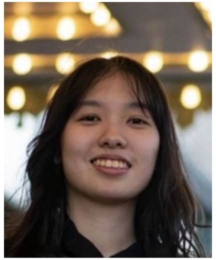
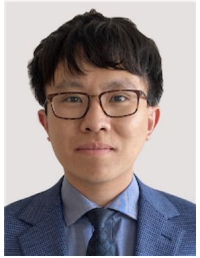
NEURAL INFORMATION PROCESSING SYSTEMS

Learning Structured Representations with Hyperbolic Embeddings

Aditya Sinha*, Siqi Zeng*, Makoto Yamada, Han Zhao



github.com/UIUCTML/HypStructure

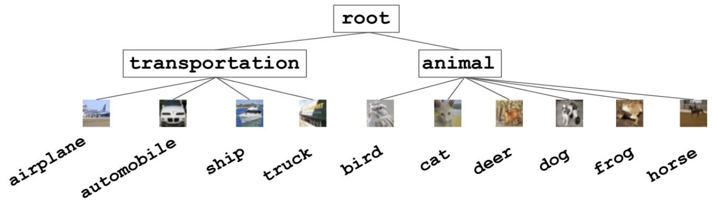
UNIVERSITY OF ILLINOIS URBANA-CHAMPAIGN

OIST

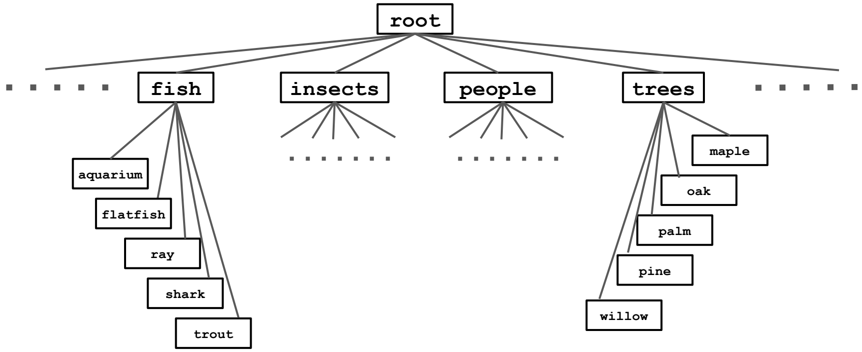
Motivation

NEURAL INFORMATION
PROCESSING SYSTEMS

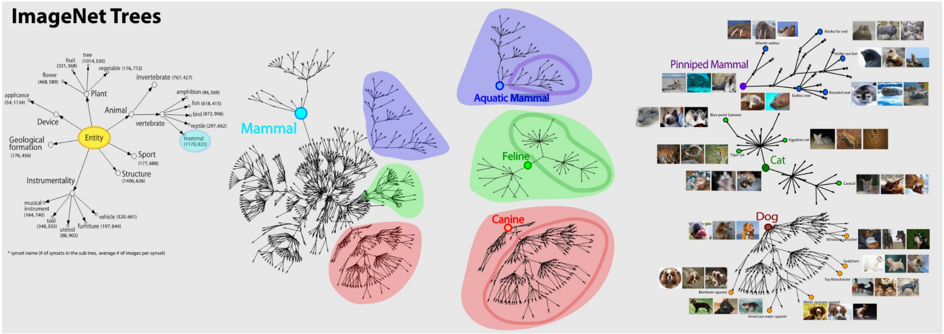
- **Hierarchical** Label Structures widely exist in many real-world datasets



CIFAR10 label hierarchy [Krizhevsky et. al, 2009]



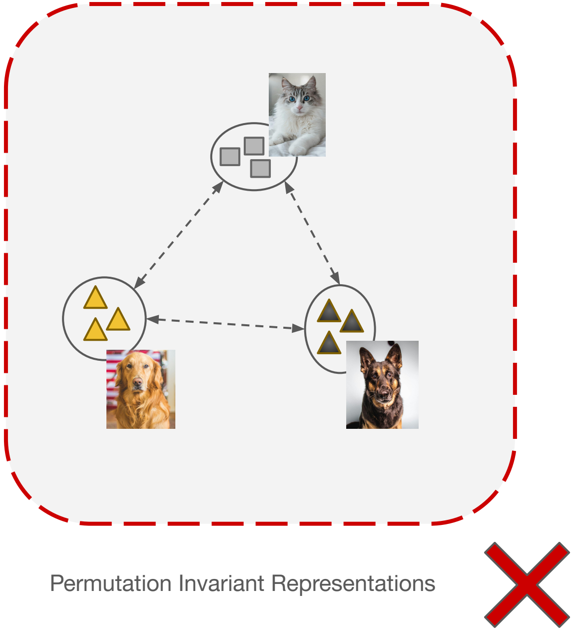
CIFAR100 label hierarchy [Krizhevsky et. al, 2009]



ImageNet-1k label hierarchy [Li et. al, 2009]

Motivation

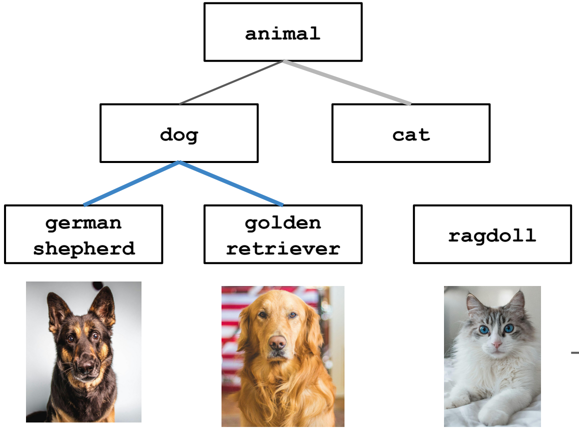
- Most representation learning methods → permutation invariant

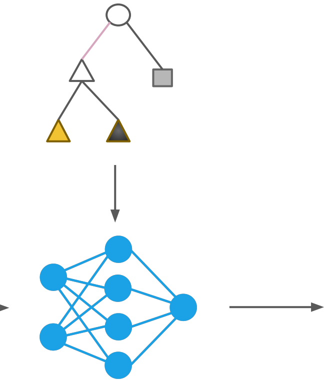


- Ignores the hierarchical semantic relationships between classes in the feature space

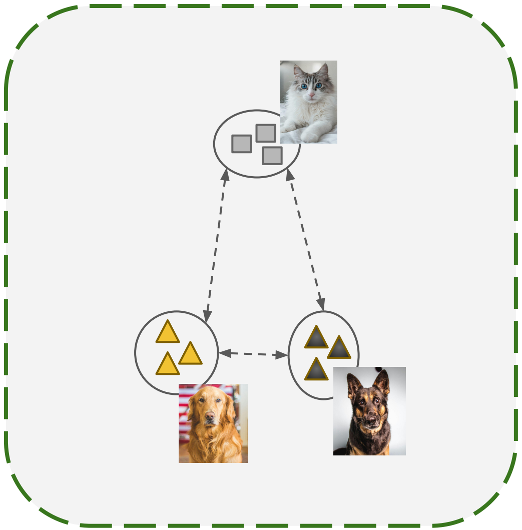
Motivation

- Structured Representation Learning → hierarchy informed representations [Zeng et. al, 2022]





Structured Representation Learning [Zeng et. al, 2022]



Structure Informed Representations

- Distances between the representations in the feature space are consistent with the semantic context.

ℓ_2-Cophenetic Correlation Coefficient (**CPCC**)

NEURAL INFORMATION PROCESSING SYSTEMS

- [Zeng et. al, 2022] → Use Cophenetic Correlation Coefficient (**CPCC**) [Sokal and Rohlf, 1962] for structural regularization
- Definition → $\mathrm{CPCC}(d_{\mathcal{T}}, \rho) := \dfrac{\sum_{i<j}(d_{\mathcal{T}}(v_i, v_j) - \overline{d_{\mathcal{T}}})(\rho(v_i, v_j) - \overline{\rho})}{\sqrt{\sum_{i<j}(d_{\mathcal{T}}(v_i, v_j) - \overline{d_{\mathcal{T}}})^2}\sqrt{\sum_{i<j}(\rho(v_i, v_j) - \overline{\rho})^2}}$

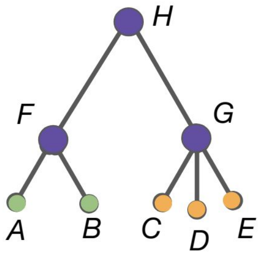
- $\rho(v_i, v_j) :=$ Euclidean (ℓ_2) distance between two **class centroids** of the fine class representations
- $d_{\mathcal{T}}(v_i, v_j) :=$ The **shortest tree distance** between the two classes in the hierarchy

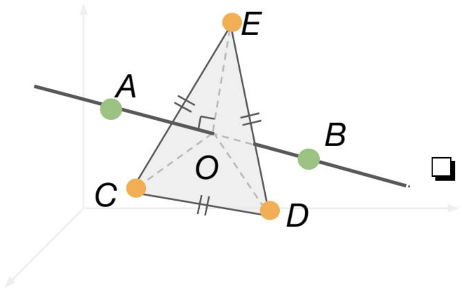
- Composite optimization objective with structured regularization on the hierarchy:

$$\mathcal{L}(\mathcal{D}) = \sum_{(\boldsymbol{x}, y) \in \mathcal{D}} \ell_{\text{Flat}}(\boldsymbol{x}, y, \theta, w) - \alpha \cdot \mathrm{CPCC}(d_{\mathcal{T}}, \rho)$$

Challenges with ℓ_2-CPCC

- Cannot embed some trees in the Euclidean space (ℓ_2) exactly → Distort the underlying semantic context in the hierarchy

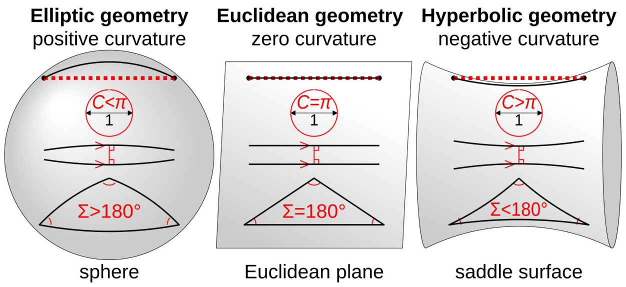




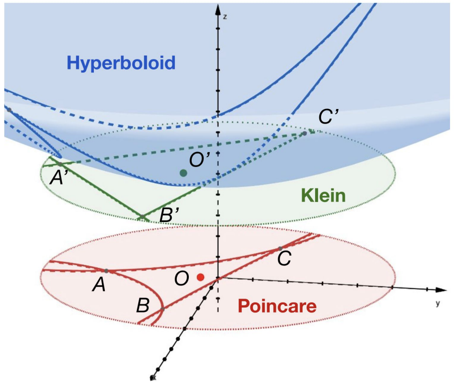
- Let us attempt to embed leaf nodes A, B, C, D, E into the Euclidean space.
- CG = DG = EG = 1, CD = DE = CE = 2
  - ⇒ CD, DE, CE must be on a plane with equilateral $\triangle_{CDE}$
  - ⇒ Green classes (A, B) have same distance 4 to Yellow classes (C,D,E)
  - ⇒ A, B must be on the line perpendicular to $\triangle_{CDE}$ and intersecting the plane at O (the barycenter of $\triangle_{CDE}$)
  - ⇒ Due to uniqueness and symmetry of A,B, we must have AO = BO = 1
  - ⇒ We must have AB = 2
- AO = 1, OE = $(2\sqrt{3})/3$, AE = 4, which **contradicts** the Pythagorean Theorem

Solution: Hyperbolic Geometry

- **Hyperbolic Geometry** → more suitable alternative:
  - Non euclidean spaces with negative curvature unlike ℓ_2
  - Hyperbolic spaces are continuous analogues of trees
  - Allow embedding tree-like data in finite dimensions and low distortion [Sarkar, 2012]
  - Used in NLP, Image Classification, Object Detection, action retrieval …
- Several **isometric** models → easy transformations between geometries
  - (right) relationship between the commonly used Poincare, Klein and Hyperboloid models



[Non-euclidean geometry, Wikipedia 2020]



`HypStructure`: `Hyp`erbolic `Structure`d regularizer

- **Goal**: *accurately* and *explicitly* embed the label hierarchy → representation space

- **`HypStructure`**: label-hierarchy based regularization approach for structured learning in hyperbolic space

- **Advantages**:
  - Can be easily combined with any standard task losses for optimization
  - Enables learning of discriminative and *hierarchy-informed* features
  - More interpretable and tree-like representations
  - Beneficial across tasks and datasets → representation learning, ID classification, OOD detection
  - Formal analysis of the *hierarchy-informed* features → better understanding of structured representation learning

HypStructure: **HypCPCC** and **HypCenter**

NEURAL INFORMATION PROCESSING SYSTEMS

- **HypStructure**: Combination of two losses (1) <u>Hyp</u>erbolic <u>Cop</u>henetic <u>C</u>orrelation <u>C</u>oefficient Loss (**HypCPCC**) and (2) <u>Hyp</u>erbolic <u>Cen</u>tering Loss (**HypCenter**)

- **HypCPCC**: extend ℓ_2-CPCC [Zeng et. al, 2022] to the hyperbolic space
  I. map Euclidean vectors to Poincare space
  II. compute class prototypes
  III. use Poincare distance for CPCC computation

- **HypCenter**: Inspired from Sarkar's construction [2012]
  - place root node at the origin
  - ℓ_{center} loss → minimize the norm of the hyperbolic representations of the root

- Learn hierarchy-informed representations by minimizing:

$$\mathcal{L}(\mathcal{D}) = \sum_{(\boldsymbol{x},y)\in\mathcal{D}} \ell_{\text{Flat}}(\boldsymbol{x}, y, \theta) - \alpha \cdot \text{HypCPCC}(d_{\mathcal{T}}, d_{\mathbb{B}_c}) + \beta \cdot \ell_{\text{center}}(\boldsymbol{x}, \theta)$$

`HypStructure`: Algorithm

Algorithm 1 `HypStructure`: Hyperbolic Structured Representation Learning

Input: Batch size B, Label tree $\mathcal{T} = (V, E, e)$, Number of epochs K, Task Loss formulation ℓ_{Flat}, Encoder f_θ, Classifier Head g_w, Learning Rate η, Hyperparameters α, β

Initialize model parameters: θ, w
for epoch = $1, 2, \ldots K$ **do**
  for batch = $1, 2, \ldots, B$ **do**
    Get image-label pairs: $\{(\boldsymbol{x}_i, y_i)\}_{i=1}^{B}$
    Forward pass to compute the representations: $(\boldsymbol{z}_1 \ldots \boldsymbol{z}_B) \leftarrow (f_\theta(\boldsymbol{x}_1) \ldots (f_\theta(\boldsymbol{x}_B))$
Flat Loss   Compute the Task loss: $\ell_{\text{Flat}}(g_w(\boldsymbol{z}_i), y_i)$
Euclidean to Poincare   Get hyperbolic representations using exp. map (eq. (6)): $\tilde{\boldsymbol{z}}_i \leftarrow \exp_{\mathbf{0}}^{c}(\boldsymbol{z}_i)$
Centroid   Calculate class prototypes using hyp. Averaging (eq. (8)): $\omega_i \leftarrow \text{HypAve}_K(\tilde{\boldsymbol{z}}_1^v, \ldots \tilde{\boldsymbol{z}}_j^v)$
Poincare Distance   Compute pairwise hyp. distances (eq. (5)) $\forall v_i, v_j \in V : \rho(v_i, v_j) \leftarrow d_{\mathbb{B}_c}(\omega_i, \omega_j)$
    Get hyp. CPCC loss (eq. (3): $\text{HypCPCC}(d_{\mathcal{T}}, \rho)$
Root Centering   Compute hyp. centering loss using (Equation (8)): $\ell_{\text{center}} = \|\text{HypAve}_B(\tilde{\boldsymbol{z}}_1, \ldots, \tilde{\boldsymbol{z}}_B\|)$
    Get total loss using Equation (10): $\mathcal{L}(\mathcal{D}_B)$
    Compute Gradients for learnable parameters at time t: $\mathbf{u_t}(\theta, w) \leftarrow \nabla_{\theta,w}\mathcal{L}(\mathcal{D}_B)$
    Refresh the parameters: $(\theta, w)_{t+1} \leftarrow (\theta, w)_t - \frac{\eta}{B}\mathbf{u_t}(\theta, w)$

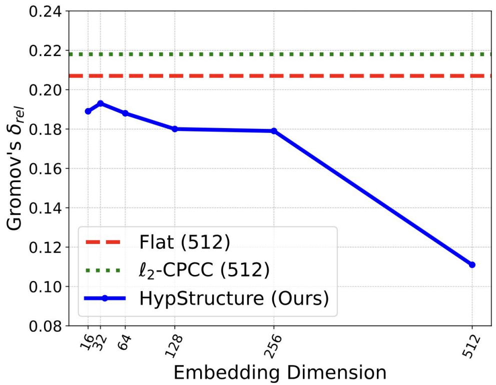
Output: $(\boldsymbol{z}_1, \ldots \boldsymbol{z}_N); \theta, w$

Results: Classification and Embedding Hierarchy

NEURAL INFORMATION PROCESSING SYSTEMS

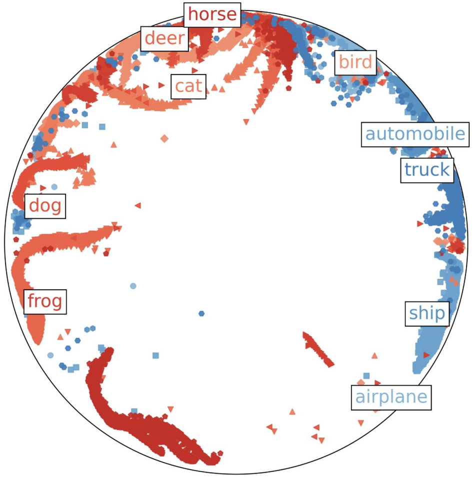
- Experiments on three benchmark datasets: CIFAR10, CIFAR100, ImageNet100
- Compared to Flat and ℓ_2-CPCC [Zeng et. al, 2022]
  - Reduced distortion in embedding the hierarchy (Gromov's **δ** and CPCC), even in low-dimensional regimes → more *tree-like* features
  - Improved Coarse and Fine Classification accuracies

Dataset (Backbone)	Method	Distortion of Hierarchy		Classification Accuracy	
		δ_{rel} (↓)	CPCC (↑)	Fine (↑)	Coarse (↑)
CIFAR10 (ResNet-18)	Flat	0.232 (0.001)	0.573 (0.002)	94.64 (0.12)	99.16 (0.04)
	ℓ_2-CPCC	0.174 (0.002)	0.966 (0.001)	94.47 (0.13)	98.91 (0.02)
	HypStructure	**0.094 (0.003)**	**0.992 (0.001)**	**94.79 (0.14)**	**99.18 (0.04)**
CIFAR100 (ResNet-34)	Flat	0.209 (0.002)	0.534 (0.119)	74.96 (0.14)	84.15 (0.19)
	ℓ_2-CPCC	0.213 (0.006)	**0.779 (0.002)**	76.07 (0.19)	85.28 (0.32)
	HypStructure	**0.127 (0.016)**	0.766 (0.007)	**76.68 (0.22)**	**86.01 (0.13)**
ImageNet100 (ResNet-34)	Flat	0.168 (0.003)	0.429 (0.002)	90.01 (0.07)	90.77 (0.11)
	ℓ_2-CPCC	0.213 (0.009)	0.834 (0.002)	89.57 (0.38)	90.34 (0.28)
	HypStructure	**0.134 (0.001)**	**0.841 (0.001)**	**90.12 (0.01)**	**90.84 (0.02)**

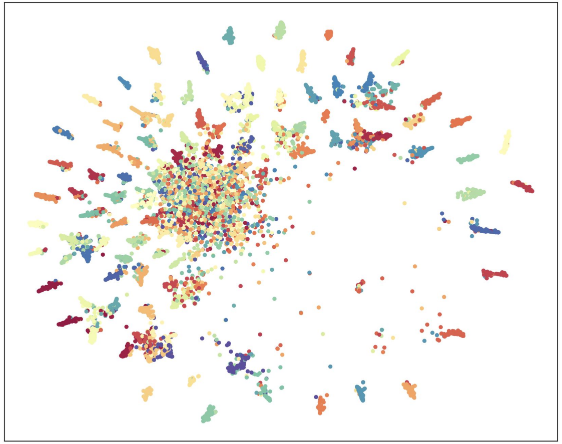


Visualization: Learnt Representations

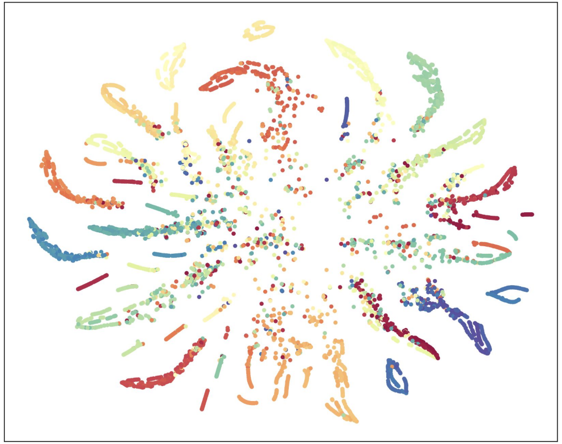
- Qualitative analysis of the learnt representations
  - Fine classes arrange on the Poincare disk according to the hierarchy
  - **HypStructure** → leads to sharper and more discriminative features
  - Fine classes of the same coarse parent (same shade of color) are grouped closer



Hyperbolic UMAP: HypStructure on CIFAR10



tSNE: Flat on CIFAR100



tSNE: HypStructure on CIFAR100

Results: OOD Detection

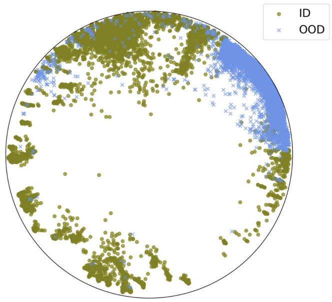
- Out-of-Distribution (OOD) detection: detection of samples that do not belong to the in-distribution (ID)
- Mahalanobis Score:

$$s(\boldsymbol{x}) = (f(\boldsymbol{x}) - \mu)^\top \Sigma^{-1}(f(\boldsymbol{x}) - \mu)$$

- Experiments on 9 real-world OOD datasets for 3 ID datasets with **`HypStructure`**:
  - Consistent improvement in the OOD detection AUROC across OOD datasets
  - Improvement in the ID vs OOD feature separation in the Poincare Disk

Method	AUROC	Method	AUROC	Method	AUROC
CIFAR10		CIFAR100		ImageNet100	
SSD+	97.38	SSD+	85.90	SSD+	92.46
KNN+	97.22	KNN+	86.14	KNN+	92.74
ℓ_2-CPCC	76.67	ℓ_2-CPCC	85.26	ℓ_2-CPCC	91.33
HypStructure	**97.75**	HypStructure	**88.21**	HypStructure	**93.83**

Average AUROC of OOD Detection using Mahalanobis Distance



Hyperbolic UMAP using HypStructure: CIFAR100 (ID) vs SVHN (OOD)

Theoretical Analysis

NEURAL INFORMATION PROCESSING SYSTEMS

- Motivation: **`HypStructure`** with Mahalanobis score leads to improved OOD detection.

$$s(\boldsymbol{x}) = (f(\boldsymbol{x}) - \mu)^\top \Sigma^{-1} (f(\boldsymbol{x}) - \mu)$$

- Main Theorem: Existence of eigenvalue gaps between each level of hierarchy for CPCC-based representations.
  - Representation Matrix Z : n x d, Kernel Matrix K = ZZ$^\top$: n x n.

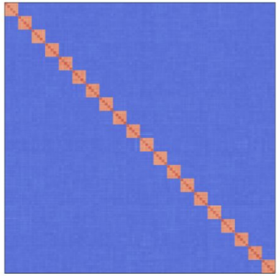
Theorem 5.1 (Eigenspectrum of Structured Representation with Balanced Label Tree). *Let $\mathcal{T}$ be a balanced tree with height H, such that each level has C_h nodes, $h \in [0, H]$. Let us denote each entry of K as r^h where h is the height of the lowest common ancestor of the row and the column sample. If $r^h \geq 0, \forall h$, then: (i) For $h = 0$, we have $C_0 - C_1$ eigenvalues $\lambda_0 = 1 - r^1$. (ii) For $0 < h \leq H - 1$, we have $C_h - C_{h+1}$ eigenvalues $\lambda_h = \lambda_{h-1} + (r^h - r^{h+1})\frac{C_0}{C_h}$. (iii) The last eigenvalue is $\lambda_H = \lambda_{H-1} + C_0 r^H$.*

- Theorem A.2: This statement can be generalized to arbitrary label tree.

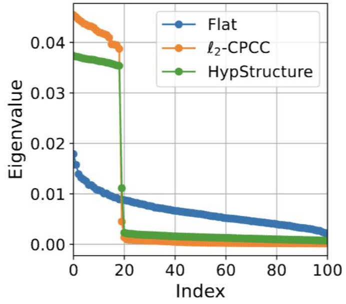
Phase Transition Pattern

- Main Theorem: Existence of eigenvalue gaps **between each level of hierarchy** for CPCC-based representations.

- Example: CIFAR100
  - 20 coarse classes
  - 1 coarse class → 5 fine classes

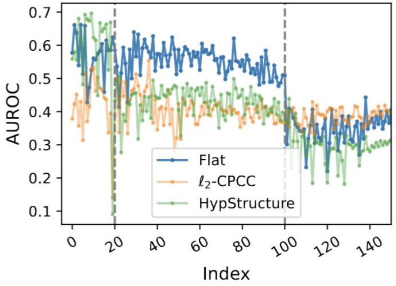


Kernel Matrix K



Eigenspectrum of K

- Implication: Coarse directions might be sufficient for OOD detection.



CIFAR100 vs SVHN with top k-th principal component

Summary, Contributions and Open Questions

- **`HypStructure`**:
  - Hyperbolic structured regularization approach to accurately and explicitly embed the label hierarchy, address the shortcomings of ℓ_2-CPCC
  - Effective for both full training and fine-tuning models across classification, hierarchy embedding and OOD detection tasks
  - More interpretable and *tree-like* representations
  - Formal analysis of the eigenspectrum of *hierarchy-informed* features

- Open Questions:
  - Understanding the impact of noisy hierarchies
  - Using different models of hyperbolic geometry
  - Error bounds of CPCC style structured regularization objectives

Thanks!