Note: The content of this article reflects its state at the time of acceptance for a workshop at ICML in July 2023. For the latest version with corrections and additional experiments, please visit https://arxiv.org/abs/2312.04823.

Assessing Neural Network Representations During Training Using Data Diffusion Spectra

Danqi Liao [*1] **Chen Liu** [*1] **Alexander Tong** [2] **Guillaume Huguet** [2] **Guy Wolf** [2] **Maximilian Nickel** [3] **Ian Adelstein** [1] **Smita Krishnaswamy** [1 3]

Abstract

Here we present information theoretic measures based on the data diffusion operator as characterisations of the representations learned by neural networks. Specifically, we define diffusion spectral entropy (DSE), i.e., entropy of the diffusion operator computed on the neural representation of a dataset as well as diffusion spectral mutual information (DSMI), which assesses the relationship between different sets of variables representing data. First, we show that these definitions form robust measures of intrinsic dimensionality and relationship strength respectively on toy data, outperforming binned Shannon entropy in terms of accuracy. Then we study the evolution of representations within classification networks and networks with self-supervised losses. In both cases, we see that generalizable training results in decrease in DSE over epochs — starting from a random initialization. We also see that there is an increase in DSMI with the class label over time. On the other hand, training with corrupt labels results in a maintenance or increase in entropy and near-zero DSMI with labels. We also assess DSMI with the input and observe differing trends. On MNIST it grows until plateaus, whereas on CIFAR it increases and then decreases. Overall results show that these measures can elucidate characteristics of network performance as well as data complexity. Code is available at https://github.com/ChenLiu-1996/DiffusionSpectralEntropy.

[*]Equal contribution. Order of co-first authors determined by coin toss. [1]Yale University [2]The Quebec AI Institute and Universite de Montreal [3]The FAIR Team, Meta AI. Correspondence to: Smita Krishnaswamy <smita.krishnaswamy@yale.edu>.

Presented at the 2nd Annual Workshop on Topology, Algebra, and Geometry in Machine Learning (TAG-ML) at the 40th International Conference on Machine Learning, Honolulu, Hawaii, USA. 2023. *Copyright 2023 by the author(s).*

1. Introduction

Deep neural networks have emerged as a major breakthrough in data science largely because of their ability to learn increasingly meaningful representations of data. In fact neural networks function by transforming data via a series of non-linear operations such that each layer learns a new representation of the data. While the representations vectors reside in high dimensional spaces, they in fact lie on a lower dimensional manifold (Fefferman et al., 2016). Assessing the properties of this embedding manifold therefore is key to better understanding the neural network.

Here, we use a powerful manifold learning paradigm — diffusion geometry — to study the representations learned by neural networks. Diffusion geometry involves learning a data diffusion operator, which is a type of Markovian transition matrix describing relationships in the data. A key contribution of this work is in introducing diffusion spectral entropy (DSE), or spectral entropy of the diffusion operator as a robust quantifier of the intrinsic information measure of data representation despite the presence of noise. Further, we extend diffusion spectral entropy to a diffusion spectral mutual information (DSMI) in order to ascertain the information the embedding manifold has on the output labels or the raw input data of the dataset.

Our key contributions are:

- Establishing diffusion geometry as a tool for studying neural network representations.
- Introducing *diffusion spectral entropy*, i.e. Shannon entropy of the spectrum of the diffusion operator as a measure of the information content in a representation of the data.
- Defining *diffusion spectral mutual information* and providing a method of its computation for assessing relationships between different layers of information in a neural network.
- Utilizing both methods to assess the evolution of representations in neural networks over training. In specific we quantify DSE of neural representations as well as DSMI between neural representations and output labels or input data over training on different datasets.

2. Methods

2.1. Diffusion Geometric Quantifications of Manifold Characteristics

Here we define and motivate quantities that we use to characterize neural network representations. More information on the background can be found in supplementary materials A and B. In the supplements, we (1) introduce manifold learning and diffusion geometry, (2) explain the construction of diffusion operator from data graphs, and (3) discuss the definitions of entropy and mutual information.

Diffusion spectral entropy for data While spectral entropy has often been used for measuring entropy on graphs, it has not been used often to compute the entropy of data. Here we make a particular choice to compute a data-centric affinity matrix and then a spectral entropy from that matrix. We utilize the anisotropically normalized diffusion operator from the diffusion maps formulation (Coifman & Lafon, 2006) (see Eqn 6). This operator can be written as a symmetric anisotropic normalization of the Gaussian kernel $G(z_i; z_j)$ by its diagonal degree matrix.

We define the symmetric matrix

$$K_{i;j} = K(z_i; z_j) \tag{1}$$

with K being the anisotropic kernel — an intermediate result during the diffusion maps computation (see Eqn 5). The row stochastic matrix

$$\mathsf{P} = D^{-1} K \tag{2}$$

with $D_{i;i} = \sum_j K_{i;j}$ is our diffusion matrix/operator.

When we compute the diffusion operator on the dataset X, we utilize the additional notation P_X. We define diffusion spectral entropy, with respect to a particular value of diffusion time t as follows.

Definition 2.1. *We define **Diffusion Spectral Entropy (DSE)** as an entropy of the eigenvalues of the diffusion operator P_X computed on a dataset X where $x \in X$ is a multidimensional vector $x_1; x_2 \dots x_d$:*

$$S_D(\mathsf{P}_X; t) := \sum_i \lambda_{i;t} \log(\lambda_{i;t}); \tag{3}$$

where $\lambda_{i;t} := \frac{\lambda_i^t}{\sum_j \lambda_j^t}$, and $\{\lambda_i\}$ are the eigenvalues of the diffusion matrix P_X.

In the matrix P each data point is encoded based on its transition probability to every other data point if one takes a random walk on the data. Thus if a data point is disconnected or far away from others then it is likely that a random walk starting at the data point would remain at the data point. In this setting eigenvectors of the diffusion operator are paths through the data that are stable states of the transition operator. Thus the entropy of the transition operator can be measured over the eigenbasis diffusion operator. Since rows of this matrix can also be thought to be representations of the data (based on their relationships to other points), this is also a measure of intrinsic dimensionality of the dataset. Note that the parameter t that parameterizes the entropy also gives us the capability of separating noise from the true entropy of the signal. As the value of t increases, the eigenspectrum shifts towards the low frequency eigenvectors (which move slowly over the graph) because the eigenvalues $|\lambda_i| < 1$ diminish at a rate inversely proportional to their value when they are raised to a power t. Indeed raising P_K^T results in identical eigenvectors and powered eigenvalues, which achieves a low-pass filtering of the data values over the created affinity graph (Van Dijk et al., 2018), which to tune computation of entropy. We note that a similar measure was used in the supplementary material of (Moon et al., 2019) to select parameters, but diffusion spectral entropy was not formally defined or discussed there.

Some properties of S_D are discussed and proved in supplementary materials C.

Diffusion spectral mutual information We further extend the diffusion spectral entropy to define mutual information for understanding the information that some variables of a data representation have on others, for example the information that neurons in a hidden layer have about the primary output.

Definition 2.2. *We define the **Diffusion Spectral Mutual Information (DSMI)** as the difference between conditional and unconditional diffusion spectral entropy*

$$I_D(X; Y) = S_D(\mathsf{P}_X; t) - \sum_{y_i \in Y} p(Y = y_i) S_D(\mathsf{P}_{X|Y=y_i}; t). \tag{4}$$

The conditioned transition matrix $\mathsf{P}_{X|Y=y_i}$ is the transition matrix computed on the subset of X that has output label Y. To avoid numeric issues that are involved in comparing spectra of different sizes of matrices, we also compute $S_D(\mathsf{P}_X; t)$ using subsamples of X the average size of the classes of Y. Since uniform subsampling maintains distributions, the sampled entropy would be the same as the total entropy as shown in our experiments (see supplementary materials G).

In addition, empirically we show that DSMI can reflect the relationships between data points and their class labels, and that these degrade with corruption.

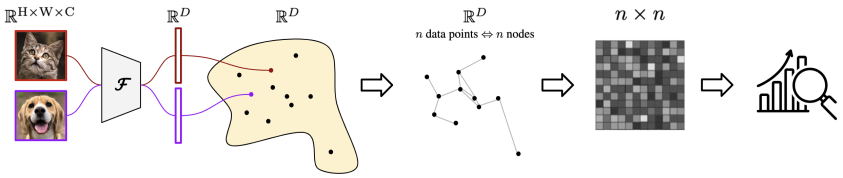

Figure 1. **Data processing to obtain the diffusion matrix of an embedding manifold.** Each data point x_i (in this case, an image) from the validation set is embedded by the network F as a vector z on a D-dimensional embedding manifold (light yellow canvas). This yields a point cloud with n points, which can be converted into a graph based on local proximity. We can compute its diffusion matrix which allows for further analysis.

2.2. Efficiently Computing Aforementioned Quantities

Details can be found in supplementary materials E.

3. Results

In this study, we used the ResNet-50 (He et al., 2016) backbone, and assessed the penultimate layer of the network. As a convolutional network, its penultimate layer is different from most preceding layers as the representations learned do not contain explicit spatial dimensions. Specifically, the representations lie in $\mathbb{R}^D$ rather than $\mathbb{R}^{h\ w\ c}$. This property allows us to interpret these vectors as points in a fixed-dimensional space, whereas for other layers we need to flatten the representations for analysis, which may raise concerns on mixing spatial information with channel information. Our evaluation framework can be easily adapted to many other vision backbones.

We trained the vision backbones under three conditions: (1) supervised learning, (2) contrastive learning (specifically, SimCLR (Chen et al., 2020)), and (3) purposeful overfitting. More details can be found in supplementary materials D.

To analyze the embedding manifold, we pass the entire validation set through the network and collect the embedding vectors. The key step is to convert this point cloud of n embedding vectors into a data graph, from which we can find the corresponding diffusion matrix. DSE and DSMI can be computed from eigenvalues of this diffusion matrix. This process is illustrated in Figure 1.

3.1. Toy Test Cases for DSE and DSMI

Results can be found in supplementary materials F.

3.2. Results on Neural Network Training Process

There are several key observations on the diffusion geometric quantities on the embedding manifold during neural network training. **Every figure in this section has an enlarged version in supplementary materials H.**

Figure 2. **Diffusion Spectral Entropy $S_D(Z)$ of embedding vectors Z.** Colors correspond to the three random seeds. t is empirically set to 1 for MNIST and 2 for CIFAR10. The same t setting is used in all subsequent figures.

Figure 3. **Diffusion Spectral Mutual Information $I_D(Z;Y)$ between embedding vectors Z and the label classes Y.**

DSE We can observe from Figure 2 that DSE in proper learning (i.e., supervised or contrastive learning on correct labels) decreases as the model performs better on the downstream classification task. Meanwhile, this trend is completely absent when the model is forced to memorize random nonsense labels. This accords with the intuition that random initialization has high variance and entropy, while class labels have much lower entropy, and even organizations created by contrastive losses have lower "surprise" than random sampling.

DSMI with output In Figure 3, it can be observed that DSMI $I_D(Z;Y)$ consistently increases during proper learning. Under the same learning rate and scheduling, the DSMI climbs more slowly in contrastive learning compared to supervised learning, and it ends up at a lower terminal value. This may be attributed to the fact that contrastive learning lacks the direct supervision from explicit class labels. However since class labels relate to the data geometry, self-

supervised learning on the data alone still yields some mutual information with labels. In nonsense memorization, DSMI quickly converges to zero. This aligns well with the expectation, since a classifier that essentially performs random guessing has zero mutual information with the class label, whereas a functioning classifier corresponds to a positive mutual information.

Taken together the DSE and DSMI trends indicate that the representations coalesce to a less noisy and more streamlined form where they mainly contain information about the the output label during good training. Results on the classic Shannon version can be found in supplementary materials I.

DSMI with input We additionally show DSMI with the input signal in Figure 4. In nonsense memorization, $I_D(Z;X)$ stays close to zero just like $I_D(Z;Y)$, which ascertains the random projection tendencies. During proper learning, the information bottleneck theory would suggest $I_D(Z;X)$ shall decrease (Tishby & Zaslavsky, 2015) while counterarguments have also been provided in (Saxe et al., 2019). Our results suggest they may both be correct, and the trend may depend on the nature of the dataset X. $I_D(Z;X)$ keeps increasing during learning on the MNIST dataset, whereas it begins to decrease after some point on the CIFAR10 dataset. This may be verified by future studies on more datasets.

Figure 4. **Diffusion Spectral Mutual Information $I_D(Z;X)$ between embedding vectors Z and the input X.** Input X is flattened and spectral-clustered into same number of categories as the output Y for fair comparison.

Embedding visualization See supplementary materials J for how the visualized embeddings corroborate with the DSE trends assessed during training.

4. Conclusion

In conclusion, we introduced diffusion operator-based information theoretic measures as tools for assessing neural network representations. Specifically we proposed diffusion spectral entropy (DSE) for measuring information in embedding manifold. We further defined diffusion spectral mutual information (DSMI) and provided an efficient method of its computation, and proved bounds on its values in extreme as well as idealistic clustered cases. Through extensive simulation on toy datasets, we demonstrated diffusion spectral entropy is a measure of intrinsic dimension on toy data, and spectral mutual information is a meaningful measure for mutual dependence between two variables. We also investigated the neural representation from the penultimate layer of ResNet-50 under supervised learning, contrastive learning and overfitting settings. We empirically showed that DSE during learning follows a general decreasing trend while such decreasing trend is not observed in overfitting setting. Further, we showed DSMI between a hidden layer and output increases during training and plateaus at some point whereas it quickly converges to zero in overfitting settings. We saw more complex data-dependent trends on DSMI with primary inputs. On the MNIST dataset DSMI with input increases until plateaus while on CIFAR it has non-monotonic trends characterized increase and later decrease.

See further discussions in supplementary materials K.

References

Chen, T., Kornblith, S., Norouzi, M., and Hinton, G. A simple framework for contrastive learning of visual representations. In *International conference on machine learning*, pp. 1597–1607. PMLR, 2020.

Coifman, R. R. and Lafon, S. Diffusion maps. *Applied and computational harmonic analysis*, 21(1):5–30, 2006.

Fefferman, C., Mitter, S., and Narayanan, H. Testing the manifold hypothesis. *Journal of the American Mathematical Society*, 29(4):983–1049, 2016.

He, K., Zhang, X., Ren, S., and Sun, J. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 770–778, 2016.

Moon, K. R., van Dijk, D., Wang, Z., Gigante, S., Burkhardt, D. B., Chen, W. S., Yim, K., Elzen, A. v. d., Hirn, M. J., Coifman, R. R., et al. Visualizing structure and transitions in high-dimensional biological data. *Nature biotechnology*, 37(12):1482–1492, 2019.

Saxe, A. M., Bansal, Y., Dapello, J., Advani, M., Kolchinsky, A., Tracey, B. D., and Cox, D. D. On the information bottleneck theory of deep learning. *Journal of Statistical Mechanics: Theory and Experiment*, 2019(12):124020, 2019.

Tishby, N. and Zaslavsky, N. Deep learning and the information bottleneck principle. In *2015 ieee information theory workshop (itw)*, pp. 1–5. IEEE, 2015.

Van Dijk, D., Sharma, R., Nainys, J., Yim, K., Kathail, P., Carr, A. J., Burdziak, C., Moon, K. R., Chaffer, C. L., Pattabiraman, D., et al. Recovering gene interactions from single-cell data using data diffusion. *Cell*, 174(3): 716–729, 2018.

Supplementary Materials

A. Background

Manifold learning and diffusion geometry A useful assumption in representation learning is that high dimensional data, which is commonly used in deep learning, originates from an intrinsic low dimensional manifold that is mapped via nonlinear functions to observable high dimensional measurements. This is commonly known as the manifold assumption (Fefferman et al., 2016). Let $\mathcal{M}^d$ be a hidden d dimensional manifold that is only observable via a collection of n d nonlinear functions $f_1, \ldots, f_n : \mathcal{M}^d \to \mathbb{R}$ that enable its immersion in a high dimensional ambient space $\mathcal{E}(\mathcal{M}^d) = \{f(z) = (f_1(z), \ldots, f_n(z))^T : z \in \mathcal{M}^d\} \subset \mathbb{R}^n$ from which data is collected. Conversely, given a dataset $\mathcal{X} = \{x_1, \ldots, x_N\} \subset \mathbb{R}^n$ of high dimensional observations, manifold learning methods assume data points originate from a sampling $\mathcal{Z} = \{z_i\}_{i=1}^N \subset \mathcal{M}^d$ of the underlying manifold via $x_i = f(z_i)$, $i = 1, \ldots, n$, and aim to learn a low dimensional intrinsic representation that approximates the manifold geometry of $\mathcal{M}^d$.

A paradigm that has emerged as useful in manifold learning in recent years is diffusion geometry (Coifman & Lafon, 2006; Moon et al., 2019; Van Dijk et al., 2018; Burkhardt et al., 2019; Huguet et al., 2022). Diffusion geometry seeks to describe data points based on random-walk probabilities to one another. This has been seen to be a noise-tolerant and adaptive way of representing data whose dimensionality reductions have yielded methods such as PHATE (Moon et al., 2019) and diffusion maps (Coifman & Lafon, 2006).

Diffusion maps begin with a kernel K, often a Gaussian kernel $\exp(-\|z_1 - z_2\|^2/\epsilon)$, where $\epsilon > 0$ is interpreted as a user-con gurable neighborhood size. Such a kernel transforms distances between data points to similarities or af nities. However, such neighborhoods encode sampling density information together with local geometric information. To construct a diffusion geometry that is robust to sampling density variations we may use an anisotropic kernel

$$K(z_1, z_2) = \frac{G(z_1, z_2)}{\|G(z_1, \cdot)\|_1 \|G(z_2, \cdot)\|_1} \qquad G(z_1, z_2) = e^{-\frac{\|z_1 - z_2\|^2}{\epsilon}} \tag{5}$$

as proposed in (Coifman & Lafon, 2006), where $0 \leq \alpha \leq 1$ controls the separation of geometry from density, with $\alpha = 0$ yielding the classic Gaussian kernel, and $\alpha = 1$ completely removing density and providing a geometric equivalent to uniform sampling of the underlying manifold. Next, the similarities encoded by K are normalized to de ne transition probabilities $p(z_1, z_2) = \frac{K(z_1, z_2)}{\|K(z_1, \cdot)\|_1}$ that are organized in an $n \times n$ row stochastic matrix

$$P_{i,j} = p(z_i, z_j) \tag{6}$$

that describes a Markovian diffusion process over the intrinsic geometry of the data. Finally, a diffusion map (Coifman & Lafon, 2006) is de ned by taking the eigenvalues $1 = |\lambda_1| \geq |\lambda_2| \geq \cdots \geq |\lambda_N|$ and corresponding eigenvectors $\{\psi_j\}_{j=1}^N$ of P, and mapping each data point $x_i \in \mathcal{X}$ to an N dimensional vector $\Psi_t(x_i) = [\lambda_1^t \psi_1(x_i), \ldots, \lambda_N^t \psi_N(x_i)]^T$, where t represents a diffusion-time, i.e., number of transitions considered in the diffusion process. In general, as t increases, most of the eigenvalues λ_j^t, $j = 1, \ldots, N$, become negligible, and thus truncated diffusion map coordinates can be used for dimensionality reduction (Coifman & Lafon, 2006). For example, PHATE involves computing a symmetric divergence between the rows of the diffusion operator and embedding this with multidimensional scaling.

Entropy and mutual information Entropy, a basic quantity in information theory, quanti es the amount of uncertainty or "surprise" when given the value of a random variable. If the variable is distributed with a distribution that has probability mass that is spread out, such as a uniform distribution, then the entropy is high. On the other extreme, if there is no uncertainty in the quantity of the variable, i.e., it is deterministic then the entropy is 0. The Shannon entropy is computed as below.

$$H(X) = E[-\log p(X)] = -\sum_{x \in \mathcal{X}} p(x) \log p(x) \tag{7}$$

The von Neumann entropy (von Neumann, 2018) from quantum information extends the entropy measure to the quantum mechanics domain, and in particular it operates on density matrices. If a density matrix has a set of eigenvalues $\{_i\}$, the von Neumann entropy is de ned as

$$H( ) = \quad tr(\ \ln \ ) = \sum_i \ _i \log \ _i \tag{8}$$

Here von Neumann entropy is considered to be an extension of Gibbs entropy, which is a measure of the spread of a distribution on the microstates of a classical system. Classical systems can only exist in pure states (or standard basis states). However, quantum systems can exist in superposition states, and depending on the distribution of superposition states the stable or ground states can be rede ned as the eigenfunctions of a density operator which describes the probabilities of superpositions.

This notion has subsequently been extended to graph spectra in several works. These methods generally compute the entropy of normalized eigenvalue of a graph adjacency matrix and have been used variously in biology and other elds to compare graphs (Su et al., 2022; de Siqueira Santos et al., 2016; Takahashi et al., 2012; Merbis & de Domenico, 2023; Villafañe-Delgado & Aviyente, 2016).

Mutual information is de ned as a function of entropy. There are many alternative formulations of mutual information that are equivalent. The most useful formulation here is as a difference between the (unconditional) entropy of a variable, and the entropy of a variable when conditioned on the value of another variable.

$$I(X;Y) = H(X) \quad H(X|Y) = H(X) \quad \sum_i p(Y = y_i)H(X|Y = y_i) \tag{9}$$

Here the conditional entropy $H(X|Y)$ is given as a weighted sum over values y_i of Y in the discrete case, and thus it is computed as in the form on the right hand of Eqn 9.

B. Related Works

Prior works attempted to study neural networks during training by visualizing the neural representation (Gigante et al., 2019) or the loss landscape (Li et al., 2018). These works provided some qualitative ways to analyze neural networks during training but did not offer quanti cation. The information bottleneck theory for deep learning (Tishby & Zaslavsky, 2015) introduced a framework for quantifying information content in neural networks during training. They binned the vectors along each feature dimension to form a probability distribution and computed the Shannon entropy and mutual information. The main limitation of their proposed method is the curse of dimensionality in the binning process that renders it impractical to analyze layers with more than a dozen neurons — which is ubiquitous in modern deep neural networks (see supplementary material I). Follow-up work (Saxe et al., 2019) used kernel density estimation (Kolchinsky & Tracey, 2017) and Kraskov estimator (Kraskov et al., 2004) for approximating mutual information, yet both estimation methods require speci c assumptions on the distributions of hidden layer activation. Our proposed method does not assume speci c distribution on hidden layer activations and also avoids the binning problem since it operates on the eigenspectrum rather than the set of embedding vectors.

C. Propositions on Diffusion Spectral Entropy and Diffusion Spectral Mutual Information

The propositions below establish some bounds on minimal and maximal values of DSE. In addition, they provide intuition on the de nition of DSMI. Note that taking $t \to 1$ allows us to talk about the major structures in the dataset.

Proposition C.1. S_D achieves minimal entropy of 0 when the diffusion operator de nes an ergodic Markov chain, and is in steady state (as $t \to 1$).

Proof. The eigenvalues of a nite ergodic Markov chain have the form $1 = |_1| > |_2| \quad |_N| \quad 0$, we see that $_i^t = 0$ as $t \to 1 \ \forall \ i > 1$. Thus the resultant entropy $1 \log(1) + \sum_{i>1} 0 = 0$, proving the proposition. □

Note that this also implies that if all data points are very similar, i.e., have equal probability of transitioning to any other point, then it has minimal entropy. This is because such a distribution is a steady state distribution for the diffusion matrix shown in Eqn 2.

Note that steady state distribution for ergodic P (which is the case of our Kernel-based de nition) is characterized by having the same transition probabilities from every starting state. Therefore the rows of the matrix P^t are identical, and there is no distinguishability between points.

Next we discuss when this entropy will reach its maximal value. This happens effectively when points are all spread very far apart.

Additionally, we can show that there exists a $T \in \mathbb{N}$, such that the diffusion spectral entropy $S_D(P_X; t)$ decreases for all $t > T$, implying that as the data is denoised the entropy decreases.

Proposition C.2. Assuming that P_X has at least one eigenvalue $\lambda_j \in (0; 1)$, then, there exist a $T \in \mathbb{N}$ such that for all $t > T$

$$\frac{\partial}{\partial t} S_D(P_X; t) < 0.$$

Proof. We recall that the eigenvalues of P_X can be ordered as $1 = |\lambda_0| \geq |\lambda_1| \geq \cdots \geq |\lambda_n| \geq 0$. Therefore $\sum_j |\lambda_j|^t \geq 1$. By computation we have:

$$\begin{aligned}
\frac{\partial}{\partial t} S_D(P_X; t) &= \frac{\partial}{\partial t}\left[\sum \frac{|\lambda_j|^t}{\sum_j |\lambda_j|^t} \log \frac{|\lambda_j|^t}{\sum_j |\lambda_j|^t}\right] \\
&= \sum_i \frac{|\lambda_i|^t}{\left(\sum_j |\lambda_j|^t\right)^2} \left(\sum_j |\lambda_j|^t \left[\log|\lambda_j| - \log|\lambda_i|\right]\right) \left(\log \sum_j |\lambda_j|^t - t \log|\lambda_i| - 1\right)
\end{aligned}$$

The rst term is always positive. The asymptotic behavior of the second term is positive. This can be seen by noting that taking the limit of $t \to \infty$, the only summands that do not converge to zero are when $|\lambda_i| = 1$. Since these are positive, the sum in the second term is positive in the limit of $t \to \infty$, and furthermore, since it is continuous, it remains positive for suf ciently big t.

Examining the third term we have that for suf ciently large t the $-t \log|\lambda_i|$ term dominates, giving a positive third term, and has positive derivative.

since all three terms are positive for suf ciently large t, $\frac{\partial}{\partial t} S_D(P_X; t) < 0$ completing the proof. □

If instead of using a Gaussian kernel for computation of P, we use a k-nearest-neighbor or other thresholded kernel we can make the following statement.

Proposition C.3. As $t \to \infty$, $S_D(P_X; t)$ on data with k well-separated clusters $\to \log k$.

Proof. If the data has k well separated clusters then P_X has eigenvalues of the form $1 = |\lambda_1| = |\lambda_2| = \cdots = |\lambda_k| > |\lambda_{k+1}| \geq \cdots \geq |\lambda_N| \geq 0$. In other words the multiplicity of 1 eigenvalues of P_X corresponds to the number of connected components in the underlying graph, which here is k, all other eigenvalues are strictly less than 1 and greater than or equal to 0. Therefore as $t \to \infty$ only these eigenvalues remain and the resultant DSE is $\frac{1}{k} \log k = \log k$ completing the proof. □

Corollary 1. S_D achieves maximal entropy in a matrix where each point only transitions to itself, and the entropy here will be $\log(n)$

Proof. In this case the transition matrix corresponds to the identity matrix, hence, each of the eigenvalues is 1. Thus the diffusion spectral entropy is the uniform distribution on n states $\sum_i^n (1/n) \log(1/n) = \log(n)$, which maximizes the entropy. □

Proposition C.4. As $t \to \infty$ on a hidden layer X with k well-separated clusters, and output labels perfectly coinciding with clusters, we will have $I_D(X; Y) = \log k$, i.e., DSMI between hidden layer and output layer being positive.

Proof. Based on proposition 3.3, the entropy of k well-separated clustered data with $t = 1$ is $\log k$, thus $S_D(P_X; t) = \log(k)$. However for each label y $S_D(P_X; t) = \log(1) = 0$. Thus $I_D(X; Y) = \log k$. □

D. Experimental Details

D.1. Three Training Conditions on Real Data

Supervised Learning We trained ResNet-50 (He et al., 2016) models end-to-end using an AdamW optimizer (Loshchilov & Hutter, 2017) at an initial learning rate 1e-5 for MNIST or 1e-3 for CIFAR10. Learning rate is modulated by a Cosine Annealing Scheduler (Loshchilov & Hutter, 2016) with a linear warmup for the first 10 epochs. Early stopping kicks in if the validation accuracy no longer increases for 15 epochs. Experiments are repeated over 3 random seeds. At the end of each epoch, we pass the entire validation set (10,000 images for MNIST or CIFAR10) through the model and collect the $\mathbb{R}^{2048}$ representation vectors as the outputs of the penultimate layer for further analysis.

Contrastive Learning For the contrastive learning experiments, we followed the SimCLR (Chen et al., 2020) paradigm: we create two augmentations of the same image and asks the model to embed them closer on the embedding manifold, while encouraging bigger separation between them and the other images in the same batch. The standard training procedure of contrastive learning first trains the backbone, with the classifier (the final, linear layer) detached, for some epochs and then either performs a linear probing or fine-tuning. In either case, a linear classifier layer is attached and trained for some more epochs. The weights of the backbone are frozen in the former case versus learnable in the latter.

Since we need to assess how well the model is learning at each epoch, we instead performs linear probing by the end of each epoch. Specifically, we freeze the backbone weights, attach a re-initialized linear classifier layer, and train the classifier for 10 epochs with the training set. Then, we record the end-to-end validation accuracy as well as collect the embedding vectors on the validation set, similar to the supervised learning case. Finally, we unfreeze the backbone weights for the next epoch of SimCLR training.

For fair comparison, the training details are otherwise the same as the supervised learning case, including learning rate and scheduling.

Purposeful Overfitting In purposeful overfitting, we train the model in the same manner as the supervised learning case, except that the data labels are randomly permuted. In this way, the models are forced to learn nonsense labels. To better overfit, we reduced the extent of data augmentation during training, since data augmentation is proven effective to mitigate overfitting. We also increased the early-stop patience from 15 to 30, with the triggering metric being the train-validation accuracy divergence instead of validation accuracy.

For fair comparison, the training details are otherwise the same as the supervised learning case, including learning rate and scheduling.

D.2. Computing DSMI with input

We compute $I_D(Z; X)$ in the same fashion as we compute $I_D(Z; Y)$. By a simple change of variables, Eqn 4 can be rewritten as:

$$\begin{aligned} I_D(Z; X) &= S_D(Z) - S_D(Z|X) \\ &= S_D(P_Z; t) - \sum_{x_i \in X} p(X = x_i) S_D(P_Z|X = x_i; t) \end{aligned}$$

Compared to DSMI between the neural representation and the output, the DSMI with the input is slightly more complicated because the input signals do not fall in discrete categories x_i i.e. are not naturally defined. To that end, for the set of input images, we flatten them respectively and perform spectral clustering. For fair comparison with $I_D(Z; Y)$, we cluster these vectors into the same number of categories as the number of classes in Y. The remaining process is the same as how we compute DSMI with the output.