta}_0$
- Trained with N i.i.d. samples from measure μ of density ρ

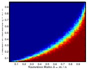


Resource Tradeoffs: From Linear to Nonlinear

The “linear” case (compressed sensing):



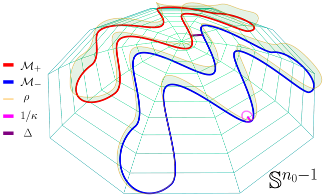




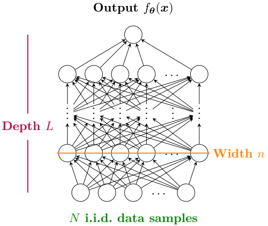
$$\boldsymbol{y} = \boldsymbol{A}\boldsymbol{x}_o; \qquad \boldsymbol{x}_\star = \arg\min_{\boldsymbol{x}\in\mathbb{R}^n} \frac{1}{2}\|\boldsymbol{y} - \boldsymbol{A}\boldsymbol{x}\|_2^2 + \lambda\|\boldsymbol{x}\|_1$$

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

Our current **nonlinear setting**:

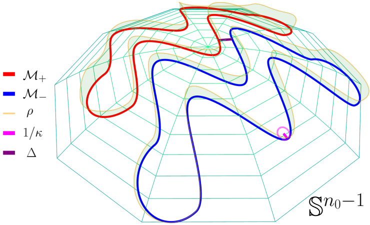


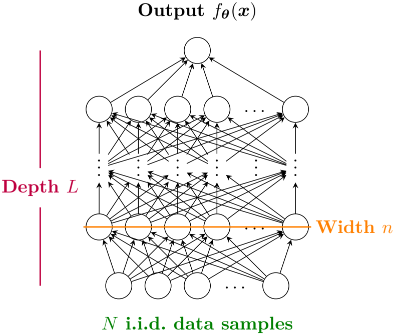
Data structure



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, $\circledast$; 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?

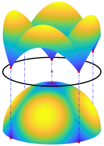
Gradient Descent Training

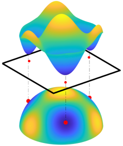
Objective: Square Loss on Training Data

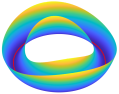
$$\min_{\boldsymbol{\theta}} \varphi(\boldsymbol{\theta}) \equiv \frac{1}{2} \int_{\mathcal{M}} \left(f_{\boldsymbol{\theta}}(\boldsymbol{x}) - y(\boldsymbol{x}) \right)^2 d\mu_N(\boldsymbol{x}).$$

Does gradient descent correctly label the manifolds?

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


Dictionary Learning


Sparse Blind Deconvolution


Matrix Recovery

See [Gilboa, B., Wright '18], survey [Zhang, Qu, Wright 20] (Lecture 3!)

Gradient Descent Training

Objective: Square Loss on Training Data

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

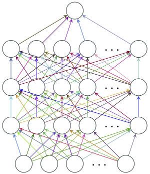
Does gradient descent correctly label the manifolds?

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

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

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:

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

Dynamics of Gradient Descent

The error evolves according to the NTK:

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

Dynamics of Gradient Descent

The error evolves according to the NTK:

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

Dynamics of Gradient Descent

The error evolves according to the NTK:

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

Dynamics of Gradient Descent

The error evolves according to the NTK:

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

Dynamics of Gradient Descent ("NTK Regime")

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

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

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.

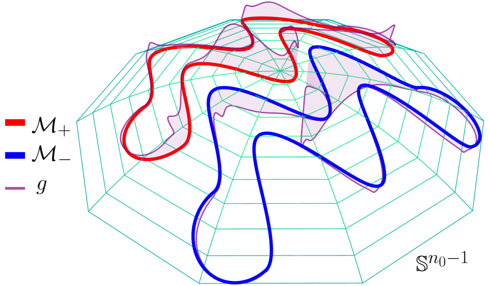
Implicit Error-NTK Alignment with Certificates

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.



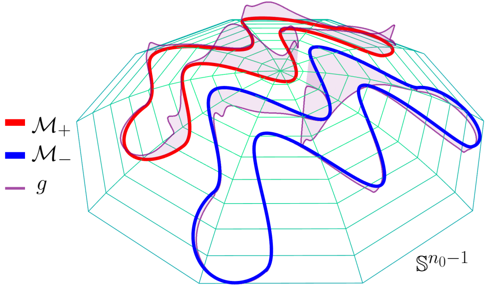
Implicit Error-NTK Alignment with Certificates

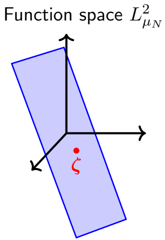
Challenge: For nonlinear $\mathcal{M}$, eigenvectors of $\boldsymbol{\Theta}$ are intractable!

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

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

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





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

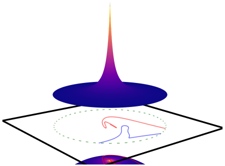
$$\|\zeta_t\|_{L^2_\mu} \lesssim \frac{L \log L}{t}.$$

Roles of Width, Depth, and Data

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

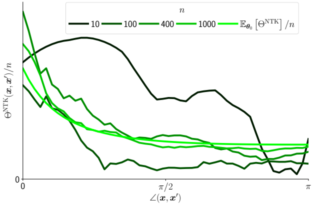
Questions:
How do width, depth, and samples affect $\boldsymbol{\Theta}$?
How does $\boldsymbol{\Theta}$ depend on the geometry of the data?

Depth L: **fitting resource**



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

Width n: **statistical resource**

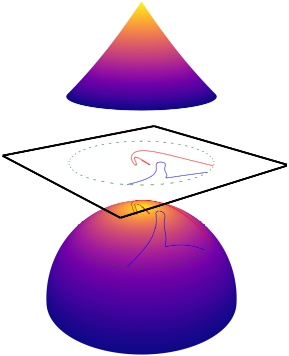


Resource Tradeoffs I: Depth as a Fitting Resource

Key insights:

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

$\implies$ Set depth based on geometry!



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

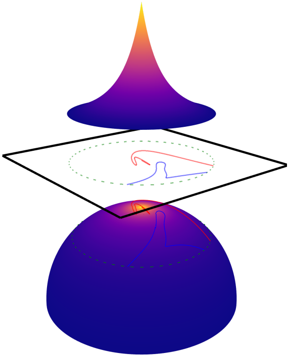
Deeper networks fit more complicated geometries.

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!



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

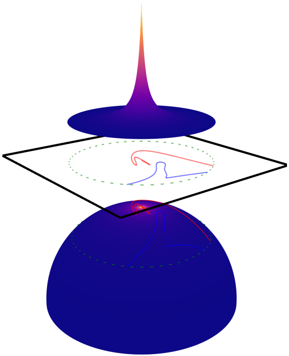
Deeper networks fit more complicated geometries.

Resource Tradeoffs I: Depth as a Fitting Resource

Key insights:

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

$\implies$ Set depth based on geometry!



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

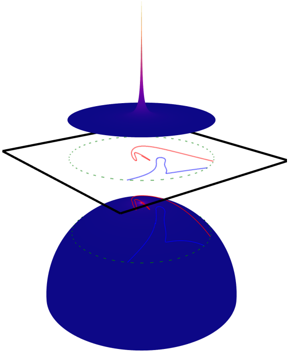
Deeper networks fit more complicated geometries.

Resource Tradeoffs I: Depth as a Fitting Resource

Key insights:

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

$\Longrightarrow$ Set depth based on geometry!

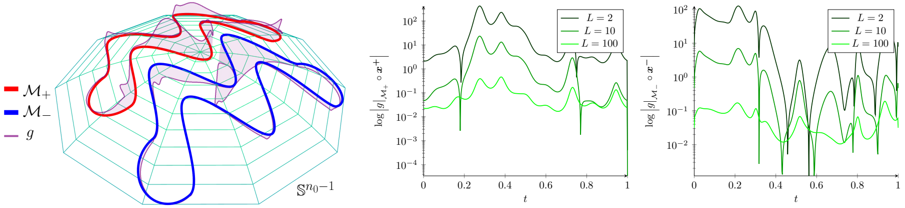


$\frac{1}{L}\Theta(\boldsymbol{e}_1, \boldsymbol{x}'),\ L = 625$

Deeper networks fit more complicated geometries.

Resource Tradeoffs I: Certificates from Depth

Numerical experiment:

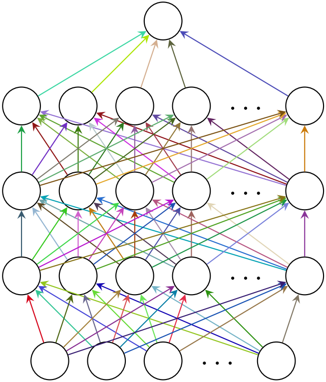


Depth as a fitting resource: Larger depth L leads to a sharper kernel Θ and a smaller certificate g

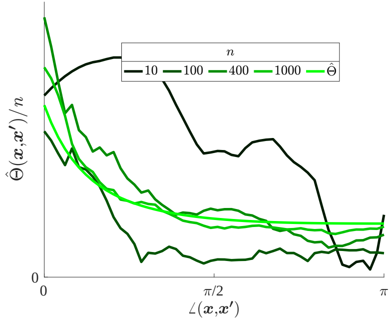
$\implies$ Easier fitting!

Resource Tradeoffs II: Width as a Statistical Resource

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



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



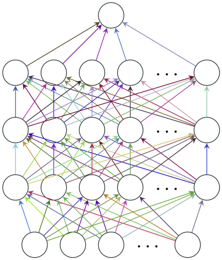
As width increases, $\Theta(\boldsymbol{x}, \boldsymbol{x}')$ concentrates about $\mathbb{E}_{\text{init weights}}[\Theta(\boldsymbol{x}, \boldsymbol{x}')]$

Resource Tradeoffs II: Width as a Statistical Resource

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

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

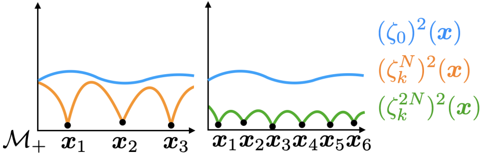
is small (simultaneously) for all $(\boldsymbol{x}, \boldsymbol{x}') \in \mathcal{M} \times \mathcal{M}$.

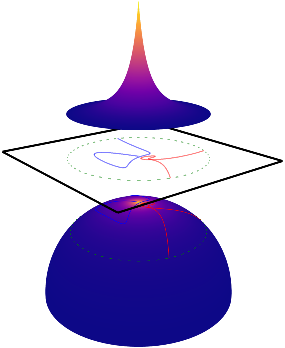


$\Rightarrow$ **set width n based on depth L**

and implicitly based on κ, Δ

Resource Tradeoffs III: Data as a Statistical Resource





Depth $L = 50$

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

End-to-End Generalization Guarantee

Theorem [B., Wang, Gilboa, Wright 2021]: 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

Image Classification Problem I

So far, Sam has talked about resources needed to ensure correctly classify two manifolds.

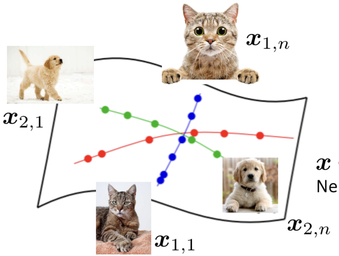
We will now focus on the general classification of K manifolds.

Instead of just on the output, we will focus more on the learned features and classifiers.

Image Classification Problem II

Labels: $k = 1, \dots, K$

- $K = 10$ classes (MNIST, CIFAR10, etc)
- $K = 1000$ classes (ImageNet)



Data in the input space

$$\boldsymbol{x} \xrightarrow[\text{Neural network}]{} \boldsymbol{y}$$

$$\begin{array}{cccc} \text{Cat} & \text{Dog} & & \text{Truck} \\ \begin{bmatrix} 1 \\ 0 \\ \vdots \\ 0 \end{bmatrix} & \begin{bmatrix} 0 \\ 1 \\ \vdots \\ 0 \end{bmatrix} & \cdots & \begin{bmatrix} 0 \\ 0 \\ \vdots \\ 1 \end{bmatrix} \end{array}$$

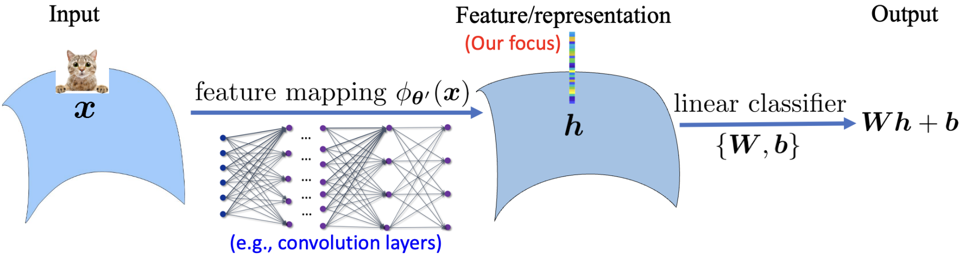
One-hot labeling vectors in $\mathbb{R}^K$

Assume balanced dataset where each class has n training samples

- If not, we can use data augmentation to make them balanced

Deep Neural Network Classifiers I

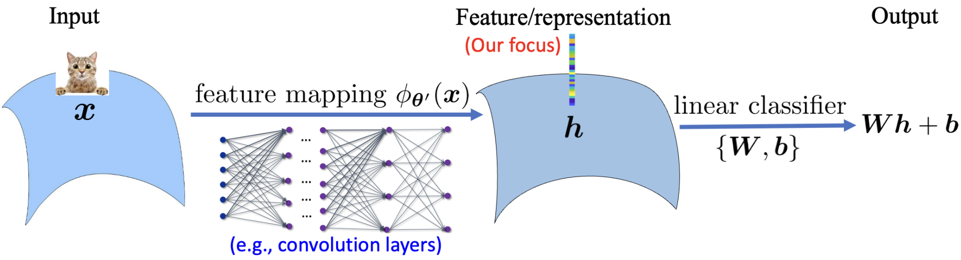
A deep neural network classifier often contains two parts: a feature mapping and a linear classifier



- Output: $f(x;\theta) = W\phi_{\theta'}(x) + b$ with $\theta = (\theta', W, b)$.
- Training problem:

$$\min_{\theta', W, b} \frac{1}{Kn} \sum_{k=1}^{K} \sum_{i=1}^{n} \underbrace{\mathcal{L}_{\mathrm{CE}}\left(W\phi_{\theta'}(x_{k,i}) + b, y_k\right)}_{\text{cross-entropy (CE) loss}} + \lambda \underbrace{\|(\theta', W, b)\|_F^2}_{\text{weight decay}}$$

Deep Neural Network Classifiers II



$$\text{Output: } f(\boldsymbol{x}; \boldsymbol{\theta}) = \begin{bmatrix} 1 \\ 0 \\ -1 \end{bmatrix} \xrightarrow{\text{Softmax function}} \begin{bmatrix} 0.6 \\ 0.3 \\ 0.1 \end{bmatrix} \begin{matrix} \text{Cat} \\ \text{Dog} \\ \text{Panda} \end{matrix} \begin{bmatrix} 1 \\ 0 \\ 0 \end{bmatrix}$$

Prediction (probability) Target

$$\begin{aligned} \mathrm{CE}(\mathrm{Cat}) :&= -q(\mathrm{Cat}) \cdot \log p(\mathrm{Cat}) \\ &= -1 \cdot \log 0.6 \\ &= 0.51\ldots \end{aligned}$$

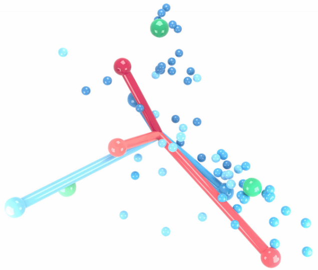
Neural Collapse in Classification I

Prevalence of neural collapse during the terminal phase of deep learning training

Vardan Papyan, X. Y. Han, and David L. Donoho
+ See all authors and affiliations

PNAS October 6, 2020 117 (40) 24652-24663; first published September 21, 2020;
https://doi.org/10.1073/pnas.2015509117

Contributed by David L. Donoho, August 18, 2020 (sent for review July 22, 2020; reviewed by Helmut Boelsckei and Stéphane Mallat)



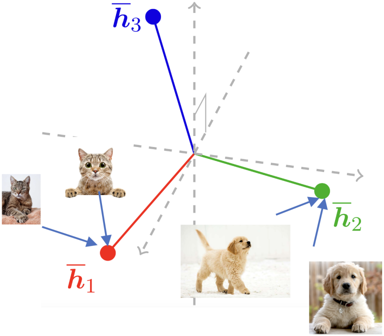
- Reveals common outcome of learned features and classifiers across a variety of architectures and dataset
- Precise mathematical structure within the features and classifier

Neural Collapse in Classification II

Neural Collapse (NC) refers to

- NC1: Within-Class Variability Collapse: features of each class collapse to class-mean with zero variability (*low-dimensional features*):

$$k\text{-th class}, i\text{-th sample}: \boldsymbol{h}_{k,i} \to \overline{\boldsymbol{h}}_k,$$

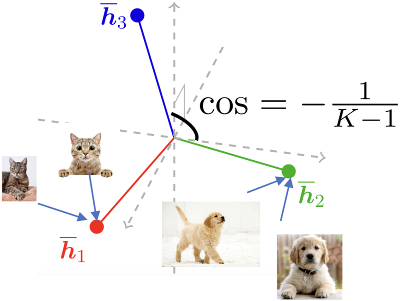


Neural Collapse in Classification III

Neural Collapse (NC) refers to

- NC2: Convergence to Simplex Equiangular Tight Frame (ETF): the class means are linearly separable, have same length, and maximal angle between each other

$$\frac{\langle \overline{\boldsymbol{h}}_k, \overline{\boldsymbol{h}}_{k'} \rangle}{\|\overline{\boldsymbol{h}}_k\| \|\overline{\boldsymbol{h}}_{k'}\|} \to \begin{cases} 1, & k = k' \\ -\frac{1}{K-1}, & k \neq k' \end{cases}$$



- If K vectors have equal angle between each other, then the largest possible cosine angle between each pair is $-\frac{1}{K-1}$.

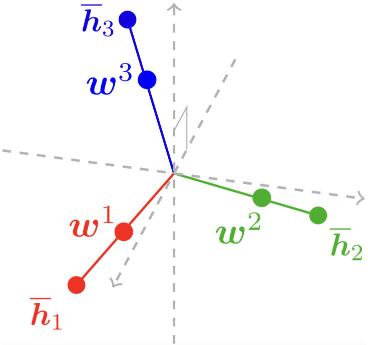
Neural Collapse in Classification IV

Neural Collapse (NC) refers to

- NC3: Convergence to Self-Duality: the last-layer classifiers are perfectly matched with the class-means of features

$$\frac{\boldsymbol{w}^k}{\|\boldsymbol{w}^k\|} \to \frac{\overline{\boldsymbol{h}}_k}{\|\overline{\boldsymbol{h}}_k\|},$$

where $\boldsymbol{w}^k$ represents the k-th row of $\boldsymbol{W}$.



Neural Collapse in Classification V

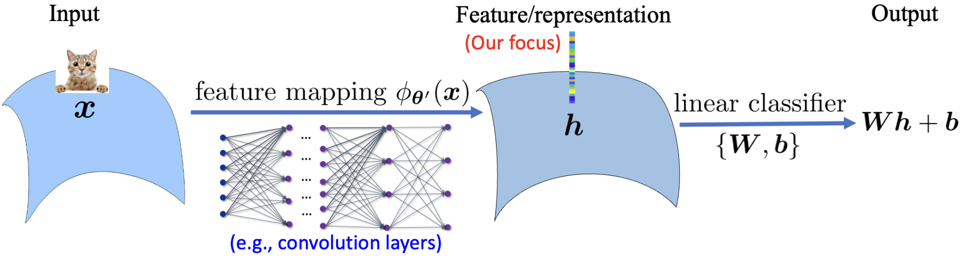
NC is preferred among every successful exercise in feature engineering [Papyan et al.'20]

- Information Theory: Simplex ETF is the optimal Shannon code
- Classification: Simple ETF features $\Rightarrow$ Simplex ETF max-margin classifier

Q: Why iterative training algorithm learns low-dimensional NC features and classifiers?

A: We will use tools developed in nonconvex optimization in Lecture 3 to understand NC phenomenon

Simplification: Unconstrained Features I

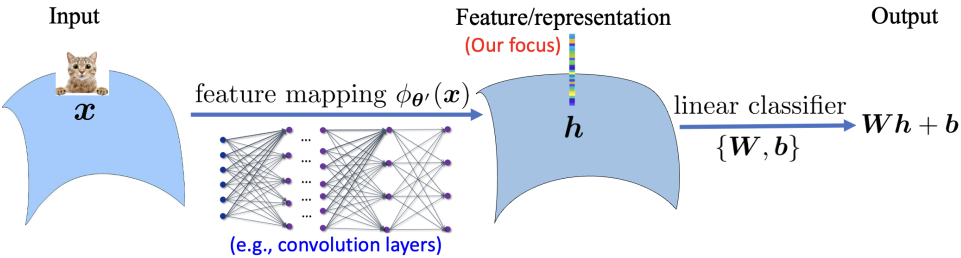


Training problem is highly nonconvex [Li et al.'18]:

$$\min_{\theta', W, b} \frac{1}{Kn} \sum_{k=1}^{K} \sum_{i=1}^{n} \mathcal{L}_{\mathrm{CE}}\left(W \phi_{\theta'}(x_{k,i}) + b, y_k\right) + \lambda \|(\theta', W, b)\|_F^2$$

- Neural Tangent Kernel focuses on output, and thus hardly provides much insights about features
- Neural Collapse is about the classifier W and the features $\phi_{\theta'}(x_{k,i})$

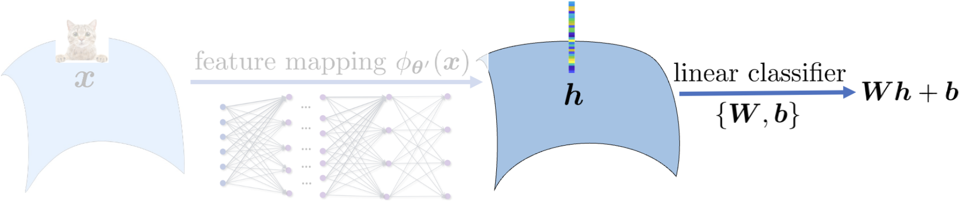
Simplification: Unconstrained Features II



- Neural Collapse is about the classifier W and the features $\phi_{\theta'}(x_{k,i})$
- To understand NC, we treat the features $h_{k,i} = \phi_{\theta'}(x_{k,i})$ as free optimization variables (unconstrained features model [Mixon et al.'21])

$$\min_{\{h_{k,i}\}, W, b} \frac{1}{Kn} \sum_{k=1}^{K} \sum_{i=1}^{n} \mathcal{L}_{\mathrm{CE}}\left(W h_{k,i} + b, y_k\right) + \lambda \|(\{h_{k,i}\}, W, b)\|_F^2$$

Simplification: Unconstrained Features III



$$\min_{\{h_{k,i}\}, W, b} \frac{1}{Kn} \sum_{k=1}^{K} \sum_{i=1}^{n} \mathcal{L}_{\mathrm{CE}}\left(W h_{k,i} + b, y_k\right) + \lambda \|(\{h_{k,i}\}, W, b)\|_F^2$$

- **Validity**: Modern networks are highly over-parameterized, that can approximate any point in the feature space
- Also called layer-peeled model and has been studied recently to understand NC
- We will show such simplification preserves the core properties of last-layer classifiers and features—the NC phenomenon

Simplification: Unconstrained Features IV

[Lu et al.'20] study the following one-example-per class model

$$\min_{\{\boldsymbol{h}_k\}} \frac{1}{K} \sum_{k=1}^{K} \mathcal{L}_{\mathrm{CE}}\left(\boldsymbol{h}_k, \boldsymbol{y}_k\right), \text{ s.t.} \|\boldsymbol{h}_{k,i}\|_2 = 1$$

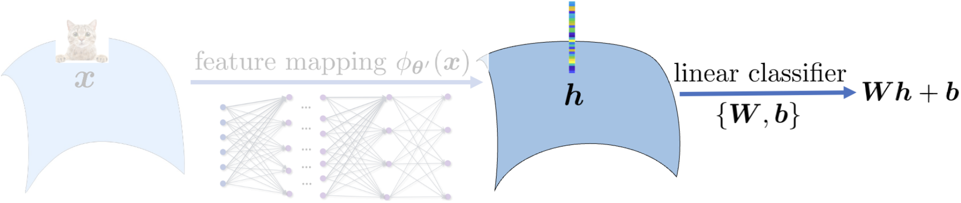
[E et al.'20, Fang et al.'21, Gral et al.'21, etc.] study constrained formulation

$$\min_{\{\boldsymbol{h}_k\}, \boldsymbol{W}} \frac{1}{Kn} \sum_{k=1}^{K} \sum_{i=1}^{n} \mathcal{L}_{\mathrm{CE}}\left(\boldsymbol{W}\boldsymbol{h}_{k,i}, \boldsymbol{y}_k\right), \text{ s.t. } \|\boldsymbol{W}\|_F \leq 1, \|\boldsymbol{h}_{k,i}\|_2 \leq 1$$

These work show that any global solution has NC, but

- What about local minima/saddle points?
- The constrained formulations are not aligned with practice

Geometric Analysis for Unconstrained Features Model I



$$\min_{\{\boldsymbol{h}_{k,i}\},\boldsymbol{W},\boldsymbol{b}} \frac{1}{Kn}\sum_{k=1}^{K}\sum_{i=1}^{n}\mathcal{L}_{\mathrm{CE}}\left(\boldsymbol{W}\boldsymbol{h}_{k,i}+\boldsymbol{b},\boldsymbol{y}_k\right)+\lambda\|(\{\boldsymbol{h}_{k,i}\},\boldsymbol{W},\boldsymbol{b})\|_F^2$$

- Closely related to the matrix factorization problem in Lecture 3: bilinear form $\boldsymbol{W}\boldsymbol{h}_{k,i}$
- We will study its global/local minima and saddle points

Geometric Analysis for Unconstrained Features Model II

$$\min_{\{\boldsymbol{h}_{k,i}\},\boldsymbol{W},\boldsymbol{b}} \frac{1}{Kn}\sum_{k=1}^{K}\sum_{i=1}^{n}\mathcal{L}_{\mathrm{CE}}\left(\boldsymbol{W}\boldsymbol{h}_{k,i}+\boldsymbol{b},\boldsymbol{y}_k\right)+\lambda\|(\{\boldsymbol{h}_{k,i}\},\boldsymbol{W},\boldsymbol{b})\|_F^2$$

Theorem (global optimality) [Zhu et al. 2021] Let feature dim. $d \geq$ #class $K-1$. Then any global solution $(\{\boldsymbol{h}_{k,i}^\star, \boldsymbol{W}^\star, \boldsymbol{b}^\star\})$ must satisfy NC: $\boldsymbol{b}^\star = 0$ and

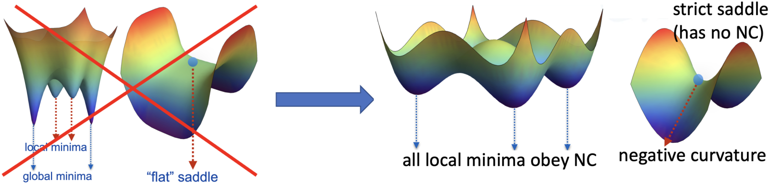
$$\underbrace{\boldsymbol{h}_{k,i}^\star = \overline{\boldsymbol{h}}_k^\star,}_{\text{NC1}} \quad \underbrace{\frac{\langle \overline{\boldsymbol{h}}_k^\star, \overline{\boldsymbol{h}}_{k'}^\star\rangle}{\|\overline{\boldsymbol{h}}_k^\star\|\|\overline{\boldsymbol{h}}_{k'}^\star\|} = \begin{cases} 1, & k=k' \\ -\frac{1}{K-1}, & k\neq k' \end{cases},}_{\text{NC2}} \quad \underbrace{\frac{\boldsymbol{w}^{k\star}}{\|\boldsymbol{w}^{k\star}\|} = \frac{\overline{\boldsymbol{h}}_k^\star}{\|\overline{\boldsymbol{h}}_k^\star\|}}_{\text{NC3}}$$

- $d \geq K-1$ is required to make K class-mean features equal angle and with cosine angle $-\frac{1}{K-1}$ (the largest possible) between each pair.

Geometric Analysis for Unconstrained Features Model III

$$\min_{\{\boldsymbol{h}_{k,i}\},\boldsymbol{W},\boldsymbol{b}} \frac{1}{Kn}\sum_{k=1}^{K}\sum_{i=1}^{n}\mathcal{L}_{\mathrm{CE}}\left(\boldsymbol{W}\boldsymbol{h}_{k,i}+\boldsymbol{b},\boldsymbol{y}_k\right)+\lambda\|(\{\boldsymbol{h}_{k,i}\},\boldsymbol{W},\boldsymbol{b})\|_F^2$$

Theorem (benign global landscape) [Zhu et al. 2021] Let feature dim. $d >$ #class K. Then the above objective function (i) has no spurious local minima, and (ii) any non-global critical point is a strict saddle with negative curvature. Conjecture: $d \geq K-1$ is sufficient.



General nonconvex problems | Our training problem

Geometric Analysis for Unconstrained Features Model IV

$$\min_{\{\boldsymbol{h}_{k,i}\},\boldsymbol{W},\boldsymbol{b}} \frac{1}{Kn}\sum_{k=1}^{K}\sum_{i=1}^{n}\mathcal{L}_{\mathrm{CE}}\big(\boldsymbol{W}\boldsymbol{h}_{k,i}+\boldsymbol{b},\boldsymbol{y}_k\big)+\lambda\|(\{\boldsymbol{h}_{k,i}\},\boldsymbol{W},\boldsymbol{b})\|_F^2 \quad \text{(NVX)}$$

Theorem (benign global landscape) [Zhu et al. 2021] Let feature dim. $d >$ #class K. Then the above objective function (i) has no spurious local minima, and (ii) any non-global critical point is a strict saddle with negative curvature.

- Proof idea: let $\boldsymbol{z}_{k,i} = \boldsymbol{W}\boldsymbol{h}_{k,i}$. Then (NVX) is equivalent to the following convex problem [Haeffele & Vidal'15, Li et al.'17, Ciliberto et al.'17]

$$\min_{\boldsymbol{Z},\boldsymbol{b}} \frac{1}{Kn}\sum_{k=1}^{K}\sum_{i=1}^{n}\mathcal{L}_{\mathrm{CE}}\big(\boldsymbol{z}_{k,i}+\boldsymbol{b},\boldsymbol{y}_k\big)+\lambda\|\boldsymbol{Z}\|_*+\lambda\|\boldsymbol{b}\|_2^2 \quad \text{(CVX)}$$

where $\|\cdot\|_*$ is the nuclear norm (sum of singular values).

Geometric Analysis for Unconstrained Features Model V

$$\min_{\{\boldsymbol{h}_{k,i}\},\boldsymbol{W},\boldsymbol{b}} \frac{1}{Kn}\sum_{k=1}^{K}\sum_{i=1}^{n}\mathcal{L}_{\mathrm{CE}}\left(\boldsymbol{W}\boldsymbol{h}_{k,i}+\boldsymbol{b},\boldsymbol{y}_k\right)+\lambda\|(\{\boldsymbol{h}_{k,i}\},\boldsymbol{W},\boldsymbol{b})\|_F^2 \quad \text{(NVX)}$$

$$\min_{\boldsymbol{Z},\boldsymbol{b}} \frac{1}{Kn}\sum_{k=1}^{K}\sum_{i=1}^{n}\mathcal{L}_{\mathrm{CE}}\left(\boldsymbol{z}_{k,i}+\boldsymbol{b},\boldsymbol{y}_k\right)+\lambda\|\boldsymbol{Z}\|_*+\lambda\|\boldsymbol{b}\|_2^2 \quad \text{(CVX)}$$

- Step 1: (NVX) and (CVX) have the "same" global solutions: if $(\boldsymbol{H}^\star,\boldsymbol{W}^\star,\boldsymbol{b}^\star)$ is a global solution of (NVX), then $(\boldsymbol{W}^\star\boldsymbol{H}^\star,\boldsymbol{b}^\star)$ is a global solution of (CVX); vice versa.

$$\text{variational form } \|\boldsymbol{Z}\|_* = \min_{\boldsymbol{Z}=\boldsymbol{W}\boldsymbol{H}} \frac{1}{2}\left(\|\boldsymbol{W}\|_F^2+\|\boldsymbol{H}\|_F^2\right)$$

Geometric Analysis for Unconstrained Features Model VI

$$\min_{\{\boldsymbol{h}_{k,i}\},\boldsymbol{W},\boldsymbol{b}} \frac{1}{Kn}\sum_{k=1}^{K}\sum_{i=1}^{n}\mathcal{L}_{\mathrm{CE}}\big(\boldsymbol{W}\boldsymbol{h}_{k,i}+\boldsymbol{b},\boldsymbol{y}_k\big)+\lambda\|(\{\boldsymbol{h}_{k,i}\},\boldsymbol{W},\boldsymbol{b})\|_F^2 \quad \text{(NVX)}$$

$$\min_{\boldsymbol{Z},\boldsymbol{b}} \frac{1}{Kn}\sum_{k=1}^{K}\sum_{i=1}^{n}\mathcal{L}_{\mathrm{CE}}\big(\boldsymbol{z}_{k,i}+\boldsymbol{b},\boldsymbol{y}_k\big)+\lambda\|\boldsymbol{Z}\|_*+\lambda\|\boldsymbol{b}\|_2^2 \quad \text{(CVX)}$$

- Step 2: if $(\boldsymbol{H},\boldsymbol{W},\boldsymbol{b})$ is a critical point but not a global min of (NVX)
  - $(\boldsymbol{Z},\boldsymbol{b})$ with $\boldsymbol{Z}=\boldsymbol{W}\boldsymbol{H}$ is not a critical point to (CVX)
  - $(\boldsymbol{Z},\boldsymbol{b})$ does not satisfy the first-order optimality condition of (CVX)
  - Exploiting this, we show the Hessian at $(\boldsymbol{H},\boldsymbol{W},\boldsymbol{b})$ has a negative eigenvalue, i.e., it is a strict saddle of (NVX)

Geometric Analysis for Unconstrained Features Model VII

$$\min_{\{\boldsymbol{h}_{k,i}\},\boldsymbol{W},\boldsymbol{b}} \frac{1}{Kn}\sum_{k=1}^{K}\sum_{i=1}^{n}\mathcal{L}_{\mathrm{CE}}\left(\boldsymbol{W}\boldsymbol{h}_{k,i}+\boldsymbol{b},\boldsymbol{y}_k\right)+\lambda\|(\{\boldsymbol{h}_{k,i}\},\boldsymbol{W},\boldsymbol{b})\|_F^2 \quad \text{(NVX)}$$

$$\min_{\boldsymbol{Z},\boldsymbol{b}} \frac{1}{Kn}\sum_{k=1}^{K}\sum_{i=1}^{n}\mathcal{L}_{\mathrm{CE}}\left(\boldsymbol{z}_{k,i}+\boldsymbol{b},\boldsymbol{y}_k\right)+\lambda\|\boldsymbol{Z}\|_*+\lambda\|\boldsymbol{b}\|_2^2 \quad \text{(CVX)}$$

- Step 1: (NVX) and (CVX) have the "same" global solutions.
- Step 2: if $(\boldsymbol{H},\boldsymbol{W},\boldsymbol{b})$ is a critical point but not a global min of (NVX)
  - the Hessian at $(\boldsymbol{H},\boldsymbol{W},\boldsymbol{b})$ has a negative eigenvalue, i.e., it is a strict saddle
- Step 2 holds for any non-global critical point $\Rightarrow$ (NVX) has benign global landscape (no spurious local minima & strict saddle function)

Geometric Analysis for Unconstrained Features Model VIII

$$\min_{\{\boldsymbol{h}_{k,i}\},\boldsymbol{W},\boldsymbol{b}} \frac{1}{Kn}\sum_{k=1}^{K}\sum_{i=1}^{n}\mathcal{L}_{\mathrm{CE}}\big(\boldsymbol{W}\boldsymbol{h}_{k,i}+\boldsymbol{b},\boldsymbol{y}_k\big)+\lambda\|(\{\boldsymbol{h}_{k,i}\},\boldsymbol{W},\boldsymbol{b})\|_F^2$$

Theorem (global optimality & benign global landscape) Let feature dim. $d >$ #class K.

- Any global solution $(\{\boldsymbol{h}_{k,i}^\star,\boldsymbol{W}^\star,\boldsymbol{b}^\star\})$ obeys Neural Collapse.
- The objective function (i) has no spurious local minima, and (ii) any non-global critical point is a strict saddle with negative curvature.

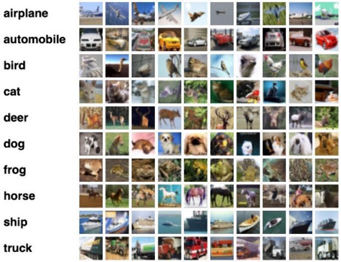
Message. Iterative algorithms such as (stochastic) gradient descent always learns Neural Collapse features and classifiers.

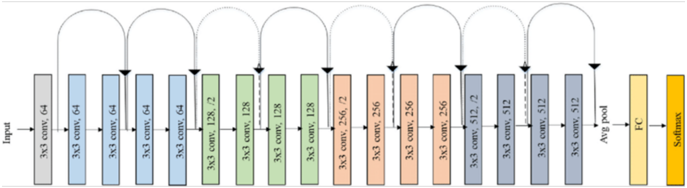
Experiments on Practical Neural Networks

Conduct experiments with **practical networks** to verify our findings on Unconstrained Features Model

Use a Residual Neural Network (ResNet) on CIFAR-10 Dataset:

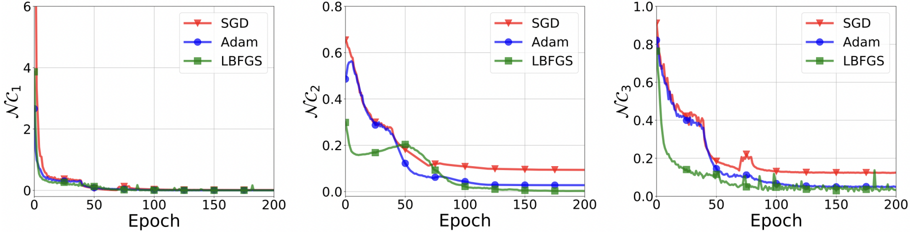
- $K = 10$ classes
- 50K training images
- 10K testing images





Experiments: NC is algorithm independent

ResNet18 on CIFAR-10 with **different training algorithms**

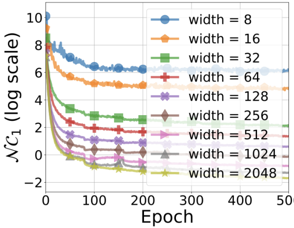


Within-Class Variability (NC1) | Between-Class Separation (NC2) | Self-Duality Collapse (NC3)

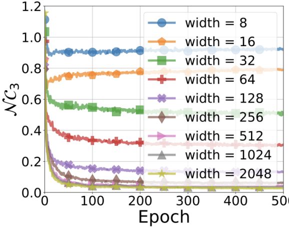
- The smaller the quantities, the severer NC
- NC across **different training algorithms**

Experiments: NC Occurs on Random Labels/Inputs

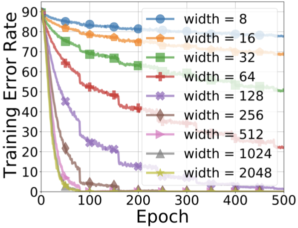
CIFAR-10 with **random** labels, multi-layer perceptron (MLP) with **varying network widths**



Within-Class Variability (NC1)



Self-Duality Collapse (NC2)



Training Error

- **Validity of unconstrained features model**: Learn NC last-layer features and classifiers for any inputs
- The network memorizes training data in a very special way: NC
- We observe similar results on **random inputs (random pixels)**

Exploit NC

Experiments in [Papyan, Han Donoho] shows NC leads to better

- Generalization performance
- Robustness

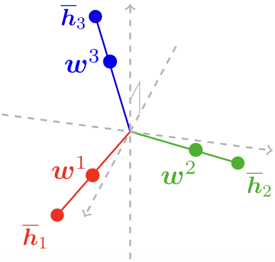
We can also exploit NC for

- Improving training efficiency & memory cost (covered later)
- Understanding the effect of loss functions (covered later)
- Understanding transferability
- etc.

Exploit NC for Improving Training & Memory I

NC is prevalent, and classifier always converges to a Simplex ETF

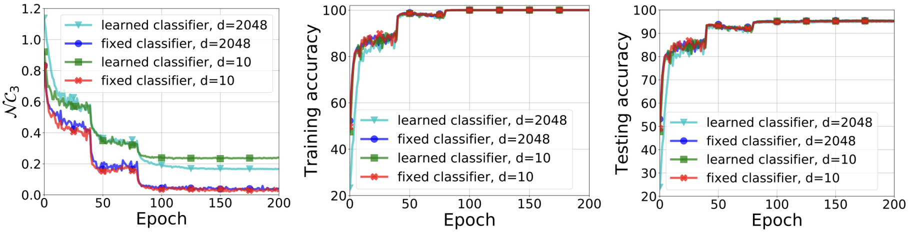
- **Implication 1: No need to learn the classifier** [Hoffer et al. 2018]
    - Just fix it as a Simplex ETF
    - Save **8%, 12%, and 53%** parameters for ResNet50, DenseNet169, and ShuffleNet!
- **Implication 2: No need of large feature dimension d**
    - Just use feature dim. $d = \#\text{class } K$ (e.g., $d = 10$ for CIFAR-10)
    - Further saves **21% and 4.5%** parameters for ResNet18 and ResNet50!



Exploit NC for Improving Training & Memory II

ResNet50 on CIFAR-10 with different settings

- Learned classifier (default) VS fixed classifier as a simplex ETF
- Feature dim $d = 2048$ (default) VS $d = 10$

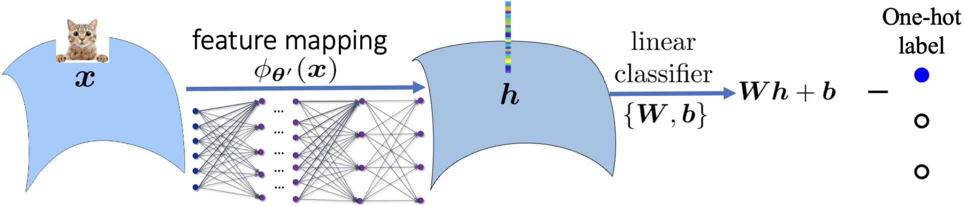


Self-Duality Collapse (NC3) | Training Accuracy | Testing Accuracy

- Training with small dimensional features and fixed classifiers achieves on-par performance with large dimensional features and learned classifiers.

Is Cross-entropy Loss Essential?

Is cross-entropy loss essential to neural collapse?



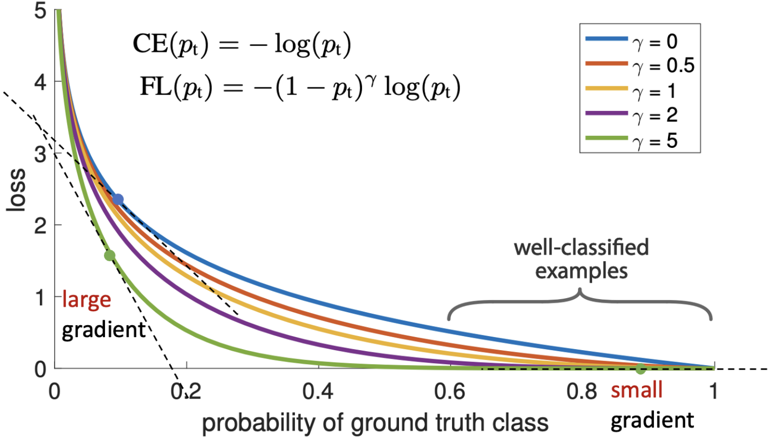
We can measure the mismatch between the network output and the one-hot label in many ways.

Various losses and tricks (e.g., label smoothing, focal loss) have been proposed to improve network training and performance[1]

[1] He et al., Bag of tricks for image classification with convolutional neural networks, CVPR'19.

Focal Loss (FL)

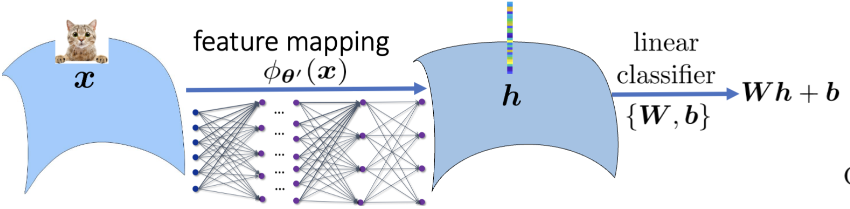
Focal loss puts more focus on hard, misclassified examples[2]



[2] Lin et al., Focal Loss for Dense Object Detection, CVPR'18.

Label Smoothing (LS)

Label smoothing replaces the hard label by a soft label [3]



Soft label

$$\begin{bmatrix} 1-\alpha \\ \alpha/2 \\ \alpha/2 \end{bmatrix}$$

$\text{CE}: \alpha = 0$

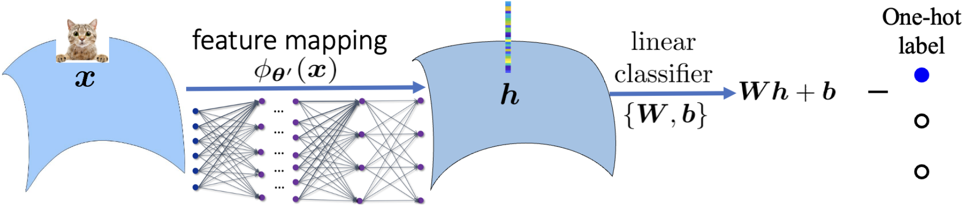
$$\text{Output: } Wh + b = \begin{bmatrix} 1 \\ 0 \\ -1 \end{bmatrix} \xrightarrow{\text{Softmax function}} \begin{bmatrix} 0.6 \\ 0.3 \\ 0.1 \end{bmatrix} \begin{array}{l} \text{Cat} \\ \text{Dog} \\ \text{Panda} \end{array} \begin{bmatrix} 1-\alpha \\ \alpha/2 \\ \alpha/2 \end{bmatrix}$$

Prediction   Target

$$\begin{aligned} \text{LS} = & -q(\text{Cat}) \cdot \log p(\text{Cat}) \\ & -q(\text{Dog}) \cdot \log p(\text{Dog}) \\ & -q(\text{Panda}) \cdot \log p(\text{Panda}) \\ = & -(1-\alpha)\log(0.6) \\ & -\frac{\alpha}{2}\log(0.3) \\ & -\frac{\alpha}{2}\log(0.1) \end{aligned}$$

[3] Szegedy et al., Rethinking the inception architecture for computer vision, CVPR'16. Muller, Kornblith, Hinton, When does label smoothing help?, NeurIPS'19.

Mean-squared Error (MSE) Loss?



$$\text{Output: } \boldsymbol{W}\boldsymbol{h} + \boldsymbol{b} = \begin{bmatrix} 1 \\ 0 \\ -1 \end{bmatrix} \begin{matrix} \text{Cat} \\ \text{Dog} \\ \text{Panda} \end{matrix} \quad \begin{bmatrix} 1 \\ 0 \\ 0 \end{bmatrix}$$

Prediction Target

$$\text{MSE:} = (1 - 1)^2 + (0 - 0)^2 + (-1 - 0)^2$$

Compared with CE, (rescaled) MSE loss produces on par/slightly worse results for computer vision tasks and on par/slightly better results for NLP tasks.[4]

[4] Hui & Belkin, Evaluation of neural architectures trained with square loss vs cross-entropy in classification tasks, ICLR 2021.

Are All Loses Created Equal?—A NC Perspective I

Do all these losses make difference?

We study them under the unconstrained feature model:

$$\min_{\{\boldsymbol{h}_{k,i}\},\boldsymbol{W},\boldsymbol{b}} \frac{1}{Kn}\sum_{k=1}^{K}\sum_{i=1}^{n}\mathcal{L}\big(\boldsymbol{W}\boldsymbol{h}_{k,i}+\boldsymbol{b},\boldsymbol{y}_k\big)+\lambda\|(\{\boldsymbol{h}_{k,i}\},\boldsymbol{W},\boldsymbol{b})\|_F^2$$

Theorem (informal) [Zhou et al.'22] With feature dim. $d >$ #class K, all the one-hot labeling based losses (e.g., CE, FL, LS, MSE) lead to (almost) the same NC features and classifiers [Han et al'21, Tirer & Bruner'22, Zhou'22].

Are All Loses Created Equal?—A NC Perspective II

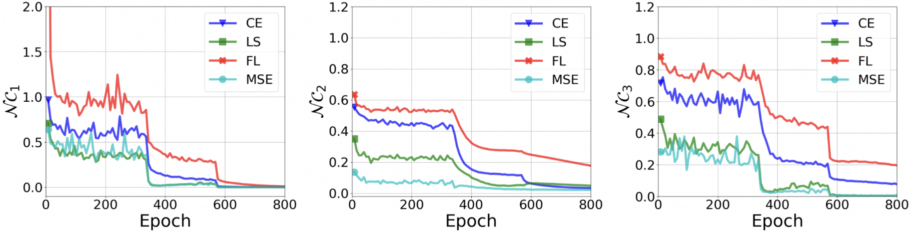
Theorem (informal) [Zhou et al.'22] With feature dim. $d >$ #class K, all the one-hot labeling based losses (e.g., CE, FL, LS, MSE) lead to (almost) the same NC features and classifiers [Han et al'21, Tirer & Bruner'22, Zhou'22].

Implication If network is large enough and trained longer enough

- All losses lead to largely identical features on training data—NC phenomena
- All losses lead to largely identical performance on test data (experiments in the following slides)

Are All Loses Created Equal?—A NC Perspective III

ResNet50 on CIFAR-10 with **different training losses**

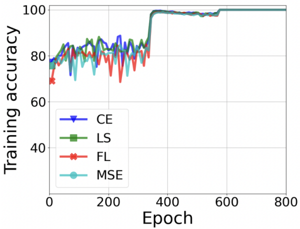


Within-Class Variability (NC1) Between-Class Separation (NC2) Self-Duality Collapse (NC3)

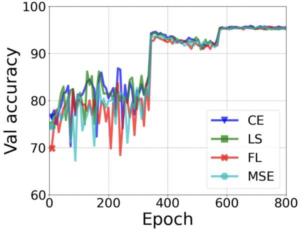
- The smaller the quantities, the severer NC
- NC across **different training losses**

Are All Loses Created Equal?—A NC Perspective IV

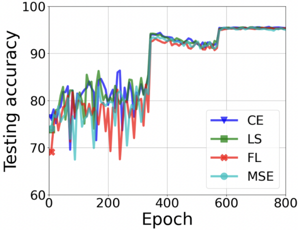
ResNet50 on CIFAR-10 with **different training losses**



Train accuracy



Validation accuracy

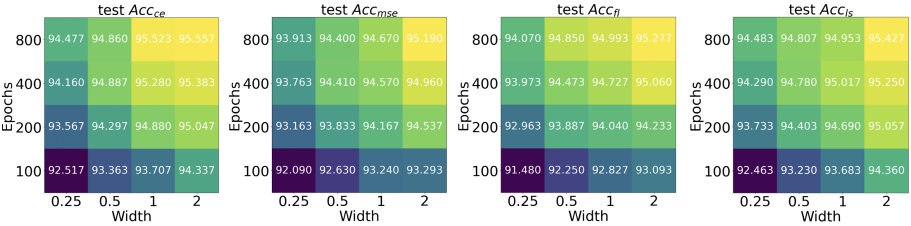


Testing accuracy

- All losses lead to largely identical performance on training, validation, and test data

Are All Loses Created Equal?—A NC Perspective V

ResNet50 (with different network widths and training epoches) on CIFAR-10 with **different training losses**



Cross-entropy | Mean-squared Error | Focal loss | Label smoothing

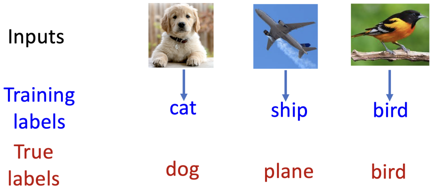
- If network is large enough and trained longer enough, all losses lead to largely identical performance on test data

Outline

NC → Overfitting to Corruptions!

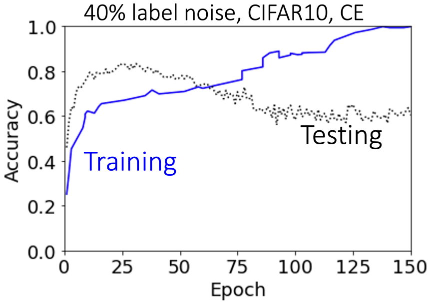
Label noise is common and often unavoidable

- Some proportion of the labels are incorrect (5-80%?)
- We don't know which labels are correct/incorrect



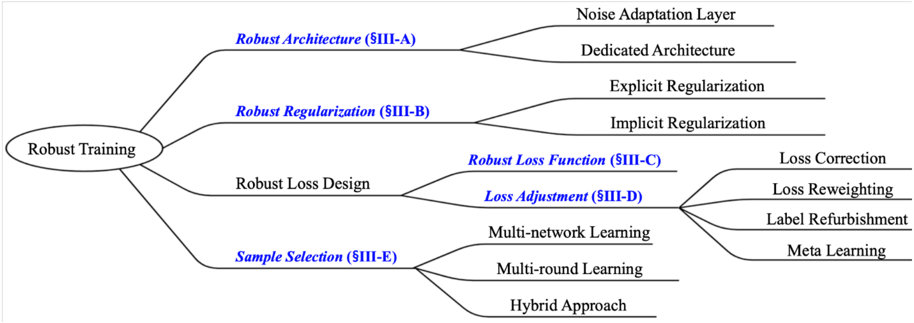
NC always happens

- Perfectly fits noisy labels (ovefitting)
- Can't predict well on new images



Prior Work on Robust Deep Learning for Noisy Labels

Various (heuristic or principled) methods have been proposed[5]



[5] Song et al., Learning from noisy labels with deep neural networks: A survey, IEEE TNNLS, 2022.

A Sparse Over-Parameterization (SOP) Method

We model the label noise and (hopefully) correct it. Only a fraction of the labels are corrupted (sparse), and the corruption in each label is also sparse

$$\begin{array}{r} \text{“cat”} \\ \text{“dog”} \\ \\ \\ \end{array} \underbrace{\begin{bmatrix} 0 \\ 1 \\ \vdots \\ 0 \end{bmatrix}}_{\text{Corrupted label}}^{\boldsymbol{y}} = \underbrace{\begin{bmatrix} 1 \\ 0 \\ \vdots \\ 0 \end{bmatrix}}_{\text{True label}}^{f(\boldsymbol{x};\boldsymbol{\theta})} + \underbrace{\begin{bmatrix} -1 \\ 1 \\ \vdots \\ 0 \end{bmatrix}}_{\text{Sparse noise}}^{\boldsymbol{s}}$$

Lecture 1 introduced principled methods for dealing with sparse corruption in compressive sensing, robust PCA[6]

[6] Candes & Tao, Decoding by linear programming, TIT 2005.
Wright et al., Robust face recognition via sparse representation, TPAMI, 2008.
Candes et al., Robust principal component analysis? JACM, 2011.

A Sparse Over-Parameterization (SOP) Method

Our approach:[7] minimize the distance between $\boldsymbol{y}$ and $f(\boldsymbol{\theta}; \boldsymbol{x}) + \boldsymbol{s}$

$$\min_{\boldsymbol{\theta}, \boldsymbol{u}_i, \boldsymbol{v}_i} \frac{1}{N} \sum_{i=1}^{N} \mathcal{L}_{\mathrm{CE}}\big(f(\boldsymbol{x}_i; \boldsymbol{\theta}) + \underbrace{\boldsymbol{u}_i \odot \boldsymbol{u}_i - \boldsymbol{v}_i \odot \boldsymbol{v}_i}_{\text{over-parameterize } \boldsymbol{s}_i \text{ to promote sparsity}}, \boldsymbol{y}_i\big)$$

Here the over-parameterization $\boldsymbol{u}_i \odot \boldsymbol{u}_i - \boldsymbol{v}_i \odot \boldsymbol{v}_i$ introduces implicit algorithmic regularization [Vaskevicius et al.'19, Zhao et al.'19]

$$\text{variational form } \|\boldsymbol{s}\|_1 = \min_{\boldsymbol{s} = \boldsymbol{u} \odot \boldsymbol{u} - \boldsymbol{v} \odot \boldsymbol{v}} \frac{1}{2}(\|\boldsymbol{u}\|^2 + \|\boldsymbol{v}\|^2)$$

[7] Liu, Zhu, Qu, You, Robust Training under Label Noise by Over-parameterization, ICML'22.

A Sparse Over-Parameterization (SOP) Method

Our approach:[7] minimize the distance between $\boldsymbol{y}$ and $f(\boldsymbol{\theta}; \boldsymbol{x}) + \boldsymbol{s}$

$$\min_{\boldsymbol{\theta}, \boldsymbol{u}_i, \boldsymbol{v}_i} \frac{1}{N} \sum_{i=1}^{N} \mathcal{L}_{\mathrm{CE}}\big(f(\boldsymbol{x}_i; \boldsymbol{\theta}) + \underbrace{\boldsymbol{u}_i \odot \boldsymbol{u}_i - \boldsymbol{v}_i \odot \boldsymbol{v}_i}_{\text{over-parameterize } \boldsymbol{s}_i \text{ to promote sparsity}}, \boldsymbol{y}_i)$$

Here the over-parameterization $\boldsymbol{u}_i \odot \boldsymbol{u}_i - \boldsymbol{v}_i \odot \boldsymbol{v}_i$ introduces implicit algorithmic regularization [Vaskevicius et al.'19, Zhao et al.'19]

$$\text{variational form } \|\boldsymbol{s}\|_1 = \min_{\boldsymbol{s} = \boldsymbol{u} \odot \boldsymbol{u} - \boldsymbol{v} \odot \boldsymbol{v}} \frac{1}{2}(\|\boldsymbol{u}\|^2 + \|\boldsymbol{v}\|^2)$$

Why not use explicit regularization?

$$\min_{\boldsymbol{\theta}, \{\boldsymbol{s}_i\}} \frac{1}{N} \sum_{i=1}^{N} \underbrace{\mathcal{L}_{\mathrm{CE}}\big(f(\boldsymbol{x}_i; \boldsymbol{\Theta}) + \boldsymbol{s}_i, \boldsymbol{y}_i\big)}_{\to 0} + \underbrace{\lambda \|\boldsymbol{s}_i\|_1}_{\to 0}$$

[7] Liu, Zhu, Qu, You, Robust Training under Label Noise by Over-parameterization, ICML'22.

A Sparse Over-Parameterization (SOP) Method

A simple model: assume $f(\boldsymbol{x};\boldsymbol{\theta})$ is a scalar function and can be approximated by first-order Taylor expansion

$$f(\boldsymbol{x};\boldsymbol{\theta}) \approx f(\boldsymbol{x};\boldsymbol{\theta}_0) + \langle \nabla f(\boldsymbol{x};\boldsymbol{\theta}_0), \boldsymbol{\theta} - \boldsymbol{\theta}_0 \rangle$$

A Sparse Over-Parameterization (SOP) Method

A simple model: assume $f(\boldsymbol{x}; \boldsymbol{\theta})$ is a scalar function and can be approximated by first-order Taylor expansion

$$f(\boldsymbol{x}; \boldsymbol{\theta}) \approx f(\boldsymbol{x}; \boldsymbol{\theta}_0) + \langle \nabla f(\boldsymbol{x}; \boldsymbol{\theta}_0), \boldsymbol{\theta} - \boldsymbol{\theta}_0 \rangle$$

WLOG, assume $f(\boldsymbol{x}; \boldsymbol{\theta}_0) + \langle \nabla f(\boldsymbol{x}; \boldsymbol{\theta}_0), \boldsymbol{\theta}_0 \rangle = 0$. For N training samples,

$$\begin{bmatrix} f(\boldsymbol{x}_1; \boldsymbol{\theta}) \\ \vdots \\ f(\boldsymbol{x}_N; \boldsymbol{\theta}) \end{bmatrix} \approx \begin{bmatrix} \nabla f(\boldsymbol{x}_1; \boldsymbol{\theta}_0)^\top \\ \vdots \\ \nabla f(\boldsymbol{x}_N; \boldsymbol{\theta}_0)^\top \end{bmatrix} \boldsymbol{\theta} = \boldsymbol{J} \cdot \boldsymbol{\theta}$$

A Sparse Over-Parameterization (SOP) Method

A simple model: assume $f(\boldsymbol{x};\boldsymbol{\theta})$ is a scalar function and can be approximated by first-order Taylor expansion

$$f(\boldsymbol{x};\boldsymbol{\theta}) \approx f(\boldsymbol{x};\boldsymbol{\theta}_0) + \langle \nabla f(\boldsymbol{x};\boldsymbol{\theta}_0), \boldsymbol{\theta} - \boldsymbol{\theta}_0 \rangle$$

WLOG, assume $f(\boldsymbol{x};\boldsymbol{\theta}_0) + \langle \nabla f(\boldsymbol{x};\boldsymbol{\theta}_0), \boldsymbol{\theta}_0 \rangle = 0$. For N training samples,

$$\begin{bmatrix} f(\boldsymbol{x}_1;\boldsymbol{\theta}) \\ \vdots \\ f(\boldsymbol{x}_N;\boldsymbol{\theta}) \end{bmatrix} \approx \begin{bmatrix} \nabla f(\boldsymbol{x}_1;\boldsymbol{\theta}_0)^\top \\ \vdots \\ \nabla f(\boldsymbol{x}_N;\boldsymbol{\theta}_0)^\top \end{bmatrix} \boldsymbol{\theta} = \boldsymbol{J} \cdot \boldsymbol{\theta}$$

This leads to the following corrupted observation problem

$$\boldsymbol{y} = \boldsymbol{J} \cdot \boldsymbol{\theta}_\star + \boldsymbol{s}_\star$$

where $\boldsymbol{\theta}_\star$ is the underlying groundtruth parameter, and $\boldsymbol{s}_\star$ is sparse.

A Sparse Over-Parameterization (SOP) Method

We over-parameterize the sparse noise by $\boldsymbol{u} \odot \boldsymbol{u} - \boldsymbol{v} \odot \boldsymbol{v}$ and solve

$$\min_{\boldsymbol{\theta}, \boldsymbol{u}, \boldsymbol{v}} g(\boldsymbol{\theta}, \boldsymbol{u}, \boldsymbol{v}) = \frac{1}{2} \| \boldsymbol{J} \cdot \boldsymbol{\theta} + \boldsymbol{u} \odot \boldsymbol{u} - \boldsymbol{v} \odot \boldsymbol{v} - \boldsymbol{y} \|_2^2$$

using gradient descent with *discrepant learning rates*

$$\boldsymbol{\theta}_{t+1} = \boldsymbol{\theta}_t - \mu \nabla_{\boldsymbol{\theta}} g(\boldsymbol{\theta}_t, \boldsymbol{u}_t, \boldsymbol{v}_t)$$

$$\begin{bmatrix} \boldsymbol{u}_{t+1} \\ \boldsymbol{v}_{t+1} \end{bmatrix} = \begin{bmatrix} \boldsymbol{u}_t \\ \boldsymbol{v}_t \end{bmatrix} - \alpha \mu \begin{bmatrix} \nabla_{\boldsymbol{u}} g(\boldsymbol{\theta}_t, \boldsymbol{u}_t, \boldsymbol{v}_t) \\ \nabla_{\boldsymbol{v}} g(\boldsymbol{\theta}_t, \boldsymbol{u}_t, \boldsymbol{v}_t) \end{bmatrix}$$

A Sparse Over-Parameterization (SOP) Method

We over-parameterize the sparse noise by $\boldsymbol{u} \odot \boldsymbol{u} - \boldsymbol{v} \odot \boldsymbol{v}$ and solve

$$\min_{\boldsymbol{\theta}, \boldsymbol{u}, \boldsymbol{v}} g(\boldsymbol{\theta}, \boldsymbol{u}, \boldsymbol{v}) = \frac{1}{2} \| \boldsymbol{J} \cdot \boldsymbol{\theta} + \boldsymbol{u} \odot \boldsymbol{u} - \boldsymbol{v} \odot \boldsymbol{v} - \boldsymbol{y} \|_2^2$$

using gradient descent with *discrepant learning rates*

$$\boldsymbol{\theta}_{t+1} = \boldsymbol{\theta}_t - \mu \nabla_{\boldsymbol{\theta}} g(\boldsymbol{\theta}_t, \boldsymbol{u}_t, \boldsymbol{v}_t)$$

$$\begin{bmatrix} \boldsymbol{u}_{t+1} \\ \boldsymbol{v}_{t+1} \end{bmatrix} = \begin{bmatrix} \boldsymbol{u}_t \\ \boldsymbol{v}_t \end{bmatrix} - \alpha \mu \begin{bmatrix} \nabla_{\boldsymbol{u}} g(\boldsymbol{\theta}_t, \boldsymbol{u}_t, \boldsymbol{v}_t) \\ \nabla_{\boldsymbol{v}} g(\boldsymbol{\theta}_t, \boldsymbol{u}_t, \boldsymbol{v}_t) \end{bmatrix}$$

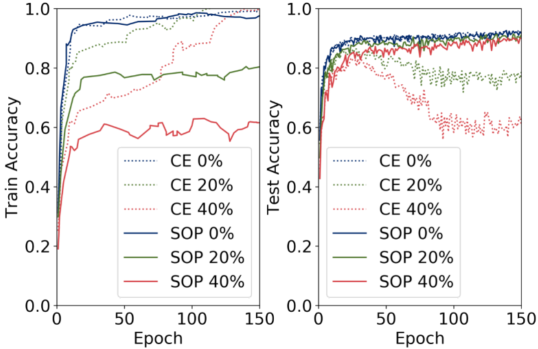
Theorem (informal) If gradient descent with infinitesimallly small initialization and step size μ converges to $(\widehat{\boldsymbol{\theta}}, \widehat{\boldsymbol{u}}, \widehat{\boldsymbol{v}})$, then $(\widehat{\boldsymbol{\theta}}, \widehat{\boldsymbol{u}} \odot \widehat{\boldsymbol{u}} - \widehat{\boldsymbol{v}} \odot \widehat{\boldsymbol{v}})$ is an optimal solution to the following convex problem

$$\min_{\boldsymbol{\theta}, \boldsymbol{s}} \frac{1}{2} \|\boldsymbol{\theta}\|_2^2 + \frac{1}{\alpha} \|\boldsymbol{s}\|_1, \text{ s.t. } \boldsymbol{y} = \boldsymbol{J} \cdot \boldsymbol{\theta} + \boldsymbol{s}$$

Exactly recover $(\boldsymbol{\theta}_\star, \boldsymbol{s}_\star)$ when $\boldsymbol{J}$ is incoherent [Candes & Tao'05].

A Sparse Over-Parameterization (SOP) Method

$\{0\%, 20\%, 40\%\}$ percent of labels for CIFAR-10 training data are randomly flipped uniformly to another class. Use ResNet34.

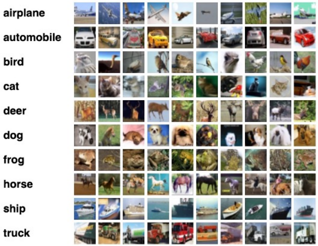


SOP trains a deep image classification networks without overfitting to wrong labels and obtain better generalization performance

SOP on CIFAR-10 with human annotated noisy labels

CIFAR-10N: provide CIFAR-10 with human annotated noisy labels[8]

- Annotated by 747 independent workers
- Provide 5 noisy label sets for CIFAR-10 train images:
- **Random** $i = 1, 2, 3$: the i-th submitted label for each image;
- **Aggregate**: aggregation of three noisy labels by majority voting
- **Worst**: label set with the highest noise rate



Label Set	CIFAR-10N Aggregate	CIFAR-10N Random 1	CIFAR-10N Random 2	CIFAR-10N Random 3	CIFAR-10N Worst
Noise Rate	9.03%	17.23%	18.12%	17.64%	40.21%

[8] Wei et al., Learning with noisy labels revisited: A study using real-world human annotations, ICLR 2022.

SOP on CIFAR-10 with human annotated noisy labels

Method	CIFAR-10N					
	Clean	*Aggregate*	*Random 1*	*Random 2*	*Random 3*	*Worst*
CE (Standard)	92.92 ± 0.11	87.77 ± 0.38	85.02 ± 0.65	86.46 ± 1.79	85.16 ± 0.61	77.69 ± 1.55
Forward T (Patrini et al., 2017)	93.02 ± 0.12	88.24 ± 0.22	86.88 ± 0.50	86.14 ± 0.24	87.04 ± 0.35	79.79 ± 0.46
Backward T (Patrini et al., 2017)	93.10 ± 0.05	88.13 ± 0.29	87.14 ± 0.34	86.28 ± 0.80	86.86 ± 0.41	77.61 ± 1.05
GCE (Zhang & Sabuncu, 2018)	92.83 ± 0.16	87.85 ± 0.70	87.61 ± 0.28	87.70 ± 0.56	87.58 ± 0.29	80.66 ± 0.35
Co-teaching (Han et al., 2018)	93.35 ± 0.14	91.20 ± 0.13	90.33 ± 0.13	90.30 ± 0.17	90.15 ± 0.18	83.83 ± 0.13
Co-teaching+ (Yu et al., 2019)	92.41 ± 0.20	90.61 ± 0.22	89.70 ± 0.27	89.47 ± 0.18	89.54 ± 0.22	83.26 ± 0.17
T-Revision (Xia et al., 2019)	93.35 ± 0.23	88.52 ± 0.17	88.33 ± 0.32	87.71 ± 1.02	87.79 ± 0.67	80.48 ± 1.20
Peer Loss (Liu & Guo, 2020)	93.99 ± 0.13	90.75 ± 0.25	89.06 ± 0.11	88.76 ± 0.19	88.57 ± 0.09	82.00 ± 0.60
ELR (Liu et al., 2020)	93.45 ± 0.65	92.38 ± 0.64	91.46 ± 0.38	91.61 ± 0.16	91.41 ± 0.44	83.58 ± 1.13
ELR+ (Liu et al., 2020)	$\mathbf{95.39 \pm 0.05}$	94.83 ± 0.10	94.43 ± 0.41	94.20 ± 0.24	94.34 ± 0.22	91.09 ± 1.60
Positive-LS (Lukasik et al., 2020)	94.77 ± 0.17	91.57 ± 0.07	89.80 ± 0.28	89.35 ± 0.33	89.82 ± 0.14	82.76 ± 0.53
F-Div (Wei & Liu, 2020)	94.88 ± 0.12	91.64 ± 0.34	89.70 ± 0.40	89.79 ± 0.12	89.55 ± 0.49	82.53 ± 0.52
Divide-Mix (Li et al., 2020)	$\mathbf{95.37 \pm 0.14}$	$\mathbf{95.01 \pm 0.71}$	$\mathbf{95.16 \pm 0.19}$	$\mathbf{95.23 \pm 0.07}$	$\mathbf{95.21 \pm 0.14}$	$\mathbf{92.56 \pm 0.42}$
Negative-LS (Wei et al., 2021)	$\mathbf{94.92 \pm 0.25}$	91.97 ± 0.46	90.29 ± 0.32	90.37 ± 0.12	90.13 ± 0.19	82.99 ± 0.36
JoCoR (Wei et al., 2020)	93.40 ± 0.24	91.44 ± 0.05	90.30 ± 0.20	90.21 ± 0.19	90.11 ± 0.21	83.37 ± 0.30
CORES^2 (Cheng et al., 2021)	93.43 ± 0.24	91.23 ± 0.11	89.66 ± 0.32	89.91 ± 0.45	89.79 ± 0.50	83.60 ± 0.53
CORES* (Cheng et al., 2021)	94.16 ± 0.11	$\mathbf{95.25 \pm 0.09}$	94.45 ± 0.14	94.88 ± 0.31	94.74 ± 0.03	91.66 ± 0.09
VolMinNet (Li et al., 2021)	92.14 ± 0.30	89.70 ± 0.21	88.30 ± 0.12	88.27 ± 0.09	88.19 ± 0.41	80.53 ± 0.20
CAL (Zhu et al., 2021a)	94.50 ± 0.31	91.97 ± 0.32	90.93 ± 0.31	90.75 ± 0.30	90.74 ± 0.24	85.36 ± 0.16
PES (Semi) (Bai et al., 2021)	94.76 ± 0.2	94.66 ± 0.18	$\mathbf{95.06 \pm 0.15}$	$\mathbf{95.19 \pm 0.23}$	$\mathbf{95.22 \pm 0.13}$	$\mathbf{92.68 \pm 0.22}$
SOP (Liu et al., 2022)	N/A	$\mathbf{95.61 \pm 0.13}$	$\mathbf{95.28 \pm 0.13}$	$\mathbf{95.31 \pm 0.10}$	$\mathbf{95.39 \pm 0.11}$	$\mathbf{93.24 \pm 0.21}$

Sparse modeling gives super performance again label noise[9]

[9] Wei et al., Learning with noisy labels revisited: A study using real-world human annotations, ICLR 2022.
Liu, Zhu, Qu, You, Robust Training under Label Noise by Over-parameterization, ICML'22.

Conclusion and Coming Attractions

Learning common deep networks for low-dim structure

- **Low-dimensional data**: understand resource tradeoffs between data structure and network architecture
- **Low-dimensional features**: understand low-dim. features (NC) learned in deep classifiers trained with one-hot labeling based losses
- **Robust training**: Exploit low-dim structure in the label noise to improve training robustness

Next lecture: New approach for learning diverse and discriminative features (beyond NC).
Designing deep network architectures for low-dimensional structures

Thank You! Questions?

Figure Credits I

- Slide 3: Dictionary learning figures from [Mairal, Elad, and Sapiro 2008]
- Slide 4: ImageNet classes from paperswithcode.com; AlexNet architecture: [Krizhevsky et al. 2012]; ResNet architecture: [He et al. 2015];
- Slide 5: ImageNet top1 from paperswithcode.com; DALL-E 1 and 2 from https://openai.com/blog/dall-e/ and https://openai.com/dall-e-2/
- Slide 7: Right image from https://www.cityscapes-dataset.com/dataset-overview/
- Slide 8: Hairbrushes from https://objectnet.dev/download.html
- Slide 9: Illumination figure from [Basri and Jacobs 2003]
- Slide 13: Left figure from [Krizhevsky et al. 2012]; right from https://openai.com/blog/microscope/;