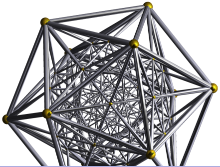
ICASSP 2023 Short Course

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

Lecture 5: Low-dimensional Representations in Deep Networks I

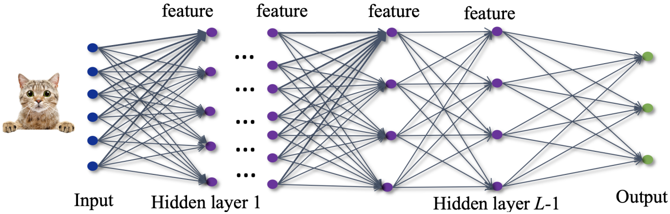
Sam Buchanan, Yi Ma, Qing Qu
Atlas Wang, John Wright, Yuqian Zhang, Zhihui Zhu

June 08, 2023



Recap: Deep Representation Learning

- A typical deep neural network has multi-layered structure

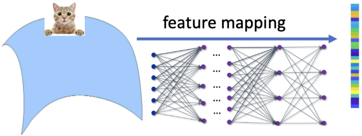


- Representation/feature: there is no (consensus) formal definition
  - any function of the input (to enable learning algorithms to better understand and make predictions)
- Today's lecture: what are the representations learned within DNNs?
  - challenge: high dimensionality; no simple criteria for good/bad features

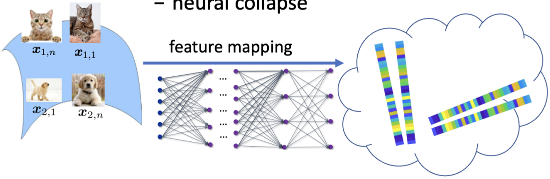
Focus: Geometrization of Learned Representations

- We will characterize different properties of the learned features from two complementary perpectives

- Micro view: individual behavior
  - sparse activations/features
  - convolutional sparse coding layer



- Macro view: collective behavior
  - topology
  - intrinsic dimension
  - neural collapse



- Various low-dimensional structures emerge in both perspectives

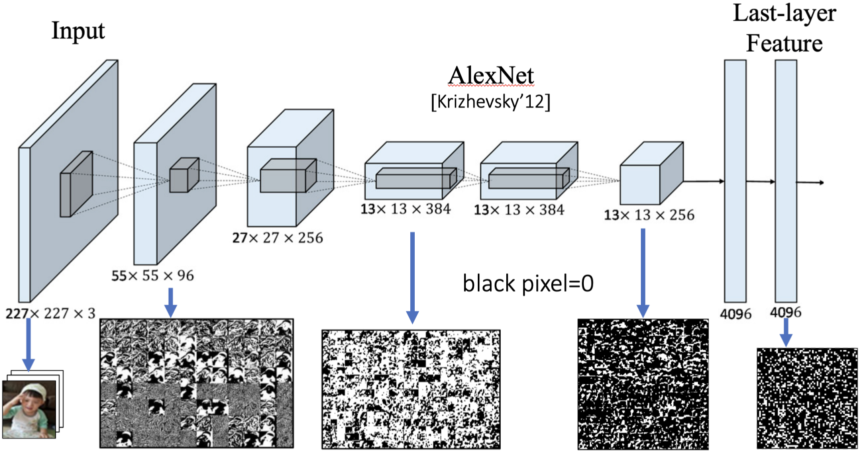
Outline

1. Learned Low-dimensional Features: Micro View
   - Sparse Features are Prevalent
   - Sparse Dictionary Net
   - Transform is sparse

2. Learned Low-dimensional Features: Macro View
   - Topology Change
   - Intrinsic Dimension
   - Neural Collapse (NC)
   - Geometric analysis for understanding NC
   - Exploit NC for improving training efficiency
   - Exploit NC for understanding the effect of loss functions
   - Progressive separation from shallow to deep layers

Sparse Features are Prevalent

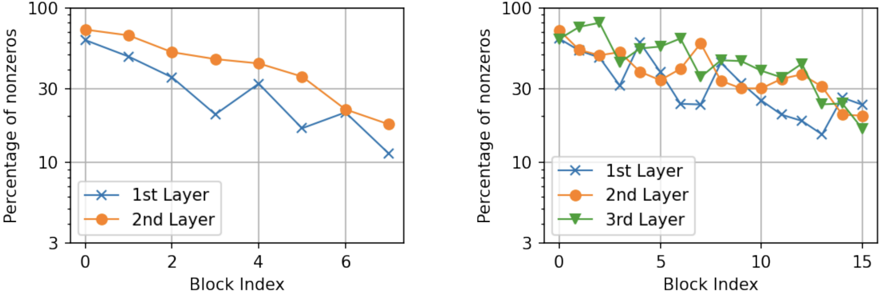
- For each input, the features learned in each layer are sparse[1]



[1]Minsoo Rhu et al, Compressing DMA Engine: Leveraging Activation Sparsity for Training Deep Neural Networks, HPCA, 2018.

Sparse Features are Prevalent

- For each input, the features learned in each layer are sparse[2]



(a) ResNet-18 (b) ResNet-50

- Similar sparse features also appear in other CNNs, e.g., VGG, GoogLeNet, SqueezeNet, etc.
- Could we explicitly control the sparsity?

[2] Zonglin Li , Chong You, et al, Large Models are Parsimonious Learners: Activation Sparsity in Trained Transformers, ICLR 2023.

Convolutional Sparse Coding (CSC) Layer I

- We can replace each layer by a convolutional sparse coding layer[3]
  - Classical convolutional layer

    $$\underbrace{z^\star}_{\text{output}} = \underbrace{W}_{\text{conv matrix}} \underbrace{x}_{\text{input}}$$

  - Analysis model
  - Sparsity is not controllable
  - Easy to implement
  - Convolutional sparse coding layer

    $$\underbrace{z^\star}_{\text{output}} = \arg\min_{z} \| \underbrace{x}_{\text{input}} - Az\|_2^2 + \lambda \|z\|_1$$

  - Synthesis model
  - Sparsity is controllable
  - High computational complexity?
- λ controls the tradeoff between the residual and sparsity
- Use FISTA to compute $z^\star$, producing an unrolled network architecture (see Lecture 2 by Atlas for the details)

Convolutional Sparse Coding (CSC) Layer II

- Use CSC layer to build sparse Sparse Dictionary Net (SDNet)

Dataset	Architecture	Model Size	Top-1 Acc	Memory	Speed
CIFAR-10	ResNet-18 [21]	11.2M	95.54%	1.0 GB	1600 n/s
	ResNet-34 [21]	21.1M	95.57%	2.0 GB	1000 n/s
	MDEQ [27]	11.1M	93.80%	2.0 GB	90 n/s
	SCN [15]	0.7M	94.36%	10.0GB	39 n/s
	SCN-18	11.2M	95.12%	3.5 GB	158 n/s
	SDNet-18 (ours)	11.2M	95.20%	1.2 GB	1500 n/s
	SDNet-34 (ours)	21.1M	95.57%	2.4 GB	900 n/s
ImageNet	ResNet-18 [21]	11.7M	68.98%	24.1 GB	2100 n/s
	ResNet-34 [21]	21.5M	72.83%	32.3 GB	1400 n/s
	SCN [15]	9.8M	70.42%	95.1 GB	51 n/s
	SDNet-18 (ours)	11.7M	69.47%	37.6 GB	1800 n/s
	SDNet-34 (ours)	21.5M	72.67%	46.4 GB	1200 n/s

- SDNet obtains on par performance with similar training time as ResNet, orders of magnitude faster than previous sparse methods

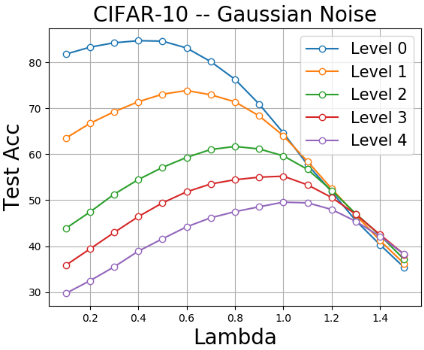
Convolutional Sparse Coding (CSC) Layer III

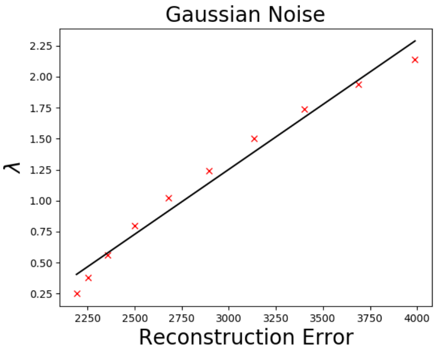
- The convolutional sparse coding model is stable to input corruptions.

Theorem (informal) [Papyan et al.'17] Suppose $\boldsymbol{x}^o = \boldsymbol{A}\boldsymbol{z}^o$ with sparse $\boldsymbol{z}^o$. Given a corrupted input $\boldsymbol{x} = \boldsymbol{x}^o + \boldsymbol{e}$, if we choose $\lambda = O(\|\boldsymbol{e}\|_2)$, then (i) $\boldsymbol{z}^\star$ is sparse with supports contained in $\boldsymbol{z}^o$, and (ii) $\|\boldsymbol{z}^\star - \boldsymbol{z}^o\| = O(\|\boldsymbol{e}\|_2)$.

- If data is corrupted, we can adjust λ to produce robust prediction.
- No need to modify the training procedure, unlike existing ones that require heavy data augmentation or additional training techniques.

Convolutional Sparse Coding (CSC) Layer IV





- The optimal λ increases with the corruption/noise level.
- As the reconstruction error correlates with the corruption level, we can estimate the optimal λ by linearly fitting the reconstruction error.

Convolutional Sparse Coding (CSC) Layer V

- SDnet is more robust compared to classical DNNs [Dai et al.'22]

Severity Level	Level-0	Level-1	Level-2	Level-3	Level-4
ResNet-18 [21]	79.43%	56.17%	34.86%	28.23%	23.45%
SCN [15]	80.89%	60.21%	44.97%	37.79%	30.11%
SDNet-18 w/ $\lambda = 0.1$	81.78%	63.50%	43.86%	35.84%	27.92%
SDNet-18 w/ adaptive λ	84.76%	74.87%	61.38%	54.77%	48.84%
λ from linear fitting	0.49	0.60	0.75	0.84	0.94

- SDnet is also robust to adversarial perturbation using PGD attack

Model	Robust Accuracy ($L_\infty = 8/255$)	Robust Accuracy ($L_2 = 0.5$)
ResNet-18 [21]	0.01%	29.47%
SDNet-18 w/ $\lambda = 0.1$	0.11%	29.95%
SDNet-18 (After tuning λ)	35.18%	62.80%

[3] Papyan et al., Working locally thinking globally: Theoretical guarantees for convolutional sparse coding, TSP 2017.
Sun et al, Supervised deep sparse coding networks for image classification, TIP 2019.
Dai et al, Revisiting Sparse Convolutional Model for Visual Recognition, NeurIPS 2022.

Larger Models, the Sparser I

- Transformers are Sparse[4]

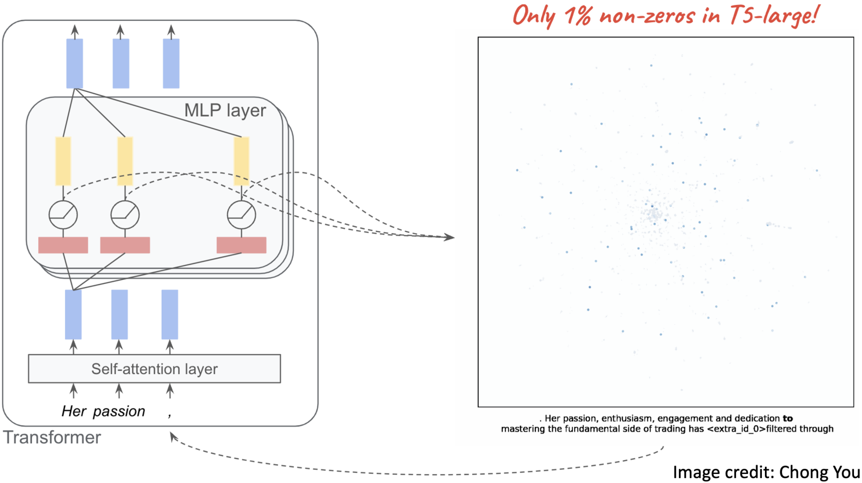
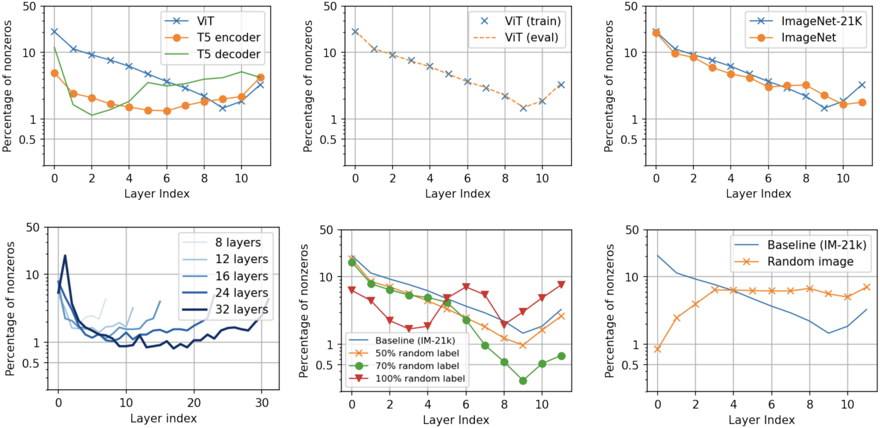


Image credit: Chong You

Larger Models, the Sparser II

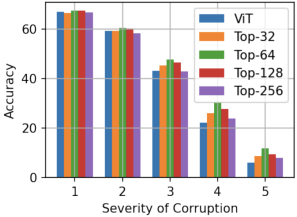
- Sparse activations (features) emerge in different transformers and datasets for both natural language processing and vision tasks[4]



Larger Models, the Sparser III

- Sparsity can be exploited to improve efficiency, robustness, and calibration: a top-k transformer that only keeps the top-k largest values of the activation maps[4]

Methods	Natural Accuracy	Accuracy w/ Train Label Noise 40%	80%	Accuracy under Input Perturbation Gaussian	Impulse	Shot
ViT	74.85%	59.44%	25.35%	39.54%	37.37%	38.56%
Top-128 ViT	74.83%	62.13%	30.80%	42.29%	40.07%	40.68%

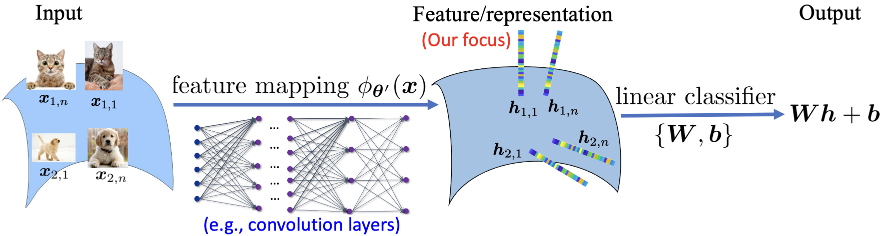


- See tomorrow Sam's lecture on **White-Box Transformers**

[4] Zonglin Li , Chong You, et al, Large Models are Parsimonious Learners: Activation Sparsity in Trained Transformers, ICLR 2023.

From individual to collective behaviors

- Characterize how the features facilitate our decision tasks
  - For classification: how the features are **separated/discriminative** across different classes.
- We will study the **collective** behaviors of the features $\{\boldsymbol{h}_{k,i}\}$ of the entire classes of objects



Outline

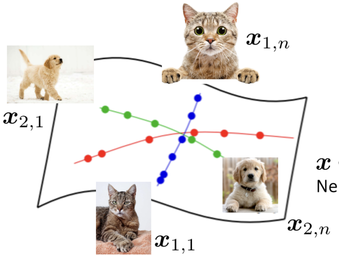
1. Learned Low-dimensional Features: Micro View
   - Sparse Features are Prevalent
   - Sparse Dictionary Net
   - Transform is sparse

2. Learned Low-dimensional Features: Macro View
   - Topology Change
   - Intrinsic Dimension
   - Neural Collapse (NC)
   - Geometric analysis for understanding NC
   - Exploit NC for improving training efficiency
   - Exploit NC for understanding the effect of loss functions
   - Progressive separation from shallow to deep layers

Setup: Image Classification Problem

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

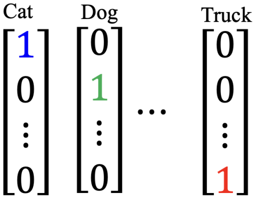
- $K = 10$ classes (MNIST, CIFAR10, etc)
- $K = 1000$ classes (ImageNet)



Data in the input space

$\boldsymbol{x} \longrightarrow \boldsymbol{y}$

Neural network



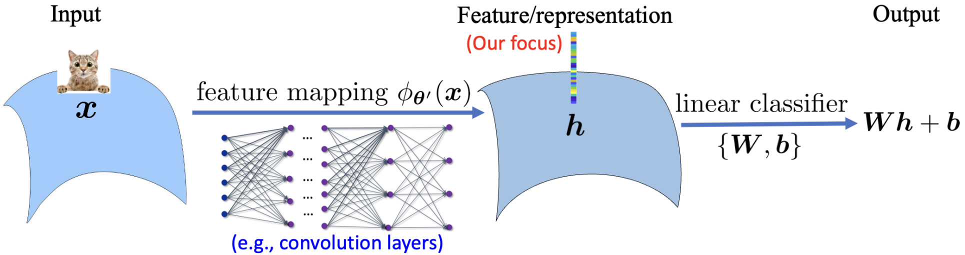
One-hot labeling vectors in $\mathbb{R}^K$

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

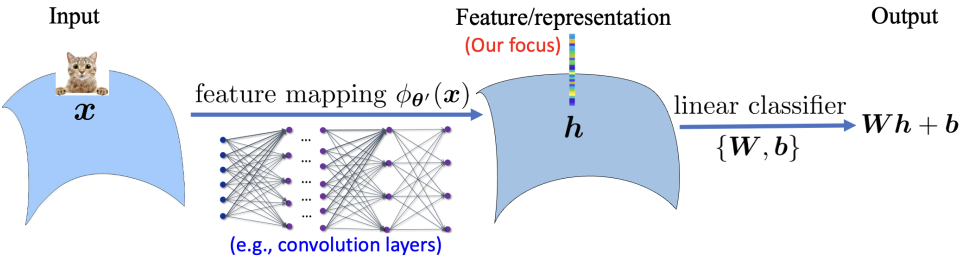
A deep neural network classifier often contains two parts: a feature mapping and a linear classifier



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

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

Deep Neural Network Classifiers II



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

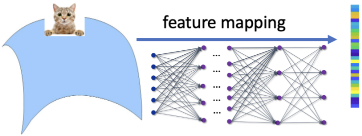
Prediction (probability) Target

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

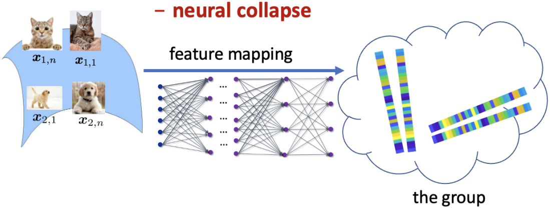
Focus: Geometrization of Learned Representations

- We will characterize different properties of the learned features from two complementary perpectives

- Micro view: individual behavior
  - sparse activations/features
  - convolutional sparse coding layer



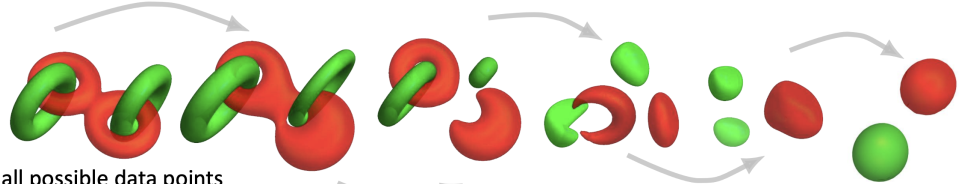
- Macro view: collective behavior
  - **topology**
  - **intrinsic dimension**
  - **neural collapse**



- Various low-dimensional structures emerge in both perspectives

Representations: Topology Change I

- The topology of two classes $\mathcal{M}_1 \cup \mathcal{M}_2$ changes through the layer-wise transformation[5]

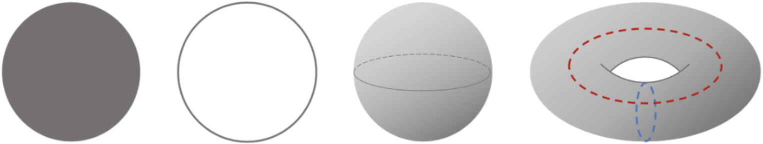


all possible data points from two classes; not a single input!

- Progressively separate the two classes from shallow to deep layers

Representations: Topology Change II

- Study of topology of shapes dates back to Leonhard Euler in 18th century
- Algebraic topology offers a mature set of tools for counting and collating holes[6]
- The number of holes of an entire class $\mathcal{M}$ is called the Betti number
  - **zeroth** Betti number $\beta_0(\mathcal{M})$: number of **connected** components
  - k-th Betti number $\beta_k(\mathcal{M})$: the number of k-dimensional holes

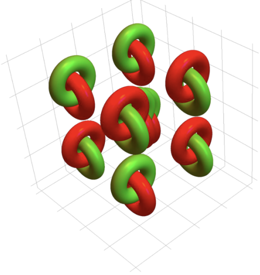


$\beta_0 = 1$	$\beta_0 = 1$	$\beta_0 = 1$	$\beta_0 = 1$
$\beta_1 = 0$	$\beta_1 = 1$	$\beta_1 = 0$	$\beta_1 = 2$
$\beta_2 = 0$	$\beta_2 = 0$	$\beta_2 = 1$	$\beta_2 = 1$

Representations: Topology Change III

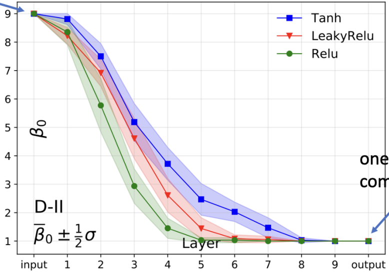
- The beti number progressively decreases from shallow to deep layers

two classes



9 connected components

change of zeroth Betti number of one class

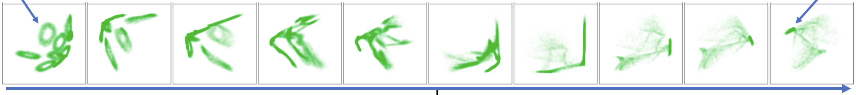


one connected component

9 holes

projection onto the first two principal components

concentration



input

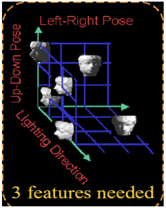
deeper

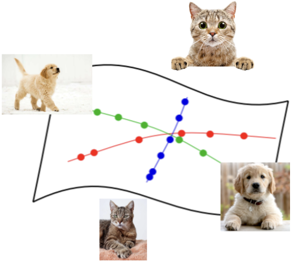
[5] Naitzat et al., Topology of deep neural networks, JMLR 2020.

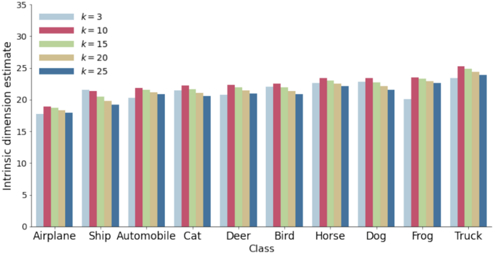
[6] Ghrist, Robert, Barcodes: the persistent topology of data, Bulletin of the American Mathematical, 2008.

Representations: Intrinsic Dimension

- Intrinsic dimension for a data set $\mathcal{M}$: viewed as the minimal number of variables to describe the data





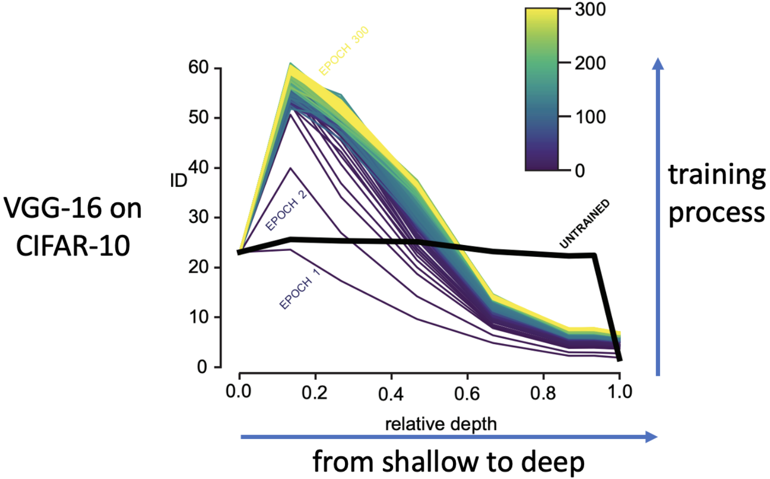


- Natural images lies on a manifold of low intrinsic dimension[7]

[7] Brown et al, Verifying the Union of Manifolds Hypothesis for Image Data, ICLR 2023.

Representations: Intrinsic Dimension

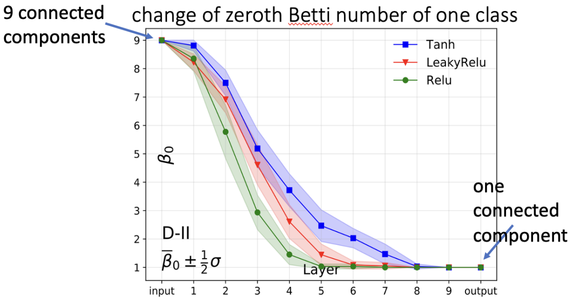
- Intrinsic dimension first increases, then progressively decreases across layers[8]

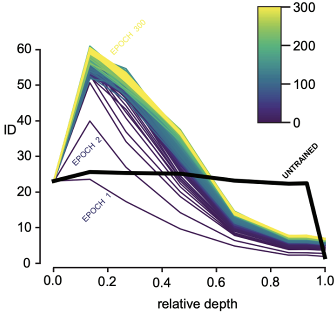


8 Ansuini et al., Intrinsic dimension of data representations in deep neural networks, NuerIPS 2019.

Representations: Topology and Intrinsic Dimension

- Both topology and intrinsic dimension perspectives capture certain low-dimensional structures in the learned representations
- But neither captures the geometry that distinguishes different classes of objects





Neural Collapse in Classification I

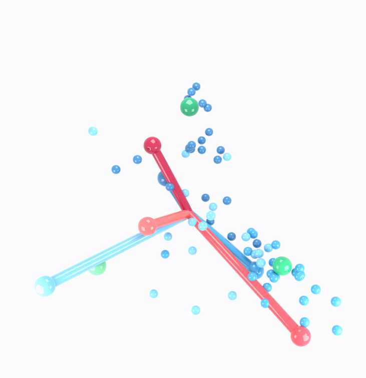
Prevalence of neural collapse during the terminal phase of deep learning training

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)



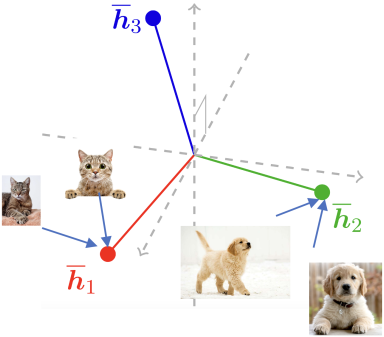
- Reveals common outcome of learned features and classifiers across a variety of architectures and dataset
- Precise mathematical structure within the features and classifier

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-dim features: they live on a K-dim subspace*):

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

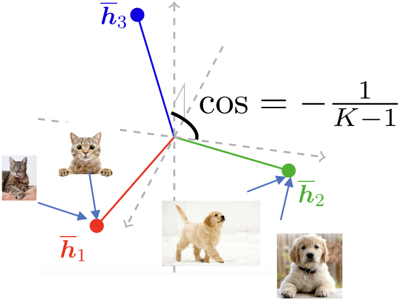


Neural Collapse in Classification III

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

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



Neural Collapse in Classification IV

- For any K unit-length vectors $\boldsymbol{u}_1, \ldots, \boldsymbol{u}_K$ in $\mathbb{R}^d$ (with $d \geq K-1$), then $\max_{k \neq k'} \langle \boldsymbol{u}_k, \boldsymbol{u}_{k'} \rangle \geq -\frac{1}{K-1}$ and the minimum is achieved when they form a simplex ETF [Rankin'55].
- The simplest case of the Optimal Packings on Spheres, or the Tammes problem.
- Proof:

$$0 \leq \Big\| \sum_{k=1}^{K} \boldsymbol{u}_k \Big\|_2^2 \leq K + K(K-1) \max_{k \neq k'} \langle \boldsymbol{u}_k, \boldsymbol{u}_{k'} \rangle$$

$$\Longrightarrow \max_{k \neq k'} \langle \boldsymbol{u}_k, \boldsymbol{u}_{k'} \rangle \geq -\frac{1}{K-1}$$

achieves equality when $\sum_{k=1}^{K} \boldsymbol{u}_k = 0$ and $\langle \boldsymbol{u}_k, \boldsymbol{u}_{k'} \rangle = -\frac{1}{K-1}, \forall k \neq k'$

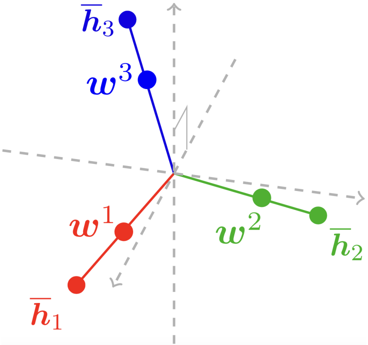
Neural Collapse in Classification V

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 VI

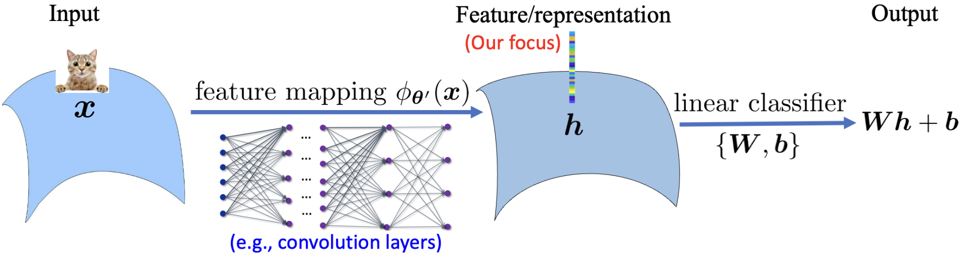
NC is preferred among every successful exercise in feature engineering [Papyan et al.'20]

- 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 4 to understand NC phenomenon

Simplification: Unconstrained Features I

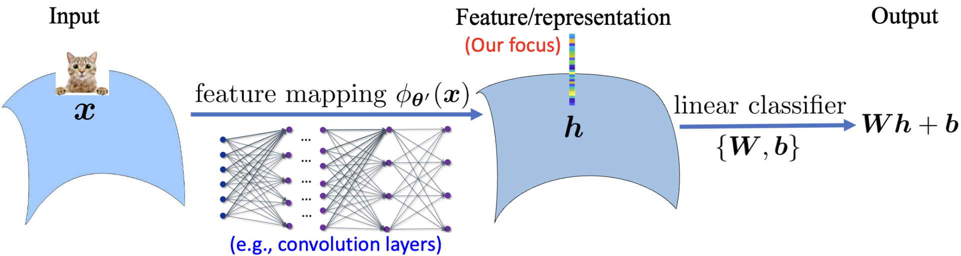


Training problem is highly nonconvex [Li et al.'18]:

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

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

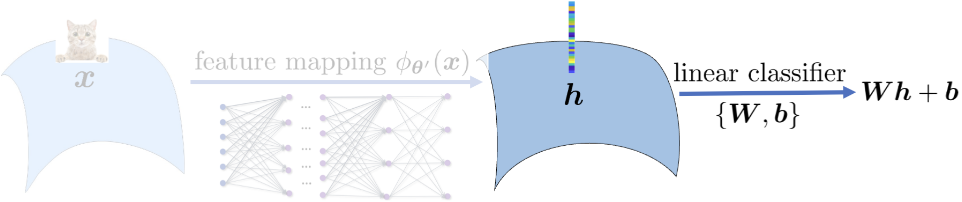
Simplification: Unconstrained Features II



- 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

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

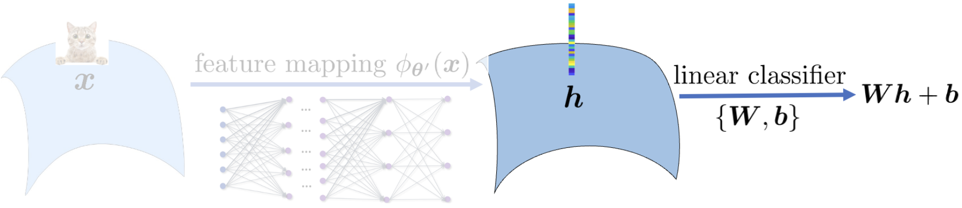
[E et al.'20, Fang et al.'21, Gral et al.'21, etc.] study constrained formulation

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

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 4: 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

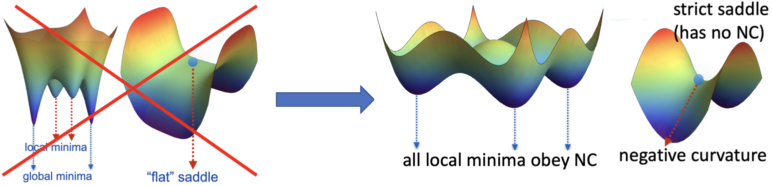
$$\underbrace{\boldsymbol{h}_{k,i}^\star = \overline{\boldsymbol{h}}_k^\star}_{\text{NC1}}, \quad \underbrace{\frac{\langle \overline{\boldsymbol{h}}_k^\star, \overline{\boldsymbol{h}}_{k'}^\star\rangle}{\|\overline{\boldsymbol{h}}_k^\star\|\|\overline{\boldsymbol{h}}_{k'}^\star\|} = \begin{cases} 1,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

- $d \geq K-1$ 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

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

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]

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

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.

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

Geometric Analysis for Unconstrained Features Model VI

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

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

- Step 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

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

Theorem (global optimality & 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.

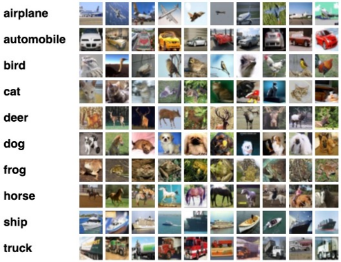
Message. Iterative algorithms such as (stochastic) gradient descent will always learn Neural Collapse features and classifiers.

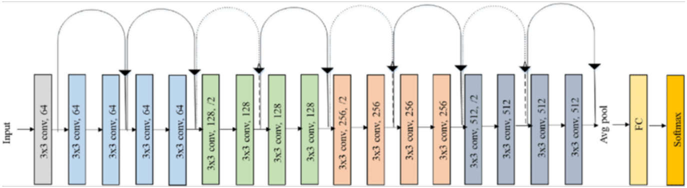
Experiments on Practical Neural Networks

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

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

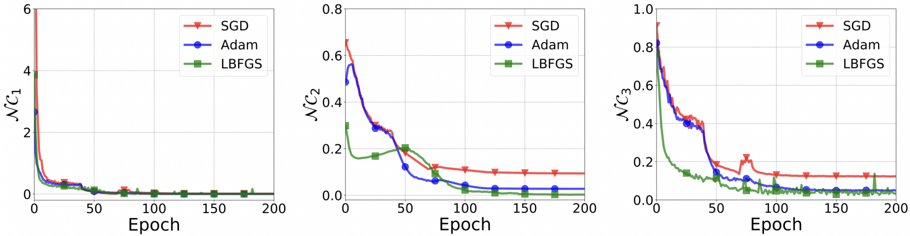
- $K = 10$ classes
- 50K training images
- 10K testing images





Experiments: NC is algorithm independent I

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



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

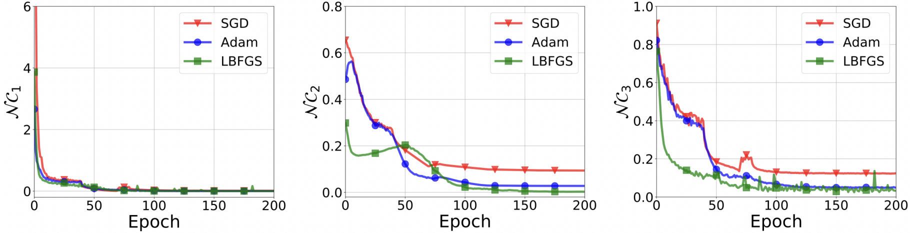
$\mathcal{NC}_1 = \mathrm{trace}(\boldsymbol{\Sigma}_W \boldsymbol{\Sigma}_B^\dagger)$ small when features are collapsed and separated

within-class covariance (noise term) $\boldsymbol{\Sigma}_W = \dfrac{1}{nK}\displaystyle\sum_{k=1}^{K}\sum_{i=1}^{n}(\boldsymbol{h}_{k,i} - \overline{\boldsymbol{h}}_k)(\boldsymbol{h}_{k,i} - \overline{\boldsymbol{h}}_k)^\top$

between-class covariance (signal term) $\boldsymbol{\Sigma}_B = \dfrac{1}{K}\displaystyle\sum_{k=1}^{K}(\overline{\boldsymbol{h}}_k - \boldsymbol{h}_G)(\overline{\boldsymbol{h}}_k - \boldsymbol{h}_G)^\top$

Experiments: NC is algorithm independent II

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

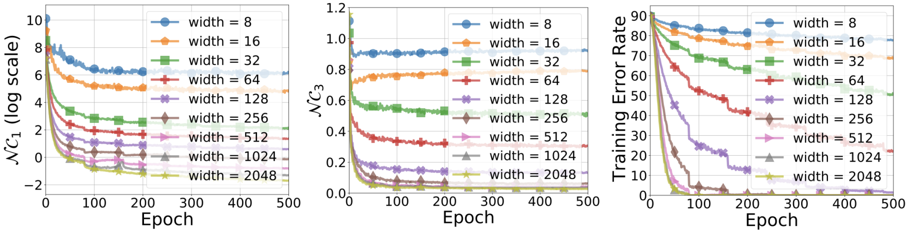


Within-Class Variability (NC1) **Between-Class Separation (NC2)** **Self-Duality Collapse (NC3)**

- 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] show NC leads to better

- Generalization performance
- Robustness

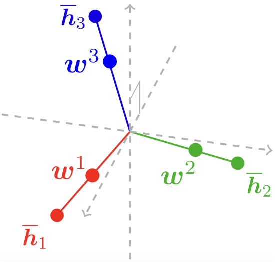
We can also exploit NC for

- Improving training efficiency (covered later)
- Understanding the effect of loss functions (covered later)
- Understanding transferability (covered in Qing's lecture)
- Imbalanced learning
- Incremental learning
- etc.

Exploit NC for Improving Training & Memory I

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

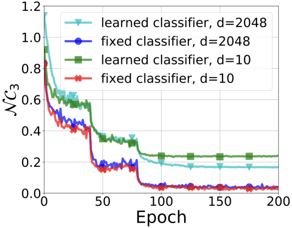
- **Implication 1: No need to learn the classifier** [Hoffer et al. 2018]
  - Just fix it as a Simplex ETF
  - Save **8%, 12%, and 53%** parameters for ResNet50, DenseNet169, and ShuffleNet!
- **Implication 2: No need of large feature dimension** d
  - Just use feature dim. $d = \#$class K (e.g., $d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



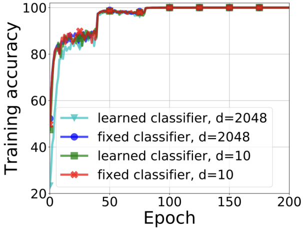
Exploit NC for Improving Training & Memory II

ResNet50 on CIFAR-10 with different settings

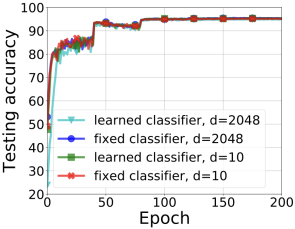
- Learned classifier (default) VS fixed classifier as a simplex ETF
- Feature dim $d = 2048$ (default) VS $d = 10$



Self-Duality Collapse (NC3)



Training Accuracy

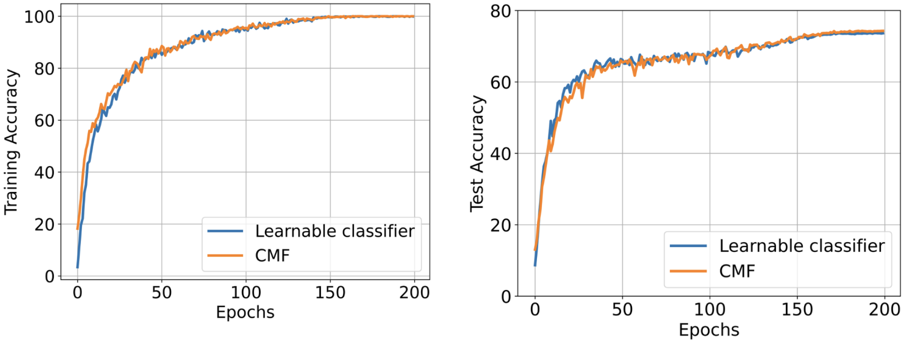


Testing Accuracy

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

Exploit NC for Improving Training & Memory III

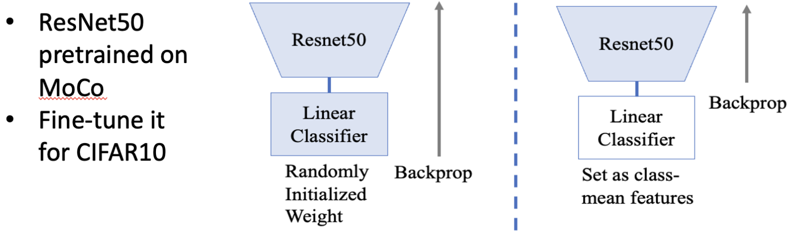
- Class-mean features (CMF) classifier: by NC3 (self-duality), we can also fix the classifier as the class-mean features during training[9]



- Achieves on-par performance with learned classifiers (ResNet18 on CIFAR100)

Exploit NC for Improving Training & Memory IV

- CMF classifier improves Out-of-distribution (OOD) performance for fine-tuning[9]



Test on CIFAR10 (ID)	97.00%	98.00%
Test on STL10 (OOD)	87.42%	90.67%

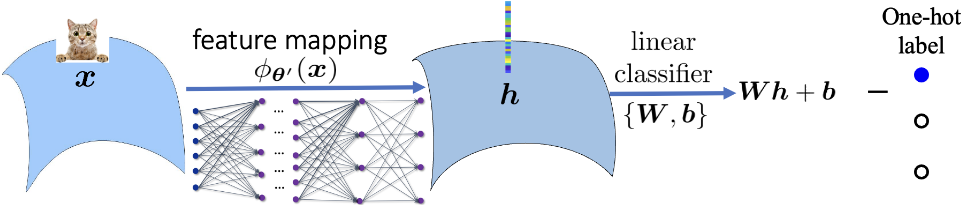
- CMF is simpler to the two-stage approach[10]

[9] Jiang, et al., Zhu, Generalized Neural Collapse for a Large Number of Classes, 2023

[10] Kumar, Ananya, et al., Fine-Tuning can Distort Pretrained Features and Underperform Out-of-Distribution, ICLR 2022.

Is Cross-entropy Loss Essential?

Is cross-entropy loss essential to neural collapse?



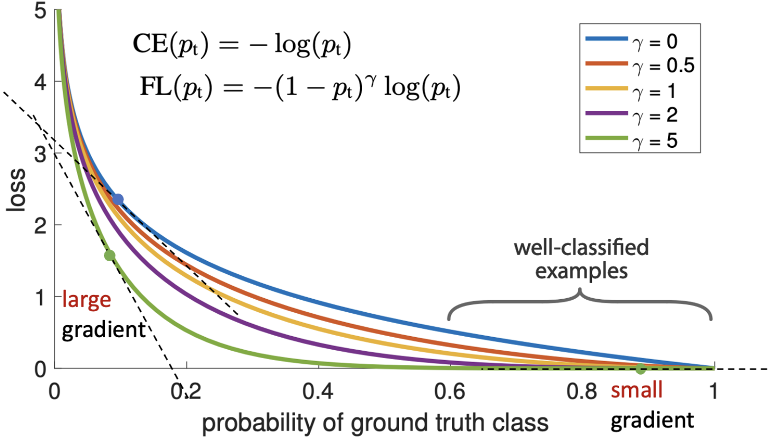
We can measure the mismatch between the network output and the one-hot label in many ways.

Various losses and tricks (e.g., label smoothing, focal loss) have been proposed to improve network training and performance[11]

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

Focal Loss (FL)

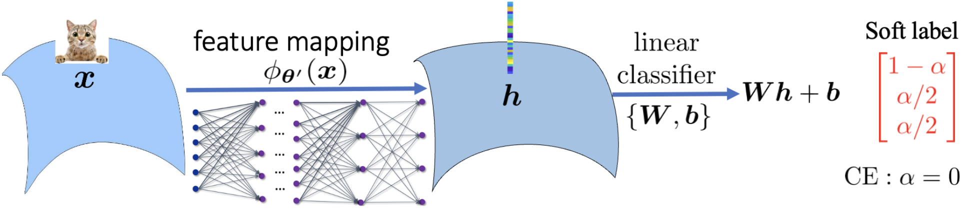
Focal loss puts more focus on hard, misclassified examples[12]



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

Label Smoothing (LS)

Label smoothing replaces the hard label by a soft label[13]



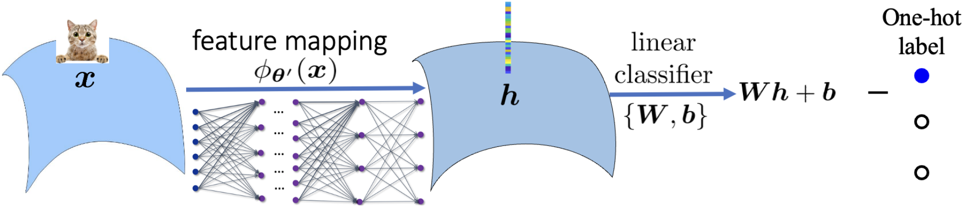
$$\text{Output: } \boldsymbol{W}\boldsymbol{h} + \boldsymbol{b} = \begin{bmatrix} 1 \\ 0 \\ -1 \end{bmatrix} \xrightarrow{\text{Softmax function}} \begin{bmatrix} 0.6 \\ 0.3 \\ 0.1 \end{bmatrix} \begin{matrix} \text{Cat} \\ \text{Dog} \\ \text{Panda} \end{matrix} \begin{bmatrix} 1-\alpha \\ \alpha/2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

Prediction **Target**

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

[13] 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?



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

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.[14]

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

Which Loss is the Best to Use?

- Which loss is the best to use is still a mystery

Testing accuracy (%) for WideResNet18 on mini-ImageNet with different widths and training iterations

Loss	CE	FL	LS	MSE
Width = × 0.25 Epoches = 200	71.95	70.20	70.40	69.15

Which Loss is the Best to Use?

- Which loss is the best to use is still a mystery

Testing accuracy (%) for WideResNet18 on mini-ImageNet with different widths and training iterations

Loss	CE	FL	LS	MSE
Width = × 0.25 Epoches = 200	71.95	70.20	70.40	69.15
Width = × 2 Epoches = 800	79.30	79.32	80.20	79.62

- The performance is also affected by the choice of network architecture, training iterations, dataset, etc.
- All the losses lead to largely identical performance when the network is sufficiently large and trained longer enough

Are All Loses Created Equal?—A NC Perspective I

We study them under the unconstrained feature model:

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

Contrastive property [Zhou et al.'22] We say a loss function $\mathcal{L}$ satisfies the contrastive property if there exists a scalar function ψ s.t.

1. $\mathcal{L}(\boldsymbol{z},\boldsymbol{y}_k)\geq\psi\big(\sum_{j\neq k}(z_j-z_k)\big)$, where the equality holds only when $z_j=z_{j'}$ for all $j,j'\neq k$;
2. $t^\star=\arg\min_t\psi(t)+c|t|$ is unique for any $c>0$ and $t^\star\leq 0$.

Intuition: (1) min $\psi\big(\sum_{j\neq k}(z_j-z_k)\big)$ contrasts the k-th output z_k simultaneously to all the other outputs, (2) $t^\star\leq 0$ ensures minimizer has the k-th output z_k being its largest entry and hence correct prediction.

Are All Loses Created Equal?—A NC Perspective II

Contrastive property [Zhou et al.'22] We say a loss function satisfies the contrastive property if there exists a scalar function ψ such that

1. $\mathcal{L}(\boldsymbol{z}, \boldsymbol{y}_k) \geq \psi\big(\sum_{j \neq k}(z_j - z_k)\big)$, where the equality holds only when $z_j = z_{j'}$ for all $j, j' \neq k$;
2. $t^\star = \arg\min_t \psi(t) + c|t|$ is unique for any $c > 0$ and $t^\star \leq 0$.

CE, FL and LS all satisfy the contrastive property.

Theorem (informal) [Zhou et al.'22] With feature dim. $d \geq$ #class $K - 1$, all the losses with contrastive property lead to the same global solutions: NC features and classifiers.

Are All Loses Created Equal?—A NC Perspective III

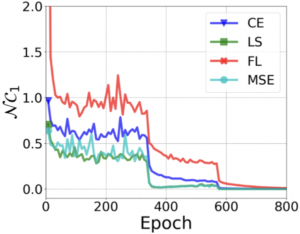
Theorem (informal) With feature dim. $d \geq \#\text{class } K - 1$, all the one-hot labeling based losses (e.g., CE, FL, LS, MSE) lead to (almost) the same NC features and classifiers [Han et al'21, Tirer & Bruner'22, Zhou'22].

Implication for practical networks If network is *large enough and trained longer enough*

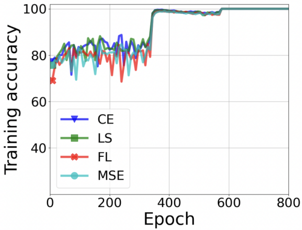
- All losses lead to largely identical features on training data—NC phenomena
- All losses lead to largely identical performance on test data (experiments in the following slides)

Are All Loses Created Equal?—A NC Perspective IV

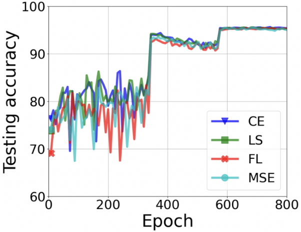
ResNet50 on CIFAR-10 with **different training losses**



Within-Class Variability (NC1)



Train accuracy

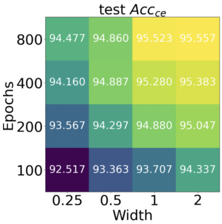


Testing accuracy

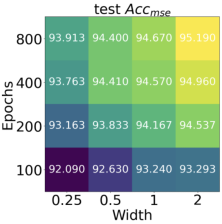
- NC across **different training losses**
- If network is *large enough and trained longer enough*
  - All losses lead to largely identical features on training data—NC phenomena
  - All losses lead to largely identical performance on test data

Are All Loses Created Equal?—A NC Perspective V

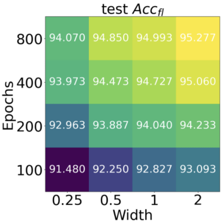
ResNet50 (with different network widths and training epoches) on CIFAR-10 with **different training losses**



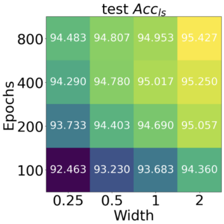
Cross-entropy



Mean-squared Error



Focal loss



Label smoothing

- Right top corners not only have better performance, but also have smaller variance than left bottom corners
- 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

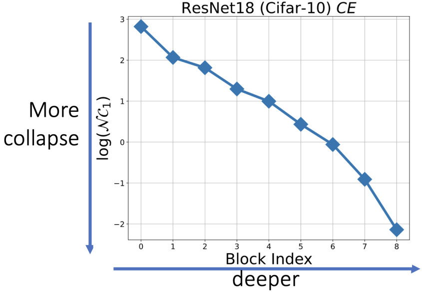
Progressive separation from shallow to deep layers

- How the data are progressively separated across the layers?[15]

$$\mathcal{NC}_1 = \text{trace}(\mathbf{\Sigma}_W \mathbf{\Sigma}_B^\dagger)$$

within-class covariance $\mathbf{\Sigma}_W$

between-class covariance $\mathbf{\Sigma}_B$

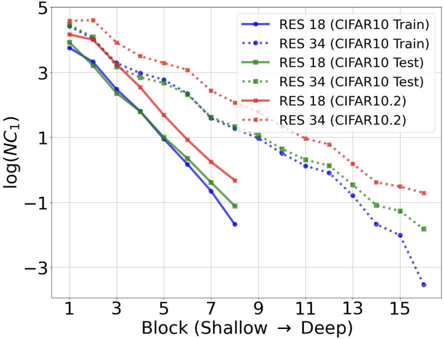


- Effect of depths: create progressive separation and concentration (geometric decay of $\mathcal{NC}_1$)

[15] V. Papyan, Traces of class/cross-class structure pervade deep learning spectra, JMLR, 2021. He & Su, A Law of Progressive Separation for Deep Learning, 2022.

Progressive separation from shallow to deep layers

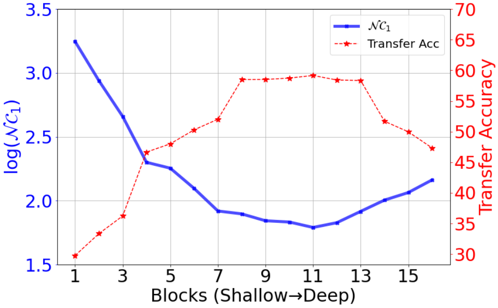
- Progressive separation is robust to distribution shift.
  - Pretrained on CIFAR10
  - Evaluate layer-wise NC on CIFAR10 training (blue), CIFAR10 testing (green), & CIFAR10.2 testing (red) [Lu'20]
  - Model is fixed without fine-tuning



- Observe similar trend of progressive separation and collapse
- Distribution shift causes slightly less collapse (worse performance)

Progressive separation from shallow to deep layers

- Progressive separation is transferable among different tasks
  - ResNet-34 pre-trained on ImageNet
  - Evaluate on CIFAR10
  - Model is fixed without fine-tuning
  - Train a linear classifier on top of the features



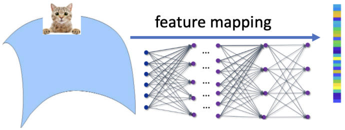
- Layer-wise NC exhibits two phases on downstream tasks:
  - Phase 1: progressively decreasing (universal feature mapping)
  - Phase 2: progressively increasing (specific feature mapping)
- Projection heads and fine-tuning help transferability [Qing's talk]

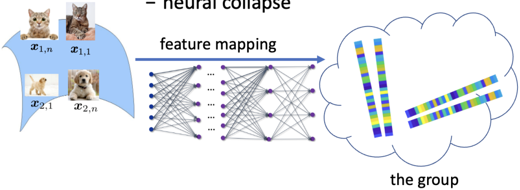
Take-home Message

- Learned features exhibit low-dimensional structures in different aspects (sparse activations and neural collapse properties)

- Micro view: individual behavior
  - sparse activations/features
  - convolutional sparse coding layer

- Macro view: collective behavior
  - topology
  - intrinsic dimension
  - neural collapse





- These structures can be exploited to understand and improve network performance

Call for Papers

IEEE JSTSP Special Issue on Seeking Low-dimensionality in Deep Neural Networks (SLowDNN)
Manuscript Due: November 30, 2023 https://signalprocessingsociety.org/sites/default/files/uploads/special_issues_deadlines/JSTSP_SI_seeking_low.pdf

Conference on Parsimony and Learning (CPAL) https://cpal.cc/
January 2024, Hongkong
Manuscript Due: August 28, 2023

Thank You! Questions?