Exploring the Manifold of Neural Networks Using Diffusion Geometry

Elliott Abel[1*] Peyton Crevasse[2*] Yvan Grinspan[3*] Selma Mazioud[2*]
Folu Ogundipe[2*] Kristof Reimann[2*] Ellie Schueler[4*] Andrew J. Steindl[5*]
Ellen Zhang[2*] Dhananjay Bhaskar[2] Siddharth Viswanath[2]
Yanlei Zhang[6] Tim G. J. Rudner[7] Ian Adelstein[2] Smita Krishnaswamy[2]

[1]Pomona College [2]Yale University [3]Amherst College [4]Bates College
[5]Vassar College [6]Mila - Quebec AI Institute [7]New York University

Abstract

Drawing motivation from the manifold hypothesis, which posits that most high-dimensional data lies on or near low-dimensional manifolds, we apply manifold learning to the space of neural networks. We learn manifolds where datapoints are *neural networks* by introducing a distance between the hidden layer representations of the neural networks. These distances are then fed to the non-linear dimensionality reduction algorithm PHATE to create a manifold of neural networks. We characterize this manifold using features of the representation, including class separation, hierarchical cluster structure, spectral entropy, and topological structure. Our analysis reveals that high-performing networks cluster together in the manifold, displaying consistent embedding patterns across all these features. Finally, we demonstrate the utility of this approach for guiding hyperparameter optimization and neural architecture search by sampling from the manifold.

1 Introduction

The development of neural networks has revolutionized various fields by allowing models to learn intricate relationships, patterns, and underlying structures directly from data [LBH15; He+15; SLD17; Sze+16]. However, understanding how different network architectures represent and process information, particularly in high-dimensional spaces, remains under-explored. This challenge grows more relevant as we design larger neural networks for which finding architectures and hyperparameters becomes increasingly taxing. Here we hypothesize that neural networks can be largely characterized by the data representations they create. We therefore create a manifold or space of neural networks organized by their hidden layer representations of a dataset. We show that well-performing neural networks, regardless of hyperparameter selections, lead to similar representations [Zha+21; PHD20; MSM18].

Our approach first collects point cloud representations of n datapoints from hidden layers of a collection of neural networks and represents each point cloud as a diffusion matrix. In such a matrix, each data point is characterized by its transition probability to other data points. This gives us a representation of each neural network as an $n \times n$ diffusion operator, independent of architectural variations in hidden layer dimensions. Then to define a distance between two such neural networks, we utilize the Frobenius norm between the respective diffusion matrices. We pass these distances to PHATE [Moo+19] (Potential of Heat-diffusion for Affinity-based Transition Embedding), which embeds the neural networks into a lower-dimensional space organized by their hidden representations of the data. This mapping allows us to uncover features of the neural networks' hidden layer representations that correlate with network performance, accuracy, and other key metrics. We are then able to explore the space of neural networks in a more structured and interpretable way.

[*] Student authors, ordered alphabetically.

When characterizing this manifold by test performance of the neural networks, we observe contiguous regions of the manifold space that are occupied by high-performing networks. This implies that high-performing neural networks have similar hidden layer representations. We found that high-performing networks were located closer to each other in the manifold than in alternative embedding methods by computing their normalized average pairwise distance in PHATE embeddings. To understand the source of this similarity, we compare these representations by their 1) class separation, 2) hierarchical cluster structure, 3) diffusion spectral entropy, and 4) persistence homology when viewed as a graph. We find that high-performing networks exhibit similarity across all four categories.

To summarize, the key contributions of this paper are as follows:

1. We construct a manifold of neural networks based on their hidden representations of data.
2. We characterize the regions of the manifold based on similarity in class separation, hierarchical cluster structure, entropy, and persistence homology.
3. We show how we can apply our method to searching for hyperparameters and architectures of neural networks based on sampling from the manifold.

We envision this approach as a step towards improving our understanding of the interplay between neural network representations, neural network hyperparamters, and predictive performance.

2 Background

In this section, we provide background on data diffusion, diffusion maps and PHATE which we utilize for our neural network manifold construction.

2.1 Computation of the Diffusion Operator

Given a point cloud $\boldsymbol{X}$ containing n points, the diffusion operator is computed by first calculating the pairwise distances between points and then constructing a similarity or adjacency matrix. We define a kernel function $\kappa : \mathbb{R}^d \times \mathbb{R}^d \to \mathbb{R}^+$ and use it to populate the adjacency matrix $\boldsymbol{W}$, where each element is defined as $\boldsymbol{W}_{ij} := \kappa(x_i, x_j)$ for all points $x_i, x_j \in \boldsymbol{X}$. Here, we employ a Gaussian kernel to construct the adjacency matrix. The resulting weighted graph $\mathcal{G}$, with its set of nodes representing observations and adjacency matrix values in $(0, 1]$, forms the basis for defining the diffusion operator. The diffusion operator is given by $\boldsymbol{P} := \boldsymbol{Q}^{-1}\boldsymbol{W}$, where $\boldsymbol{Q}$ is the degree matrix. This operator describes a random walk or diffusion on the graph, making it a fundamental tool for uncovering the structure of datasets in manifold learning.

2.2 Diffusion Maps

Based on the diffusion operator, Coifman and Lafon [CL06] defines an embedding in k dimensions via the first k non-trivial right eigenvectors of the t-steps diffusion operator $\boldsymbol{P}^t$ weighted by their eigenvalues. The embedding preserves the ***diffusion distance*** $DM_{\boldsymbol{P}}(x_i, x_j) := \|(\boldsymbol{\delta_i P}^t - \boldsymbol{\delta_j P}^t)(1/\pi)\|_2$, where $\boldsymbol{\delta}_i$ is a vector such that $(\boldsymbol{\delta}_i)_j = 1$ if $j = i$ and 0 otherwise, and π is the stationary distribution of $\boldsymbol{P}$. Intuitively, $DM_{\boldsymbol{P}}(x_i, x_j)$ considers all the t-steps paths between x_i and x_j. A larger diffusion time can be seen as a low-frequency graph filter, keeping only information from low-frequency transitions such as the stationary distributions. The diffusion matrix converges to a stationary distribution $\pi_i := \boldsymbol{Q}_{ii} / \sum_i \boldsymbol{Q}_{ii}$ under mild assumptions, making diffusion with $t > 1$ useful in denoising relationships between observations.

2.3 PHATE

PHATE is a diffusion-based method that preserves an information-theoretic ***potential distance*** $PH_{\boldsymbol{P}} := \| - \log \boldsymbol{\delta}_i \boldsymbol{P}^t + \log \boldsymbol{\delta}_j \boldsymbol{P}^t\|_2$ (Moon et al. [Moo+19]) and justifies this approach using the log transformation to prevent nearest neighbors from dominating the distances. Unlike diffusion maps, this method allows us to capture manifolds in very low dimensions (in particular, 2 or 3 for visualization). In our neural network manifold construction, we represent each neural network using a data diffusion operator, and then use a distance between the operators themselves to create a PHATE embedding of the space of neural networks. Note that PHATE also uses a diffusion operator internally which, in this manuscript, operates on "datapoints" that each represent a neural network.

3 Related Work

Our work builds on a rich body of research that examines and leverages the internal representations neural networks develop during training. For instance, Gigante et al. [Gig+19] employs PHATE to visualize how these representations evolve across network layers, track changes throughout training, and observe class-based variation in classification networks. Similarly, Rossem and Saxe [RS24] investigates how latent embeddings are affected by various hyperparameters, exploring the extent to which these traits are preserved across different network architectures.

The challenge of comparing latent embeddings has also seen considerable focus. For example, Wang et al. [Wan+18] explores subspace alignment techniques but finds they yield inconsistent structures when the same network is trained with different initializations. Canonical Correlation Analysis (CCA)-based methods have been explored by Kornblith et al. [Kor+19], who highlight desirable properties in similarity measures between learned representations, particularly those invariant to transformations. Persistent homology, a topological data analysis technique introduced by Carlsson [Car20], has also been applied in this domain. Moor et al. [Moo+20] designs an autoencoder that incorporates persistent homology changes between latent and ambient spaces within its loss function, while Wayland et al. [WCR24] leverages persistent homology to compare neural networks, creating a “multiverse” structured by differences in persistence landscapes—an approach similar to ours, which employs persistent homology along with hierarchical microstructure and diffusion spectral entropy.

Beyond empirical studies, theoretical investigations have examined neural network geometry. For example, extensions of the Johnson-Lindenstrauss lemma to non-linear contexts, such as fully connected and convolutional neural networks, reveal intriguing results: under certain conditions, natural image-based data maintains its geometric structure, measured by cosine similarity, while Gaussian inputs tend to exhibit contraction [Nac+22]. Such findings underscore the potential for novel research into how network layers transform input data geometry, adding valuable perspective to our understanding of neural network function.

4 Methods

We introduce the following notation. Let $Y = \{y_i\}_{i=1}^n$ be a set of n test points. Throughout the manuscript, we assume that the points in Y are registered, that is, that they remain consistent and in the same order throughout. Now, consider a set of m neural networks, denoted $Z = \{F_i(X, \theta_i)\}_{i=1}^m$, where each F_i is a neural network trained on a dataset X, and θ_i represents the hyperparameters. For each neural network F_i, we define an embedding function ϕ_{F_i} from the input space to the hidden layer's embedding space $\mathbb{R}^{d_i}$ of network F_i. The hidden layer embeddings that we compare across networks are given by $\{\phi_{F_i}(Y)\}_{i=1}^m$, the collection of activations in a specific hidden layer resulting from passing the test points Y into various networks F_i.

4.1 The Manifold of Neural Networks

Given the set of neural networks Z, our embedding method uses the set Y of test points to obtain a representative embedding for each neural network. For a neural network $F_z \in Z$, the embedding $\phi_{F_z}(y_k)$ is extracted for each data point $y_k \in Y$. The resulting embeddings are concatenated by rows for all $k \in Y$, forming an $n \times d_z$ matrix E_z, where d_z is the dimensionality of a hidden layer embedding for network F_z. The pairwise Euclidean distances between the embeddings are then computed, resulting in an $n \times n$ distance matrix D_z, where each entry $D_{z,ij} = d(\phi_{F_z}(y_i), \phi_{F_z}(y_j))$.

Next, D_z is passed through a standard Gaussian kernel (with $\sigma = 0.5$) to obtain an $n \times n$ pairwise similarity

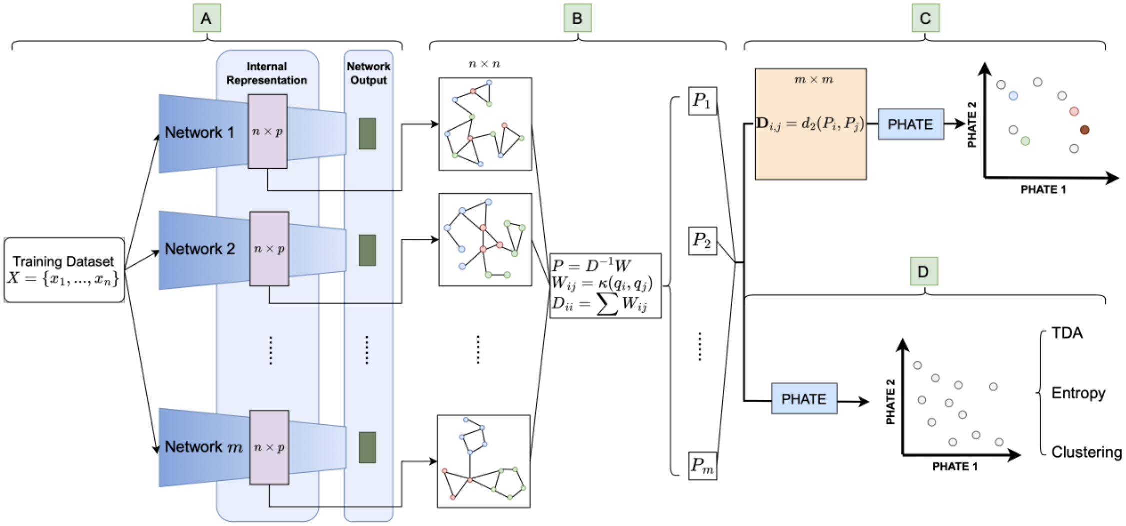


Figure 1: Schematic of our method. (A) Train neural networks and retrieve their internal representation; (B) Construct graphs with edges representing pairwise diffusion probabilities; (C) Construct distance matrix of network representations and visualize network landscape for downstream tasks (D) Visualize the internal representation for characterization via topological data analysis, spectral entropy, and hierarchical clustering.

matrix W_z:

$$W_{z,ij} = \exp\left(-\frac{d(\phi_{F_z}(y_i), \phi_{F_z}(y_j))^2}{2\sigma^2}\right),$$

representing the edge weight between vertices y_i and y_j based on the distances of their hidden layer activations. The matrix W_z is a fully connected graph representation of the neural network F_z. To ensure that the metric adheres to the underlying manifold of the data, W_z is row-normalized resulting diffusion operator $P_z := Q_z^{-1} W_z$, where $Q_{z,ij} = \delta_{ij} \sum_j W_{z,ij}$ is a continuous analog of the degree matrix.

This transformation from the hidden representation matrix E_z to the diffusion operator P_z is repeated for all neural networks $F_z \in Z$. These diffusion operators encapsulate how each neural network classifies data by describing the intrinsic geometry of their representations. Once these matrices are computed, we define the distance between neural networks as the similarity or dissimilarity of their diffusion operators P_z. We therefore compute the pairwise Frobenius norm between each pair of diffusion operators from the trained neural networks, resulting in the $m \times m$ distance matrix N where $N_{ij} = \|P_i, P_j\|_F$. Using the non-linear dimension reduction technique PHATE to represent the distances from N, we are able to visualize the landscape of neural networks. This process is depicted in Figure 1 A-C.

4.2 Characterizing the Landscape of Neural Networks

We characterize the manifold of neural networks over their internal representation using class separation, hierarchical cluster microstructure, diffusion spectral entropy, and persistence homology. Figure 2 shows the hidden layer representations of neural networks at varying accuracies. Such attributes have strong relationships with network stability and predictive accuracy while being largely impartial to hyperparameter adjustment.

Class Separation and Variance. We characterize networks primarily by the degree of class separation within datasets that have classification labels. Intuitively, high classification accuracy requires well-separated classes, so the extent of this separation serves as a key indicator of network performance.

To quantify this separation, we measure the similarity in class distinctness by constructing a distance matrix between the centroids of each class. For a given class C_k, its centroid is calculated as

$$\mu_k = \frac{1}{|C_k|} \sum_{x \in C_k} \phi(x),$$

where $\phi(x)$ represents the embedding of data point x. Pairwise Euclidean distances between these centroids provide an overall sense of class structure within the representation layer. Additionally, to evaluate within-class variance, we compute the average Euclidean distance from each point in a class to its respective centroid μ_k. This within-class variance serves as a measure of compactness for each class embedding.

Hierarchical Clustering Structure. Advancing our understanding of the cluster structures formed by neural networks is essential to uncover how internal representations relate to network accuracy. We denote the hidden layer representation of n aligned data points as E_z for all $F_z \in Z$ and disregard any true labels. Using this representation, we apply the Ward agglomerative clustering method from [ML14], which defines inter-cluster distances based on the increase in the sum of squares upon merging clusters.

After performing agglomerative clustering, we generate dendrograms for each model to analyze the microstructure of clusters within the network's representations. These dendrograms offer insights into the organization of the network's internal representations. To quantify the accuracy of these cluster structures, we compute the Adjusted Rand Index (ARI). By cutting each dendrogram at 10 clusters, we assess the similarity of each E_z clustering structure to the ground truth labels through ARI, which reveals how closely each network's clusters align with true label groups.

Additionally, we calculate pairwise ARI values between the dendrograms of any two networks, F_n and F_m, to assess similarity in their microstructures. This enables us to explore correlations between microstructure similarity and network accuracy across different networks.

Information Theory and Diffusion Spectral Entropy. The diffusion matrix of an embedding effectively captures its geometry. Looking at hidden layer embeddings of different neural network architectures, we use an information theoretic measure called Diffusion Spectral Entropy (DSE), which was introduced in Liao et al. [Lia+23] to compute a fast entropy on data by first creating a data diffusion operator, and then computing the entropy of the eigenvalues. The DSE is defined as the entropy of the eigenvalues of the diffusion operator P_z computed on a dataset Y. $S_D(P_z, t) := -\sum_i \alpha_{i,t} \log(\alpha_{i,t})$ where $\alpha_{i,t} := \frac{|\lambda_i^t|}{\sum_j |\lambda_j^t|}$, and $\{\lambda_i\}$ are the eigenvalues of the diffusion matrix P_z and t is the diffusion time. Defined this way, the DSE captures the spread of pointcloud data over its dimensions. A high DSE would correspond to an even spread across diffusion dimensions.

Homology and Wasserstein Distance. We also compute up to second order persistence homology of the hidden layer representations to assess similarity across higher homological dimensions. For a comprehensive overview of persistence homology and topological data analysis, we refer readers to Zomorodian and Carlsson [ZC05]. Notably, zero-th order homology captures connected components, which, due to varying scales, may show similarities to hierarchical clustering results. Higher-order homology captures loops, voids, and additional dimensions of "holes," providing unique insights into the data's structure.

To compare two persistence diagrams, we use the Wasserstein distance [PC19], commonly employed in topological data analysis [Cab+21] due to its robustness to noise and preservation of topological features. Prominent topological features, which appear as longer intervals on persistence diagrams, are thus given more weight in this distance calculation. The Wasserstein p-distance between two measures μ and ν with finite p-moments is defined as:

$$\mathscr{W}_p(\mu, \nu) = \left(\inf_{\gamma \in \Gamma(\mu,\nu)} \int_{M \times M} d(x,y)^p \, d\gamma(x,y) \right)^{1/p}, \tag{1}$$

where $\Gamma(\mu, \nu)$ is the set of all couplings of μ and ν.

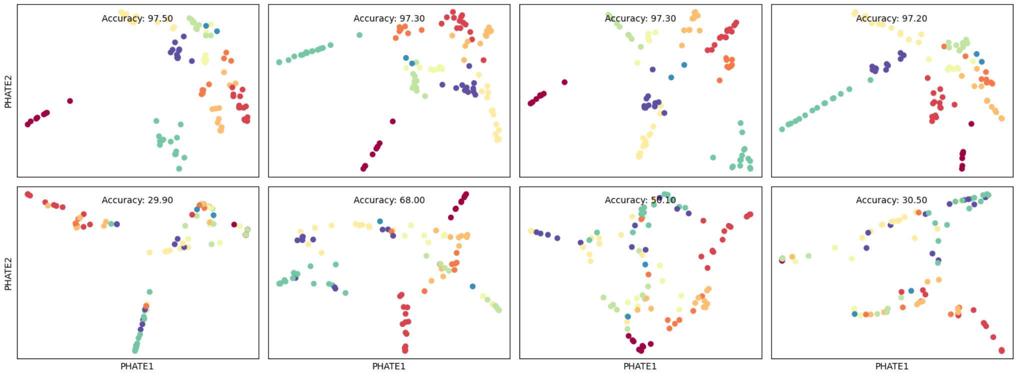


Figure 2: PHATE hidden layers of neural networks trained for MNIST classification for 4 high-performing and 4 low-performing models.

Table 1: Normalized average pairwise distances between PHATE embeddings of the N highest-performing MLPs trained to classify MNIST. Distances are defined by Frobenius norms, first between diffusion matrices P_z, then between distance matrices D_z, then between k-NN adjacency matrices.

	Diffusion Matrices	**Distance Matrices**	**k-NN Matrices**
Top 10	0.0790	0.2349	0.1788
Top 20	0.1347	0.4449	0.2257
Top 30	0.1373	0.4751	0.3312

5 Results

5.1 Evaluating the Manifold of Neural Networks

Here we compare our manifold construction technique, described in Figure 1, with another more rudimentary method: computing the pairwise Frobenius norm between the raw pairwise distance matrices D_z for the internal representations. This process is identical to our chosen method, except that we do not apply a Gaussian kernel onto and do not normalize the pairwise distance matrix to produce the diffusion matrix. In the second alternative method, we first construct a k-Nearest Neighbor graph and then compute the pairwise Frobenius norms between the resulting adjacency matrices. To assess our manifold construction, we compute the average pairwise distance between PHATE embeddings of the top N highest-accuracy of 200 MLPs trained for MNIST classification. To normalize the distances for the purpose of comparison across methods, we divide by the average pairwise distance between networks for that manifold. We observed that for each of the Top- 10, 20, and 30 highest-accuracy networks, our method mapped the networks much closer together.

5.1.1 Architecture Search

We hypothesized that hyperparameters extrapolated from models in the high-accuracy region of learned manifold could lead to high-performing models. To test this hypothesis, we selected the 30 models with the highest accuracies from our set of 200 MLPs trained on MNIST. For these models, we identified the two most frequent combinations of weight decay and momentum, corresponding to the most common learning rate. We then trained an MLP with the same architecture using the average of these two most frequent values for weight decay and momentum and achieved a test accuracy of 97.04% on a new set of hyperparameters. This demonstrates that our approach is a promising method for hyperparameter search. Further discussion, including how this approach may generalize across architectures, is provided in the Appendix.

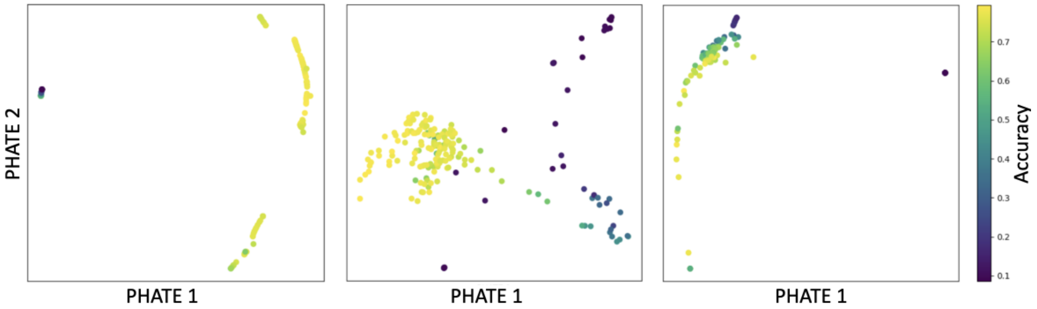


Figure 3: PHATE landscapes of neural networks revealing structure by accuracy (color). **Left:** MLPs for MNIST classification. **Middle:** ResNets for CIFAR-10 classification. **Right:** CNNs for CIFAR-10 classification.

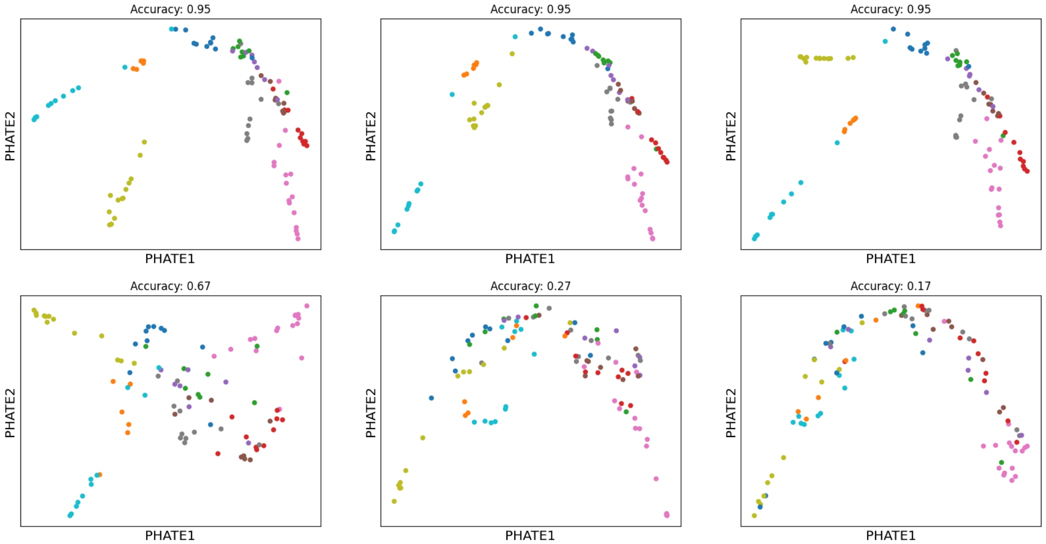


Figure 4: PHATE visualizations of embedding space for three high performing ResNets (top) and three random ResNets (bottom).

5.2 Characterizing the Landscape of Neural Networks

We performed the characterization tests on three groups of 200 neural networks: ResNet-9s pre-trained on CIFAR-10 [Kri09] and finetuned for two epochs, MLPs trained to classify MNIST [Den12], and simple CNNs trained to classify CIFAR-10. All models had fixed initialization but varied in learning rate, momentum, and weight decay, with unique hyperparameter combinations for each model.

Class Structure. In the internal representation space, we assign each of the 100 test points a class label. We find that high accuracy correlates with larger average pairwise distances between class centroids (ResNet: $r^2 = 0.89$, CNN: $r^2 = 0.74$, MLP: $r^2 = 0.89$).

Pairwise distances between the 10 class centroids are nearly identical for higher-performing models ($A > 93\%$). This result holds despite the fact that the networks were trained on notably differing hyperparameters (learning rate $\texttt{lr} \in (0.005, 0.15)$, momentum $\texttt{m} \in (0.5, 0.9)$, weight decay $\texttt{wd} \in (10^{-5} - 10^{-2})$). Figure 4 and Figure 5 demonstrate that accurate networks learn similar embeddings, where the same classes are consistently structured in similar ways within the embedding space.

We also analyze variance within classes to measure how tightly a network clusters data from the same class. Intra-class variance is comparable across the highest-performing models within each architecture. The following statistics pertain to ResNets, which exhibited the most significant accuracy range; however, similar trends were observed in CNNs and MLPs.

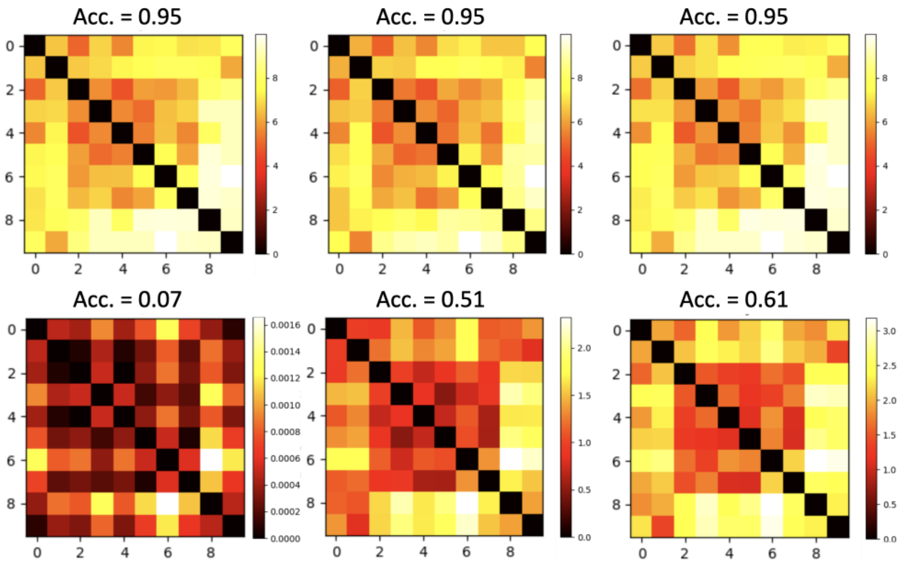


Figure 5: Class structure heatmaps for three high performing ResNets (top row) and three ResNets (bottom row). Structure for high performing models is nearly indistinguishable across all pairwise class distances. For random models, patterns are ambiguous and there is less overall separation between classes.

ResNet models were grouped into 3% accuracy ranges, and for each range, we computed the average class variance and standard deviation across all classes in the dataset. The highest-performing models ($A = 93\% - 96\%$) demonstrated the lowest standard deviation in variance across classes (range: $0.36 - 0.52$, $\mu = 0.42$). Models with slightly lower accuracy ($90\% - 93\%$) exhibited higher variance (range: $0.76 - 1.03$, $\mu = 0.857$), with this trend persisting as accuracy decreased. Additionally, we find that average class variance is correlated with higher neural network accuracy (ResNet: $r^2 = 0.84$, CNN: $r^2 = 0.64$, MLP:$r^2 = 0.91$). This trend can also be observed in each class in a dataset individually.

In the internal representation space, most accurate models have classes that are farther apart and less compact. This observation is invariant to our hyperparameter adjustments and are consistent with the highest preforming models, less so as performance decreases.

Hierarchical Cluster Structure. Hierarchical clustering analysis shows whether there are structural similarities between embeddings at finer levels of granularity than under class separation. We quantify hierarchical cluster structure with the Adjusted Rand Index (ARI) and find that when computed from a single model's embedding space with respect to the ground truth, higher ARI is correlated with higher network accuracy (ResNet: $r^2 = 0.92$, CNN: $r^2 = 0.71$, MLP: $r^2 = 0.89$).

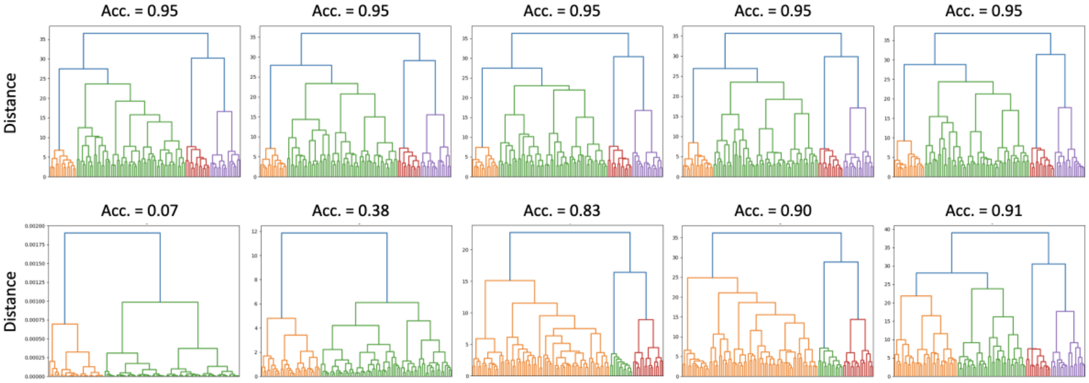


Figure 6: Dendrograms depicting microstrcuture of five high performing ResNet models (top row) and five random ResNet models (bottom row)

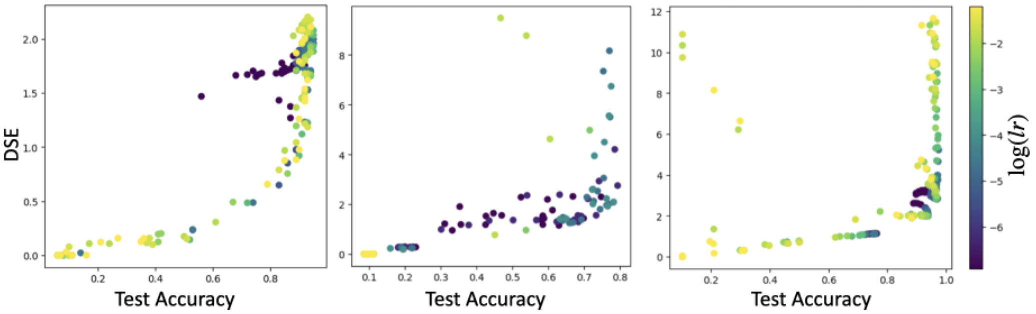


Figure 7: Diffusion Spectral Entropy increases with test accuracy across the three architectures (MLP, ResNet, CNN). Points are colored by learning rate.

The overall cluster structure of each model's internal representation of the same 100 test points can be visualized through dendrograms and compared numerically through pairwise ARI. For ResNets, the similarity amongst the hierarchical clustering structure of highest-performing networks is evidenced in Figure 6. We observe very similar overall "microstructure" between highest-performing models. These patterns are invariant across hyperparameters and the similarity in microstructure certainly does not hold as accuracy decreases as evidenced in the bottom row of the figure. Pairwise ARI between models is higher for better performing models than that of moderate and poorly-performing models. Using the 3% accuracy batches from earlier we find a moderate correlation between accuracy range midpoint and average pairwise ARI (ResNet: $r^2 = 0.60$, CNN: $r^2 = 0.52$) and a strong correlation for the simplest class of networks (MLP: $r^2 = 0.97$). Our analysis shows that the highest-performing models cluster data closer to ground truth labels and have similar microstructures, while moderate and poorly performing models are more divergent.

Diffusion Spectral Entropy High-performance networks across architectures tend to have higher Diffusion Spectral Entropy (DSE) associated with their internal representations, using $t = 0$ because of the low noise in the dataset. Figure 7 indicates this positive relationship between network performance and spread across diffusion matrix dimensions.

These results indicate that high-performing networks exhibit more pronounced cluster structures in their hidden layer embeddings. This pattern is in agreement with the findings of Liao et al. [Lia+23]: for k separated clusters, they prove that as t grows to infinity, $S_D(P_z, t)$ tends to $\log(k)$. This makes sense because for graphs that have disconnected components, we would expect to have multiple eigenvectors with non-trivial eigenvalues, or a more even spread of these eigenvalues across the dimensions represented by the eigenvectors. Therefore, we can intepret DSE as measuring the number of significant eigendirections in the data. On the other hand, if the dataset only forms one uniform cluster, since it can be represented by a fully connected graph (i.e. an ergodic process), we would expect the DSE to approach 0 as t tends to infinity since we would have only one non-trivial eigendirection. In other words, the diffusion matrix that represents such data would approach a rank 1 steady-state matrix.

Persistent Homology Across architectures, we have found that topological features can characterize neural network accuracy, independent of hyperparameters, in the sense that Wasserstein distances between persistence diagrams of internal representations are smaller for high preforming networks. Using PHATE to visualize these Wasserstein distances, Figure 8 illustrates our observed result for 200 Convolution Neural Networks, coloring by accuracy.

Figure 9 summarizes these findings, illustrating through distance heatmaps how accuracy shapes topology across each architecture observed. This result aligns closely with the structural characterization findings above, suggesting that the way a successful neural network clusters its data is not only geometrically but also topologically identifiable and remains consistent across varying hyperparameters.

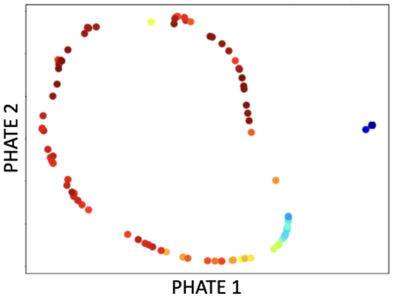
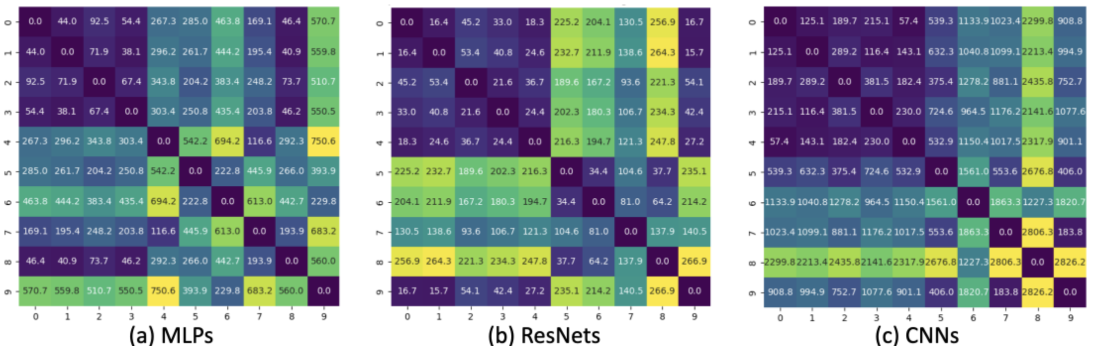


Figure 8: PHATE plot of 200 CNNs colored by their accuracies. Wasserstein Distance between persistence diagrams are used as the distance metric.



	0	1	2	3	4	5	6	7	8	9
0	0.0	44.0	92.5	54.4	267.3	285.0	463.8	169.1	46.4	570.7
1	44.0	0.0	71.9	38.1	296.2	261.7	444.2	195.4	40.9	559.8
2	92.5	71.9	0.0	67.4	343.8	204.2	383.4	248.2	73.7	510.7
3	54.4	38.1	67.4	0.0	303.4	250.8	435.4	203.8	46.2	550.5
4	267.3	296.2	343.8	303.4	0.0	542.2	694.2	116.6	292.3	750.6
5	285.0	261.7	204.2	250.8	542.2	0.0	222.8	445.9	266.0	393.9
6	463.8	444.2	383.4	435.4	694.2	222.8	0.0	613.0	442.7	229.8
7	169.1	195.4	248.2	203.8	116.6	445.9	613.0	0.0	193.9	683.2
8	46.4	40.9	73.7	46.2	292.3	266.0	442.7	193.9	0.0	560.0
9	570.7	559.8	510.7	550.5	750.6	393.9	229.8	683.2	560.0	0.0

(a) MLPs

	0	1	2	3	4	5	6	7	8	9
0	0.0	16.4	45.2	33.0	18.3	225.2	204.1	130.5	256.9	16.7
1	16.4	0.0	53.4	40.8	24.6	232.7	211.9	138.6	264.3	15.7
2	45.2	53.4	0.0	21.6	36.7	189.6	167.2	93.6	221.3	54.1
3	33.0	40.8	21.6	0.0	24.4	202.3	180.3	106.7	234.3	42.4
4	18.3	24.6	36.7	24.4	0.0	216.3	194.7	121.3	247.8	27.2
5	225.2	232.7	189.6	202.3	216.3	0.0	34.4	104.6	37.7	235.1
6	204.1	211.9	167.2	180.3	194.7	34.4	0.0	81.0	64.2	214.2
7	130.5	138.6	93.6	106.7	121.3	104.6	81.0	0.0	137.9	140.5
8	256.9	264.3	221.3	234.3	247.8	37.7	64.2	137.9	0.0	266.9
9	16.7	15.7	54.1	42.4	27.2	235.1	214.2	140.5	266.9	0.0

(b) ResNets

	0	1	2	3	4	5	6	7	8	9
0	0.0	125.1	189.7	215.1	57.4	539.3	1133.9	1023.4	2299.8	908.8
1	125.1	0.0	289.2	116.4	143.1	632.3	1040.8	1099.1	2213.4	994.9
2	189.7	289.2	0.0	381.5	182.4	375.4	1278.2	881.1	2435.8	752.7
3	215.1	116.4	381.5	0.0	230.0	724.6	964.5	1176.2	2141.6	1077.6
4	57.4	143.1	182.4	230.0	0.0	532.9	1150.4	1017.5	2317.9	901.1
5	539.3	632.3	375.4	724.6	532.9	0.0	1561.0	553.6	2676.8	406.0
6	1133.9	1040.8	1278.2	964.5	1150.4	1561.0	0.0	1863.3	1227.3	1820.7
7	1023.4	1099.1	881.1	1176.2	1017.5	553.6	1863.3	0.0	2806.3	183.8
8	2299.8	2213.4	2435.8	2141.6	2317.9	2676.8	1227.3	2806.3	0.0	2826.2
9	908.8	994.9	752.7	1077.6	901.1	406.0	1820.7	183.8	2826.2	0.0

(c) CNNs

Figure 9: Heat maps for Wasserstein distance between persistence diagrams of five high performing networks (indices 0-4) and five random networks (indices 5-9).

Overall, our results indicate that high-performing networks exhibit similar representations with comparable micro and macro scale structures, as quantified by both persistent homology and hierarchical clustering. Additionally, these networks display greater class separation, increased within-class variance, and generally higher entropy. This suggests that such embeddings spread data points in meaningful patterns across diffusion dimensions, capturing informative structural relationships.

6 Conclusion

In conclusion, we introduced a novel framework for exploring and understanding the landscape of neural networks by examining their internal representations, with each representation corresponding to a single network. This approach allowed us to define a distance metric between these internal representations, which we then used to embed networks into a low-dimensional manifold. By characterizing the manifold with features such as hierarchical clustering, spectral entropy, and topological structure, we revealed that high-performing networks consistently exhibit similar internal representations.

Acknowledgments

This project was conducted as part of the Summer Undergraduate Math Research at Yale (SUMRY) REU program, supported by NSF DMS (Award 2050398). D.B. acknowledges support from the Yale-Boehringer Ingelheim Biomedical Data Science Fellowship and the Kavli Institute for Neuroscience Postdoctoral Fellowship. I.A. and S.K. acknowledge support from the NSF DMS Grant (Award 2327211). Additionally, S.K. is supported by the NSF CAREER Award (IIS 2047856) and the NSF MEDIUM Grant (IIS 2403317).

References

[Cab+21] G. Cabanes, Y. Bennani, R. Verde, and A. Irpino. *On the use of Wasserstein metric in topological clustering of distributional data*. 2021. arXiv: `2109.04301 [cs.LG]` (cit. on p. 5).

[Car20] G. Carlsson. *Persistent homology and applied homotopy theory*. 2020. arXiv: `2004.00738 [math.AT]` (cit. on p. 3).

[CL06] R. R. Coifman and S. Lafon. "Diffusion maps". *Applied and Computational Harmonic Analysis* 1 (2006) (cit. on p. 2).

[Den12] L. Deng. "The mnist database of handwritten digit images for machine learning research". *IEEE Signal Processing Magazine* 6 (2012) (cit. on p. 7).

[Gig+19] S. Gigante, A. S. Charles, S. Krishnaswamy, and G. Mishne. "Visualizing the PHATE of neural networks". *Advances in neural information processing systems* (2019) (cit. on p. 3).

[He+15] K. He, X. Zhang, S. Ren, and J. Sun. "Deep residual learning for image recognition". *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)* (2015) (cit. on p. 1).

[Kor+19] S. Kornblith, M. Norouzi, H. Lee, and G. Hinton. "Similarity of neural network representations revisited". In: *International conference on machine learning*. PMLR. 2019 (cit. on p. 3).

[Kri09] A. Krizhevsky. "Learning multiple layers of features from tiny images". In: 2009 (cit. on p. 7).

[LBH15] Y. LeCun, Y. Bengio, and G. Hinton. "Deep learning". *Nature* (2015) (cit. on p. 1).

[Lia+23] D. Liao, C. Liu, B. W. Christensen, A. Tong, G. Huguet, G. Wolf, M. Nickel, I. Adelstein, and S. Krishnaswamy. *Assessing neural network representations during training using noise-resilient diffusion spectral entropy*. 2023. arXiv: `2312.04823 [cs.CV]` (cit. on pp. 5, 9).

[MSM18] G. Montavon, W. Samek, and K.-R. Müller. "Methods for interpreting and understanding deep neural networks". *Digital Signal Processing* (2018) (cit. on p. 1).

[Moo+19] K. R. Moon, D. van Dijk, Z. Wang, S. Gigante, D. B. Burkhardt, W. S. Chen, K. Yim, A. v. d. Elzen, M. J. Hirn, R. R. Coifman, N. B. Ivanova, G. Wolf, and S. Krishnaswamy. "Visualizing structure and transitions in high-dimensional biological data". *Nat Biotechnol* 12 (2019) (cit. on pp. 1, 2).

[Moo+20] M. Moor, M. Horn, B. Rieck, and K. Borgwardt. "Topological autoencoders". In: *International conference on machine learning*. PMLR. 2020 (cit. on p. 3).

[ML14] F. Murtagh and P. Legendre. "Ward's hierarchical agglomerative clustering method: which algorithms implement Ward's criterion?" *Journal of Classification* 3 (2014) (cit. on p. 5).

[Nac+22] I. Nachum, J. Hazla, M. Gastpar, and A. Khina. "A Johnson–Lindenstrauss framework for randomly initialized CNNs". *ICLR 2022* (2022) (cit. on p. 3).

[PHD20] V. Papyan, X. Y. Han, and D. L. Donoho. "Prevalence of neural collapse during the terminal phase of deep learning training". *Proceedings of the National Academy of Sciences* 40 (2020) (cit. on p. 1).

[PC19] G. Peyré and M. Cuturi. "Computational optimal transport: with applications to data science". *Foundations and Trends® in Machine Learning* 5-6 (2019) (cit. on p. 5).

[RS24] L. van Rossem and A. M. Saxe. *When representations align: universality in representation learning dynamics*. 2024. arXiv: `2402.09142 [cs.LG]` (cit. on p. 3).

[SLD17] E. Shelhamer, J. Long, and T. Darrell. "Fully convolutional networks for semantic segmentation". *IEEE Transactions on Pattern Analysis and Machine Intelligence* 4 (2017) (cit. on p. 1).

[Shu+13] D. I. Shuman, S. K. Narang, P. Frossard, A. Ortega, and P. Vandergheynst. "The emerging field of signal processing on graphs: extending high-dimensional data analysis to networks and other irregular domains". *IEEE Signal Processing Magazine* 3 (2013) (cit. on p. 15).

[Sze+16] C. Szegedy, V. Vanhoucke, S. Ioffe, J. Shlens, and Z. Wojna. "Rethinking the inception architecture for computer vision". In: *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. 2016 (cit. on p. 1).

[Wan+18] L. Wang, L. Hu, J. Gu, Z. Hu, Y. Wu, K. He, and J. Hopcroft. "Towards understanding learning representations: to what extent do different neural networks learn the same representation". *Advances in neural information processing systems* (2018) (cit. on p. 3).

[WCR24] J. Wayland, C. Coupette, and B. Rieck. "Mapping the multiverse of latent representations". In: *ICML*. 2024 (cit. on p. 3).

[Zha+21] C. Zhang, S. Bengio, M. Hardt, B. Recht, and O. Vinyals. “Understanding deep learning (still) requires rethinking generalization”. *Commun. ACM* 3 (2021) (cit. on p. 1).

[ZC05] A. Zomorodian and G. Carlsson. “Computing persistent homology”. *Discrete & Computational Geometry* 2 (2005) (cit. on p. 5).

Appendix

A Datasets for the Manifold of Neural Networks

To compare methods for defining the manifold of neural networks, the manifold was computed and visualized for three classes of 200 neural networks: ResNet-9s pre-trained to classify images from the CIFAR-10 dataset and finetuned for two additional epochs, Simple MLPs trained for MNIST classificatioj and Simple CNNs trained for CIFAR-10 classification. Models all had fixed weight initialization but spanned a range of learning rate, momentum, and weight decay with each model having a unique combination of hyperparameters. Below we outline our method using the MNIST dataset as an example, but we used the same set of hyperparameters for stochastic gradient descent optimization on ResNets and CNNs. Code for each is provided in a GitHub repository.

The architecture of the network trained to classify MNIST is an input layer with dimension 784, followed by two fully connected hidden layers with dimension 50 and 20, respectively. The activation function used for both hidden layers is ReLU. The output layer contains 10 neurons, with the softmax activation function. The 200 models for each architecture and task exhausted all combinations of the following hyperparameters: Learning Rates $\in \{0.001, 0.005, 0.01, 0.05, 0.1, 0.15, 0.2, 0.3\}$ Momentum $\in \{0.5, 0.6, 0.7, 0.8, 0.9\}$ Weight Decay $\in \{1 \times 10^{-5}, 1 \times 10^{-4}, 1 \times 10^{-3}, 1 \times 10^{-2}, 0.1\}$.

Machines Our networks were trained using Google Colab Pro (ResNets), NVIDIA a100 GPU (MLPs), and GeForce GTX 1080Ti GPUs (CNNs).

B Manifold Method Comparisons

Alternative 1 - Euclidean Distance Let $z \in [Z]$ be a neural network. Similarly as in section 4.1, we concatenate internal representations a_z^k $\forall k \in [Y]$ to obtain E_z and its $n \times n$ pairwise distance matrix D_z. Here, D_z is our characteristic embedding of neural network z.

We repeat this embedding process $\forall z \in [Z]$. Using these embeddings, we compute the pairwise Frobenius norm between $\{(E_i, E_j) : i, j \in [Z]\}$ for each trained neural network z to obtain the $m \times m$ distance matrix N_1, where $m = |Z|$. PHATE was then used to visualize N_1. Note that this process is identical to our chosen metric, except for neglecting to pass the pairwise distance matrix into a Gaussian kernel.

In this alternative, the distance between neural networks is defined by the similarity or dissimilarity of their pairwise Euclidean matrix E_z. The resulting landscape can be seen in Figure 10.

Alternative 2 - Comparing k-NN Graphs We repeat the same steps as above to obtain E_z and D_z. Then, use the k-Nearest Neighbor algorithm, with $k = 5$, to construct a k-NN similarity matrix W_z. Here, W_z is our characteristic embedding of neural network z.

We repeat this embedding process $\forall z \in [Z]$. Using these embeddings, we compute the pairwise Frobenius norm between $\{(E_i, E_j) : i, j \in [Z]\}$ for each trained neural network z to obtain the $m \times m$ distance matrix N_2. PHATE was then used to visualize N_2.

Alternative 3 - Comparing Weight Matrices Let $z \in [Z]$ be a neural network. Its output after the k^{th} layer is defined by $h_k = f_k(W^{(k)} + b^{(k)})$. We define our distance matrix, D_z, across the manifold, such that $D_{i,j} = \|W_i^{(k)} - W_j^{(k)}\|_F$, where F is the Frobenius norm. The distance matrix is then visualized using PHATE. Here, the distance between neural networks is defined by the similarity or dissimilarity of their weight matrix. The resulting landscape can be seen in Figure 11. Using the evaluation metric from Table 1, the weights-based manifold places the normalized average distance between Top-N highest accuracy models at: Top 10: 0.3721, Top 20: 0.5316, and the Top 30: 0.5504.

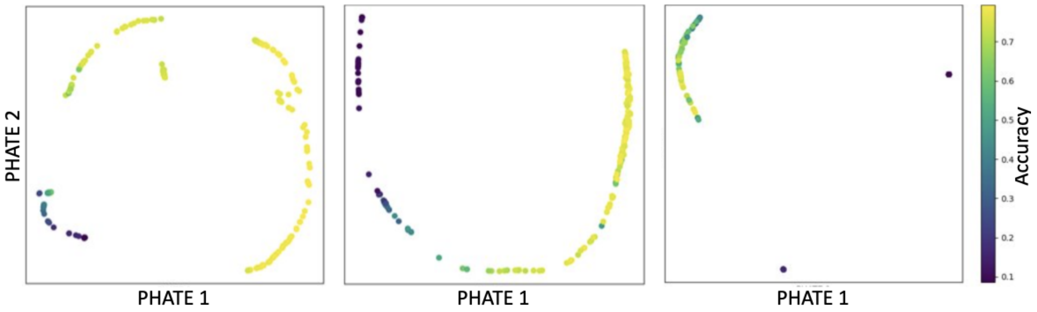


Figure 10: Landcape of neural networks by PHATE using only Euclidean distance. **Left:** MLPs for MNIST classification. **Middle:** ResNets for CIFAR-10 classification. **Right:** CNNs for CIFAR-10 classification.

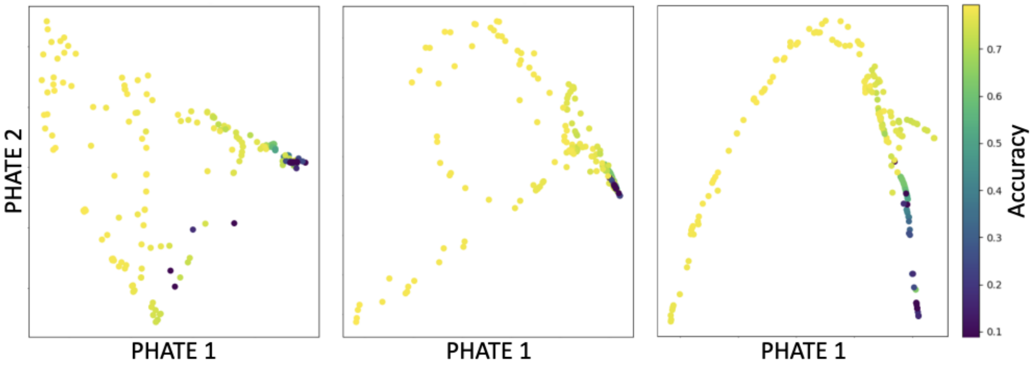


Figure 11: Landscape of neural networks by PHATE using distance between the (left to right) first, second, and third layer weight matrices of the MLPs.

C TDA

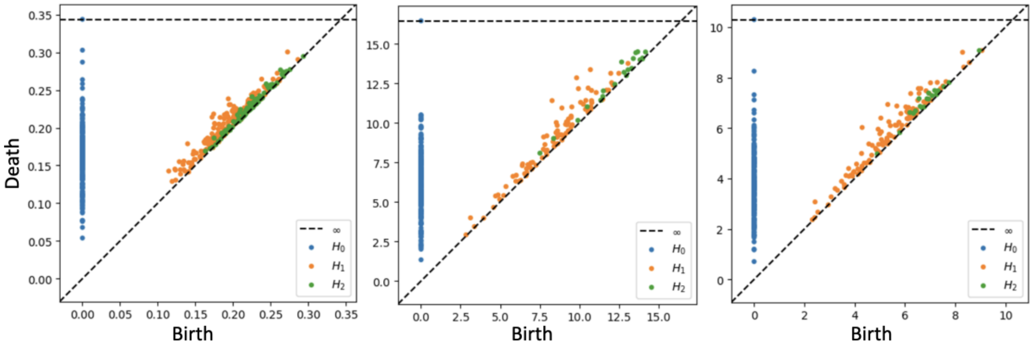


Figure 12: Persistence diagrams representing three trained MLPs with different accuracies (**Left** : 0.1, **Middle** : 0.97, **Right** : 0.97). The H_0 in high-performing networks survive longer than those in the low-performing network, illustrating the improved seperation of clusters.

Figure 12 presents one low-performing and two high-performing MLPs, all trained on their unique set of hyperparameters. Similarities in persistence diagrams are visually clear, despite each network being unique.

D Diffusion Spectral Mutual Information

The Shannon mutual information between two variables measures how much information one captures about the other, or the decrease in the entropy from knowing one variable. The mutual information between two random variables is zero if and only if they are independent. It is defined as $I(X;Y) = H(X) - H(X|Y) = H(X) - \sum_i p(Y = y_i)H(X|Y = y_i)$ Based on the previous definition of DSE, we can now define Diffusion Spectral Mutual Information (DSMI) as $I_D(Z;Y) = S_D(P_z, t) - \sum_{y_i \in Y} p(Y = y_i)S_D(P_{Z|Y=y_i}, t)$ where $P_{Z|Y=y_i}$ is the transition matrix computed on the subset of Z that has class label Y.

The DSMI allows us to compare the distribution of eigenvalues of two diffusion matrices. In this paper, we limit ourselves to this definition and only compute mutual information between the penultimate layer embeddings and the output of the network, or the true labels.

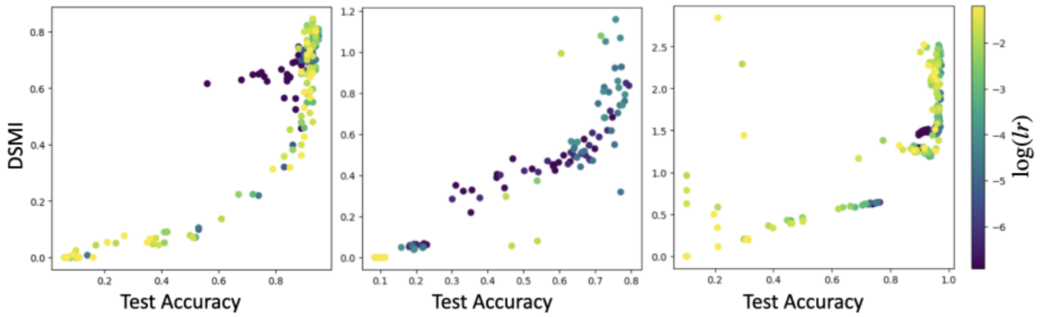


Figure 13: Plots of DSMI between penultimate layer embeddings and true class labels, versus test accuracy. The points are colored by learning rate.

We also find that neural networks with high test accuracy demonstrate higher diffusion mutual information between their penultimate layer embedding and the true class labels (Figure 13), across all three architectures. This is not a surprising result, since we would expect good neural networks to already showcase cluster structure at the penultimate layer. In other words, one would expect the penultimate layer embeddings to be very similar to the output of the network.

Graph Fourier Transform For each architecture, we generate a $n \times n$ affinity matrix W, based on the Frobenius norms between penultimate layer diffusion matrices. We look at the eigenspectrum of the Laplacian $L = WD^{-1}$ of the affinity matrix (where D is the degree matrix). We compute Graph Fourier Transform (GFT) Shuman et al. [Shu+13] for this matrix. We analyze the inner product between the Laplacian eigenspectrum computed from these harmonics and scalar signals on the graph, which we define as signals ∈ {`accuracy`, `learning rate`, `weight decay`, `momentum`}. Essentially, we use the GFT of the neural manifold graph to understand which properties of the neural networks are encoded as low vs. high frequency on the manifold (Figures 14 and 15).

- **Accuracy**: We observe that across different architectures, accuracy emerges as a low-frequency signal. This indicates that the manifold we're constructing effectively organizes neural networks by their performance. The implication is that our approach to building the manifold is appropriate and beneficial for identifying good hyperparameter ranges.
- **Learning Rate**: Learning rate is also a low-frequency signal. This implies there is a region where we can interpolate the learning rate effectively, providing justification for the selected hyperparameter values and highlighting ranges that are worth exploring.
- **Momentum and Weight Decay**: These are high-frequency signals. This indicates more variability and suggests these parameters might require more granular tuning and cannot be easily interpolated.

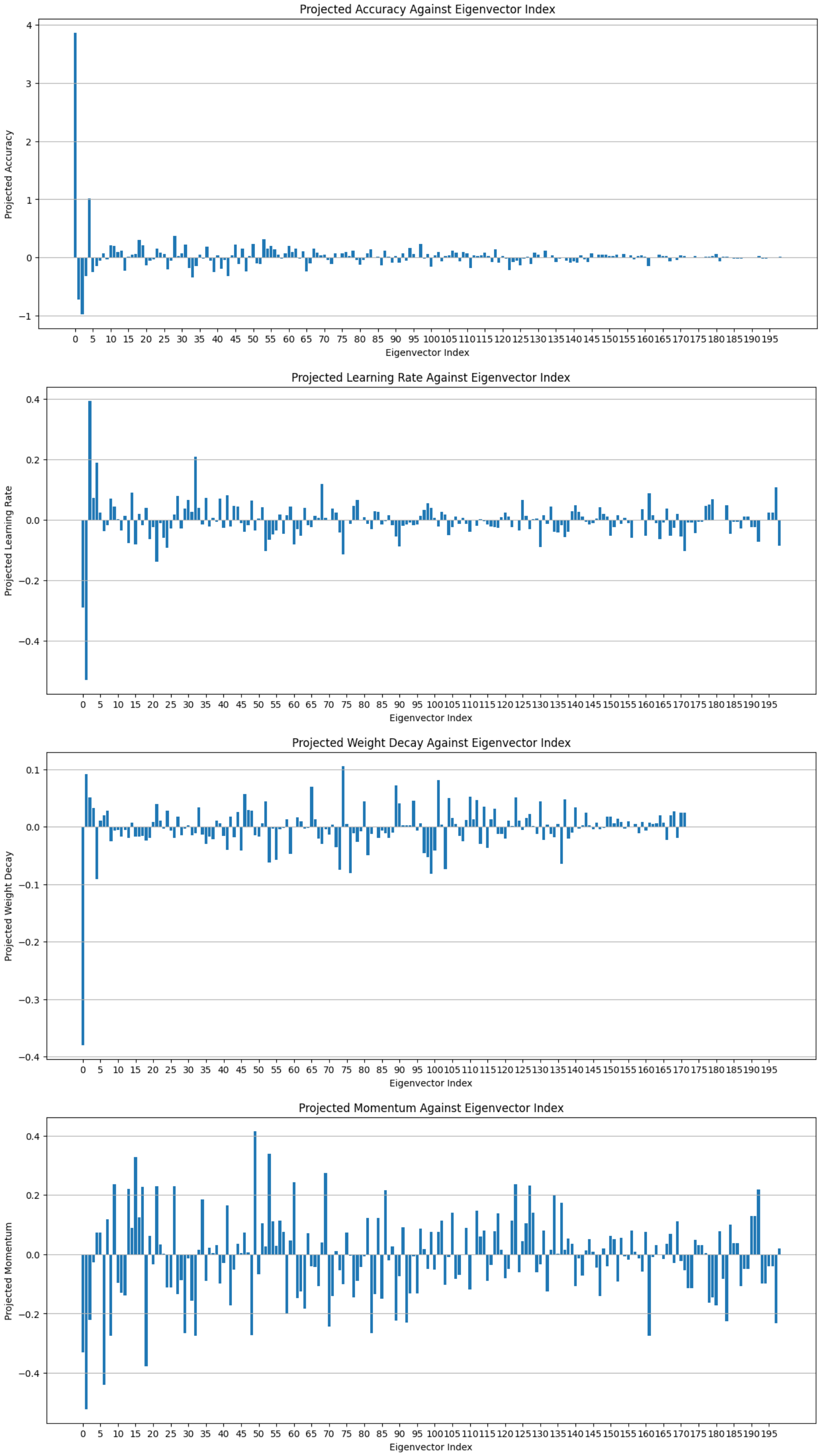


Figure 14: Plots of inner product between signals (top to bottom: **a)** accuracy, **b)** learning rate, **c)** weight decay, **d)** momentum) and the Laplacian's eigenspectrum for the manifold of ResNets. The trivial eigenvector is omitted. We compute the Graph Laplacian quadratic form to provide a measure for smoothness: **a)** 1335.4, **b)** 800.3, **c)** 388.1, **d)** 1525.5.

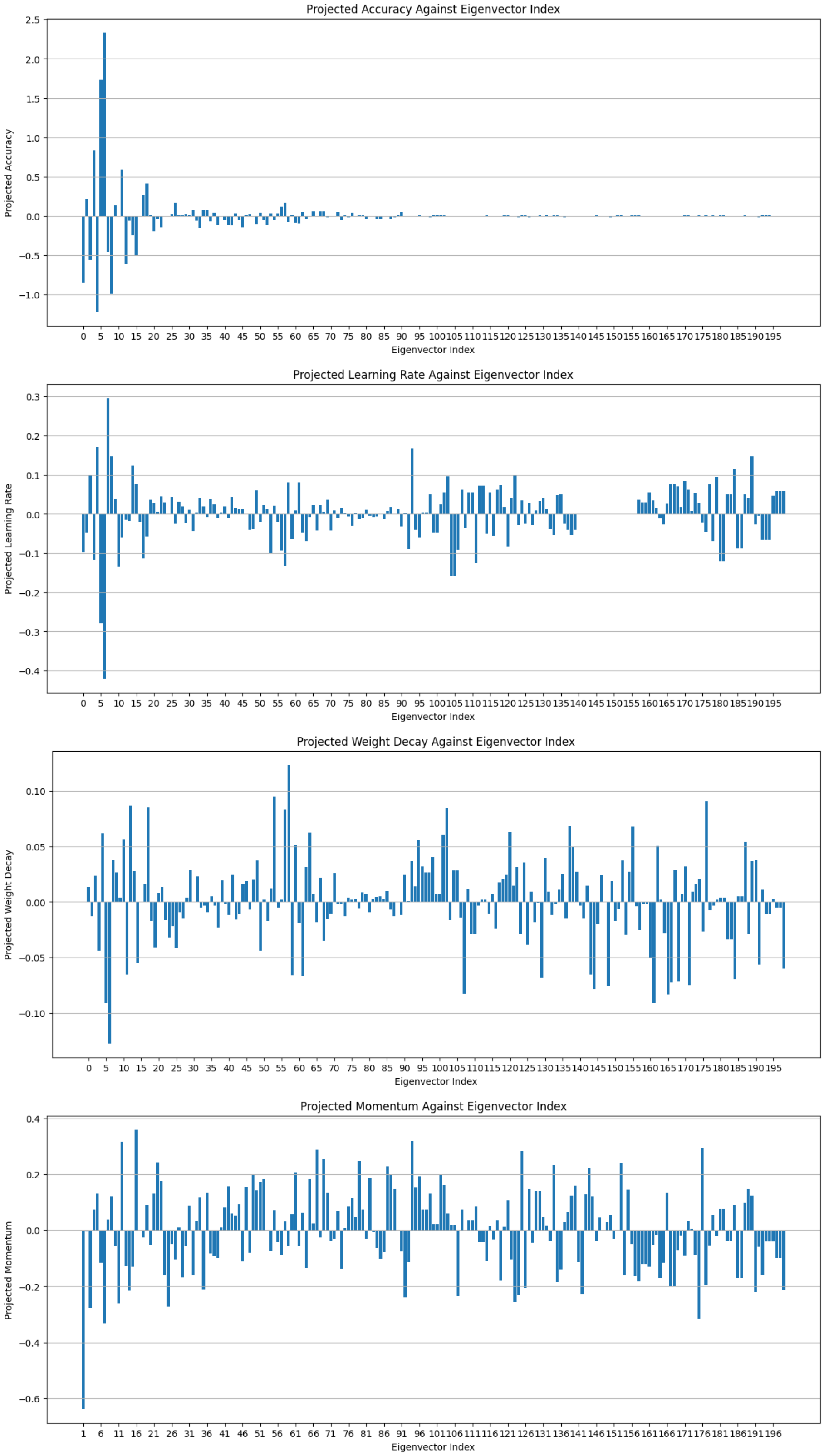


Figure 15: Plots of inner product between signals (top to bottom: accuracy, learning rate, weight decay, momentum) and the Laplacian's eigenspectrum for the manifold of CNNs. We compute the Graph Laplacian quadratic form to provide a measure for smoothness: **a)** 891.8, **b)** 688.5, **c)** 384.1, **d)** 1497.5.

Architecture Search We trained 200 neural networks on MNIST. The architecture consisted of an input layer, two hidden layers, and an output layer. The number of nodes in both hidden layers was randomly chosen between 32 and 512. We extracted each model's internal representation (penultimate embeddings) of the test set and constructed a 200 × 200 pairwise distance matrix based on these embeddings. We visualized this matrix using PHATE (Figure 16).

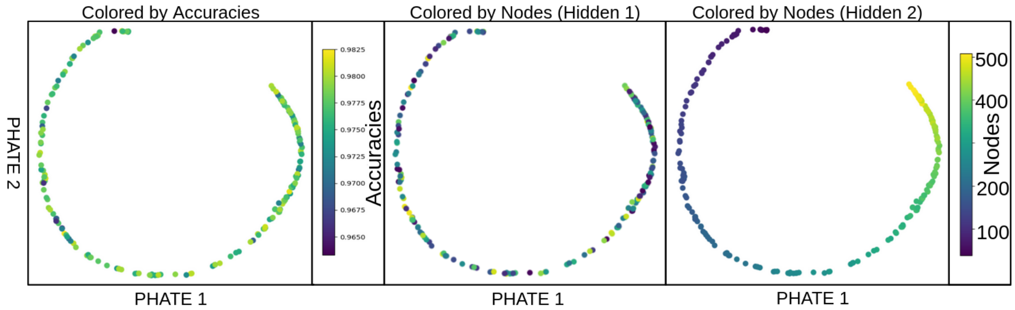


Figure 16: PHATE Plot of Penultimate Embeddings. **From left to right**: Colored by Accuracy, Colored by Number of nodes in the first hidden layer, Colored by Number of nodes in the second hidden layer.

For each internal representation, we computed an individual pairwise distance matrix, yielding 200 square matrices with dimensions equal to the size of the MNIST test set. From these individual distance matrices, we formed a single 200 × 200 pairwise distance matrix by computing the Frobenius norm between every pair of distance matrices. This new pairwise distance matrix was then visualized using PHATE (Figure 17).

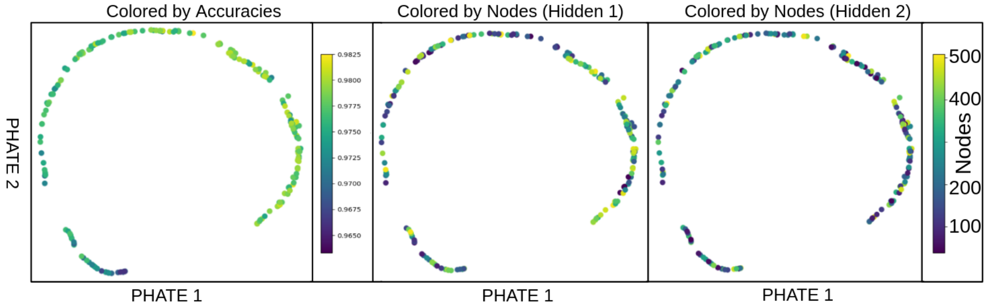


Figure 17: PHATE Plot of Distance Matrices from Penultimate Embeddings, **From left to right**: Colored by Accuracy, Colored by Number of nodes in the first hidden layer, Colored by Number of nodes in the second hidden layer.

These figures indicate that the number of nodes of the second hidden layer is a low-frequency signal on the manifold of neural networks. These findings indicate that there is a region in the manifold on which we can interpolate the second-layer size effectively, shoring the ranges that are worth exploring.