Supplementary Materials – On convex decision regions in deep network representations

1 Theory

In this appendix, we present some mathematical background and intuition for the approximate convexity workflow.

1.1 Euclidean convexity is invariant to affine transformations

Theorem 1. Let $f : \mathbb{R}^n \to \mathbb{R}^n$ be an affine transformation (i.e., $f(\mathbf{x}) = \boldsymbol{A}\mathbf{x} + \mathbf{b}$, where $\mathbf{b} \in \mathbb{R}^n$ and $\boldsymbol{A} : \mathbb{R}^n \to \mathbb{R}^n$ is an invertible linear transformation). Let $X \subset \mathbb{R}^n$ be a convex set. Then $f(X)$ is convex [1].

Proof. Let $\mathbf{x}, \mathbf{y} \in f(X)$ and $\lambda \in [0, 1]$. There exist $\overline{\mathbf{x}}, \overline{\mathbf{y}} \in X$ such that $f(\overline{\mathbf{x}}) = \mathbf{x}$ and $f(\overline{\mathbf{y}}) = \mathbf{y}$. Since X is convex, we have $\lambda\overline{\mathbf{x}} + (1 - \lambda)\overline{\mathbf{y}} \in X$. Therefore,

$$f\left(\lambda\overline{\mathbf{x}} + (1 - \lambda)\overline{\mathbf{y}}\right) \in f(X). \tag{1}$$

Moreover, because of the linearity of A,

$$f\left(\lambda\overline{\mathbf{x}} + (1 - \lambda)\overline{\mathbf{y}}\right) = \boldsymbol{A}\left(\lambda\overline{\mathbf{x}} + (1 - \lambda)\overline{\mathbf{y}}\right) + \mathbf{b} = \lambda\boldsymbol{A}\overline{\mathbf{x}} + (1 - \lambda)\boldsymbol{A}\overline{\mathbf{y}} + \mathbf{b}. \tag{2}$$

Finally, we have

$$\begin{aligned} \lambda\boldsymbol{A}\overline{\mathbf{x}} + (1 - \lambda)\boldsymbol{A}\overline{\mathbf{y}} + \mathbf{b} &= \lambda\left(\boldsymbol{A}\overline{\mathbf{x}} + \mathbf{b}\right) + (1 - \lambda)\left(\boldsymbol{A}\overline{\mathbf{y}} + \mathbf{b}\right) && (3)\\ &= \lambda f(\overline{\mathbf{x}}) + (1 - \lambda)f(\overline{\mathbf{y}}) && (4)\\ &= \lambda\mathbf{x} + (1 - \lambda)\mathbf{y}. && (5) \end{aligned}$$

Hence,

$$f\left(\lambda\overline{\mathbf{x}} + (1 - \lambda)\overline{\mathbf{y}}\right) = \lambda\mathbf{x} + (1 - \lambda)\mathbf{y} \tag{6}$$

and $\lambda\mathbf{x} + (1 - \lambda)\mathbf{y} \in f(X)$. Therefore, $f(X)$ is convex. □

1.2 Graph convexity is invariant to isometry and uniform scaling

Theorem 2. Graph convexity is invariant to isometry and uniform scaling.

Proof. Isometry (with respect to Euclidean distance) is a map $I : \mathbb{R}^n \to \mathbb{R}^n$ such that

$$\forall \mathbf{x}, \mathbf{y} \in \mathbb{R}^n : \|\mathbf{x} - \mathbf{y}\|_2 = \|I(\mathbf{x}) - I(\mathbf{y})\|_2. \tag{7}$$

Since isometry preserves distances, it induces the same distance graph. Therefore, graph convexity is invariant to isometries.

Universal scaling is a map $S : \mathbb{R}^n \to \mathbb{R}^n$ such that

$$\exists C \in \mathbb{R} \;\; \forall \mathbf{x} \in \mathbb{R}^n : \;\; S(\mathbf{x}) = C\mathbf{x}. \tag{8}$$

It holds that

$$\|S(\mathbf{x}) - S(\mathbf{y})\|_2 = \|C\mathbf{x} - C\mathbf{y}\|_2 = |C| \cdot \|\mathbf{x} - \mathbf{y}\|_2. \tag{9}$$

All the distances are scaled by $|C|$. It follows that the chosen nearest neighbours are the same. We show by contradiction that all the shortest paths are also the same. We denote G the graph with the original edges and G_S the graph with the scaled edges.

Given $\mathbf{x}, \mathbf{y} \in \mathbb{R}^n$, suppose that there exists a shortest path:

$$P = (\mathbf{p_0} = \mathbf{x}, \mathbf{p_1}, \ldots, \mathbf{p_j} = \mathbf{y}) \tag{10}$$

from $\mathbf{x}$ to $\mathbf{y}$, and a different path:

$$Q = (\mathbf{q_0} = S(\mathbf{x}), \mathbf{q_1}, \ldots, \mathbf{q_k} = S(\mathbf{y})) \tag{11}$$

from $S(\mathbf{x})$ to $S(\mathbf{y})$ such that

$$L(Q) < L(S(P)), \tag{12}$$

where

$$S(P) = (S(\mathbf{p_0}), S(\mathbf{p_1}), \ldots, S(\mathbf{p_j})) \tag{13}$$

and $L(\cdot)$ is an operator measuring the length of a path. From equation 12, it follows that $C \neq 0$. Because all the edges are scaled by a factor $|C|$, it holds that $L(S(P)) = |C|L(P)$. Moreover, for every pair of nodes $\mathbf{v_1}$, $\mathbf{v_2}$, there exists an edge in G from $\mathbf{v_1}$ to $\mathbf{v_2}$ if and only if there exists an edge in G_S from $S(\mathbf{v_1})$ to $S(\mathbf{v_2})$. Hence, there exists a path R from $\mathbf{x}$ to $\mathbf{y}$ such that $Q = S(R)$. It follows that

$$|C|L(R) = L(S(R)) < L(S(P)) = |C|L(P). \tag{14}$$

Therefore, $L(R) < L(P)$ and that contradicts the assumption that P is the shortest path from $\mathbf{x}$ to $\mathbf{y}$. □

1.3 Softmax induces convexity

Theorem 3. The preimage of each decision region under the last dense layer and softmax function is a convex set. More precisely: Let us denote the output of the last dense layer by

$$a_k(\mathbf{z}) = \sum_{j=1}^{J} w_{k,j} z_j \tag{15}$$

for $\mathbf{z} \in \mathbb{R}^n$ and the probabilities

$$p_k(\mathbf{z}) = \frac{\exp a_k(\mathbf{z})}{\sum_{k'=1}^{K} \exp a_{k'}(\mathbf{z})}. \tag{16}$$

Denote

$$C_k \subseteq \mathbb{R}^n = \{\mathbf{x} \in \mathbb{R}^n : p_k(\mathbf{x}) > p_j(\mathbf{x}) \; \forall j \in \{1, \ldots, K\}, j \neq k\}. \tag{17}$$

Then C_k is a convex set for any $k \in \{1, \ldots, K\}$ [2].

Proof. Let us define regions

$$\mathbf{x} \in \mathcal{R}_k \iff (a_k(\mathbf{x}) > a_j(\mathbf{x}) \;\; \forall j \neq k). \tag{18}$$

Because $\sum_{k'=1}^{K} \exp a_{k'} > 0$ and by the monotonicity of the exponential function, it holds that

$$k_{opt}(\mathbf{x}) := \arg\max_k p_k(\mathbf{x}) = \arg\max_k \frac{\exp a_k(\mathbf{x})}{\sum_{k'=1}^{K} \exp a_{k'}(\mathbf{x})} = \arg\max_k a_k(\mathbf{x}), \tag{19}$$

Therefore,

$$\mathcal{R}_k = \mathcal{C}_k \;\; \forall k \in \{1, \ldots K\} \tag{20}$$

and we can from now on work with $\mathcal{R}_k$.

Following the definitions from [2], pages 182-184:
Let $\mathbf{x}_A, \mathbf{x}_B \in \mathcal{R}_k$ and define any point $\hat{\mathbf{x}}$ that lies on the line connecting the two:

$$\hat{\mathbf{x}} = \lambda \mathbf{x}_A + (1 - \lambda)\mathbf{x}_B, \tag{21}$$

where $0 \leq \lambda \leq 1$. Cf. equation 15, the discriminant functions, a_k, are linear and it follows that:

$$a_k(\hat{\mathbf{x}}) = \lambda a_k(\mathbf{x}_A) + (1 - \lambda) a_k(\mathbf{x}_B) \tag{22}$$

for all k. From the definition of $\mathcal{R}_k$, we know that $a_k(\mathbf{x}_A) > a_j(\mathbf{x}_A)$ and $a_k(\mathbf{x}_B) > a_j(\mathbf{x}_B)$ for all $j \neq k$. Therefore, $a_k(\hat{\mathbf{x}}) > a_j(\hat{\mathbf{x}})$ for all $j \neq k$ and $\hat{\mathbf{x}}$ belongs to $\mathcal{R}_k$. Hence, $\mathcal{R}_k = \mathcal{C}_k$ is convex. □

1.4 Runtime complexity

We have a dataset $\mathcal{D}$ with C classes each with N_c data points, N total data points, and each data point has a representation in $\mathbf{z} \in \mathbb{R}^D$.

Graph Convexity: Computing the graph convexity score requires three steps:

1. Compute the nearest neighbor matrix. This cost $1/2N^2D$.

2. Create the adjacency matrix. This cost $1/2N^2$. The adjacency matrix will at most have $E = NK$ edges, where K is the K chosen for K-nearest neighbors
3. Compute the convexity score for each pair of data points for each class or concept using Dijkstra's algorithm. This cost $1/2CN_c(N_c - 1)(N \log N + E)$.

The total computational cost is then

$$\text{cost} = 1/2N^2D + 1/2N^2 + 1/2CN_c(N_c - 1)\left(N \log N + NK\right) \tag{23}$$

This simplifies to

$$\text{cost} = \mathcal{O}\left(N^2(D+1) + CN_c(N_c - 1)N\left(\log N + K\right)\right) \tag{24}$$

Since $CN_c = N$, then if $(N_c - 1)(\log N + K) \gg (D + 1)$, the dominating term will be the latter. This can be seen by rewriting:

$$\text{cost} = \mathcal{O}\left(N^2(D+1) + N^2(N_c - 1)\left(\log N + K\right)\right) \tag{25}$$

Instead of computing the convexity score for a graph exact, we can estimate it using N_s samples per class/concept, we get

$$\text{cost} = \mathcal{O}\left(N^2(D+1) + CN_s(N_s - 1)N\left(\log N + K\right)\right) \tag{26}$$

Euclidean Convexity: Computing the Euclidean convexity score for each class/concepts entails the following operations

1. Sample N_p point pairs, denoting their representations $\mathbf{z}_1$ and $\mathbf{z}_2$. This operation scales with N_p.
2. Per point pair, generate N_s representations equidistant on a line between $\mathbf{z}_1$ and $\mathbf{z}_2$. This scales with N_s .
3. For each generated representation, compute the classification. This scales with D.

Since all the costs are scaling linearly, the total cost of computing the Euclidean convexity score is

$$\text{cost} = \mathcal{O}\left(CN_sN_pD\right) \tag{27}$$

1.5 Voronoi tessellation

If we classify based on closeness to a set of prototypes, we get convex decision regions. Formally:

Theorem 4. Let $\{\mathbf{p}_i\}_{i=1}^{C} \subset \mathbb{R}^d$ be a finite set of points, *prototypes*, in $\mathbb{R}^d$ and for all $\{1, \ldots, C\}$, we define

$$R_i = \left\{\mathbf{x} \in \mathbb{R}^d : \forall j \in \{1, \ldots, C\} \setminus \{i\} : \|\mathbf{x} - \mathbf{p}_i\|_2 < \|\mathbf{x} - \mathbf{p}_i\|_2\right\}. \tag{28}$$

Then $\forall i \in \{1, \ldots, C\} : R_i$ is Euclidean-convex.

Proof. Let $\mathbf{x}, \mathbf{y} \in R_i$, $j \in \{1, \ldots, C\}, j \neq i$, and $t \in (0, 1)$ and consider a point on the segment between $\mathbf{x}$ and y, $t\mathbf{x} + (1 - t)\mathbf{y}$. Since $\mathbf{x} \in R_i$, the following holds for any $j \in \{1, \ldots, C\} \setminus \{i\}$:

$$\|\mathbf{x}\|_2^2 + \|\mathbf{p}_i\|_2^2 - 2\mathbf{x} \cdot \mathbf{p}_i = \|\mathbf{x} - \mathbf{p}_i\|_2^2 < \|\mathbf{x} - \mathbf{p}_i\|_2^2 = \|\mathbf{x}\|_2^2 + \left\|\mathbf{p}_j\right\|_2^2 - 2\mathbf{x} \cdot \mathbf{p}_j. \tag{29}$$

Hence,

$$\|\mathbf{p}_i\|_2^2 - 2\mathbf{x} \cdot \mathbf{p}_i < \left\|\mathbf{p}_j\right\|_2^2 - 2\mathbf{x} \cdot \mathbf{p}_j. \tag{30}$$

Similarly, the same is true for y. We use these inequalities in the following:

$$\|t\mathbf{x} + (1-t)\mathbf{y} - \mathbf{p}_i\|_2^2 = \|t\mathbf{x} + (1-t)\mathbf{y}\|_2^2 + \|\mathbf{p}_i\|_2^2 - 2(t\mathbf{x} + (1-t)\mathbf{y}) \cdot \mathbf{p}_i \tag{31}$$

$$= \|t\mathbf{x} + (1-t)\mathbf{y}\|_2^2 + +t(\|\mathbf{p}_i\|_2^2 - 2\mathbf{x} \cdot \mathbf{p}_i) + (1-t)(\|\mathbf{p}_i\|_2^2 - 2\mathbf{x} \cdot \mathbf{p}_i) \tag{32}$$

$$< \|t\mathbf{x} + (1-t)\mathbf{y}\|_2^2 + +t(\left\|\mathbf{p}_j\right\|_2^2 - 2\mathbf{x} \cdot \mathbf{p}_j) + (1-t)(\left\|\mathbf{p}_j\right\|_2^2 - 2\mathbf{x} \cdot \mathbf{p}_j) \tag{33}$$

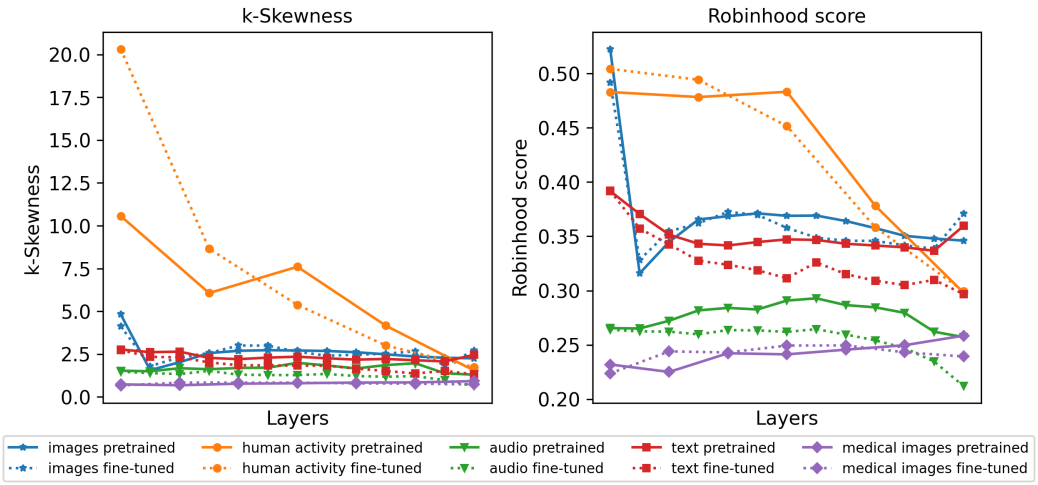
$$= \left\|t\mathbf{x} + (1-t)\mathbf{y} - \mathbf{p}_j\right\|_2^2. \tag{34}$$

Hence, $t\mathbf{x} + (1 - t)\mathbf{y} \in R_i$ and, therefore, R_i is convex. □

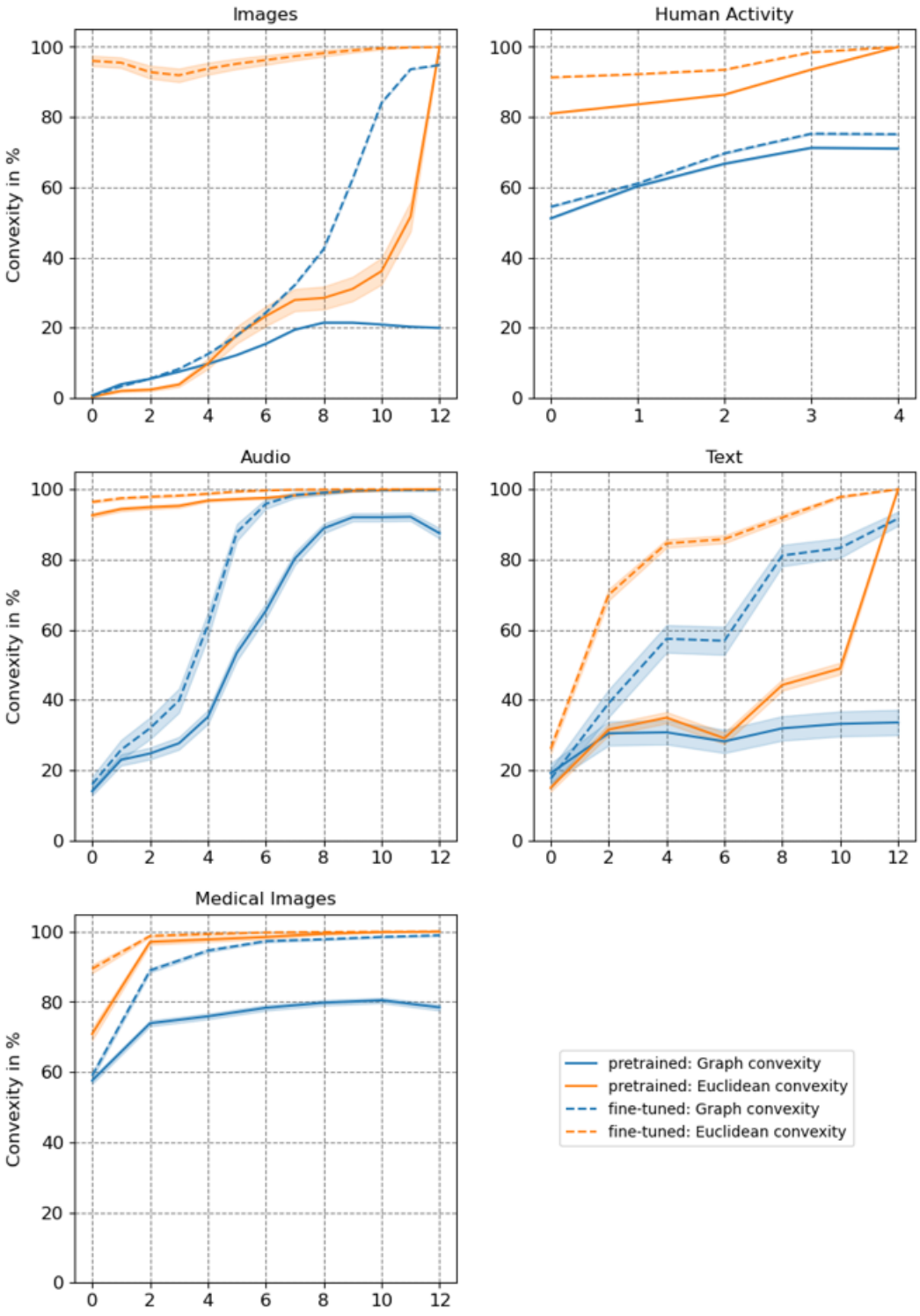
2 Additional Results

2.1 Hubness analysis

Supplementary Figure 1 shows hubness metrics (k-skewness and Robinhood score) for all models and domains. It follows that hubness is not a problem for any of the domains. Supplementary Figure 2 shows the individual convexity results for all domains including the standard error of the mean. We observe an increase in convexity across layers for all models and an increased convexity in fine-tuned models.



Supplementary Figure 1: Hubness evaluation for all modalities. No measures to correct for hubness were taken, as the numbers for k-skewness and Robinhood score are generally low.



Supplementary Figure 2: Individual convexity results for all domains. Error bars show the error of the mean. All results are aggregated in Fig. 3b in the main paper. We observe an increase in convexity across layers for all models and an increased convexity in fine-tuned models.

	graph convexity	accuracy	ID: DANCO	ID: PCA 10	ID: kNN 10
graph convexity					
accuracy	0.265				
ID: DANCO	0.087	0.083			
ID: PCA 10	-0.072	0.015	0.012		
ID: kNN 10	0.042	0.019	-0.007	-0.033	

Supplementary Table 1: Pearson correlations for data2vec: graph convexity in the pretrained model (mean over all layers), accuracy of the fine-tuned model, intrinsic dimension (in the last layer Nature Communications of the pretrained model) estimated using the Dimensionality from Angle and Norm Concentration algorithm (DANCO) [4], the PCA algorithm (10 nearest neighbors) [5], and the kNN algorithm (10 nearest neighbors) [6]. Correlation is computed for 1000 ImageNet classes.

2.2 Comparison to other latent space metrics

To show how similar is the proposed convexity score to existing metrics evaluating latent spaces, we measure Pearson correlations for graph convexity in the pretrained model, accuracy in the fine-tuned model, and three estimators or intrinsic dimension. The intrinsic dimension is the minimal number of parameters needed to accurately describe the class representations learned by the network. Ansuini et al. show that the intrinsic dimension of the last hidden layer predicts classification performance [3].

We report the correlation coefficients in Supplementary Table 1. It shows that the graph convexity scores are distinct from all estimators of intrinsic dimension. Moreover, the stronger relationship is between graph convexity and accuracy, i.e. the relation this paper focuses on.

2.3 Application of convexity in an unsupervised setting

While the main use of the presented work is in a supervised setting, here we suggest how it could be extended beyond known labels. In an unsupervised setting, the class/concept labels could be cluster assignments determined by any clustering method. With these labels, the rest of the analysis can be done in the same way as in the supervised setting. Here we show that the graph convexity of clusters could be used as an additional criterion for evaluating the quality of the clusterings.

For this experiment, we use a subset of 100 randomly chosen ImageNet classes and the corresponding images from the validation set, i.e. in total 5000 images. We extract representations of the last layer in data2vec [7]. We choose four clustering methods (k-means clustering [8], spectral clustering, Gaussian mixture, Ward's hierarchical agglomerative clustering [9]) and various combinations of hyperparameters for them: number of clusters for all the methods (10, 50, 100), initialization strategy for k-means (random, k-means++ [10]), and two hyperparameters for spectral clustering: label assignment (k-means, discretize) and method for construction of the affinity matrix (nearest neighbors, radial basis function kernel). These combinations give us in total 24 clustering methods.

We evaluate each clustering with a number of methods. First, we evaluate the graph convexity (with labels assigned by the clusterings). Since we want to compare results for different number of clusters/classes, we need to normalize the convexity score, so that it becomes independent of this number. We do it in the following way:

$$\text{convexity}_{\text{normalized}} = \frac{\text{convexity} - b}{1 - b}, \tag{35}$$

where b is the baseline convexity score depending on the number of classes/clusters ($b = \frac{1}{n_{\text{classes}}}$).

Since, in this case, we know the ImageNet labels, we can use them to evaluate the clustering. Specifically, we compute the adjusted Rand index (number of agreeing pairs over the number of all pairs, corrected for chance) [11] and adjusted mutual information (mutual information adjusted to account for chance) [12].

We also evaluate the clustering without any labels with Silhouette coefficient [13] (using the mean intra-cluster distance and the mean nearest-cluster distance), Calinski-Harabasz [14] (ratio of the sum of between-cluster dispersion and of within-cluster dispersion), and Davies-Bouldin [15] (average similarity measure of each cluster with its most similar cluster, where similarity is the ratio

	Convexity	Rand	MI	Silhouette	C-H	D-B
Convexity normalized						
Rand adjusted	0.632					
Adjusted mutual information	0.913	0.868				
Silhouette	0.862	0.68	0.855			
Calinski-Harabasz	0.751	0.068	0.473	0.594		
Davies-Bouldin	-0.713	-0.636	-0.764	-0.41	-0.449	

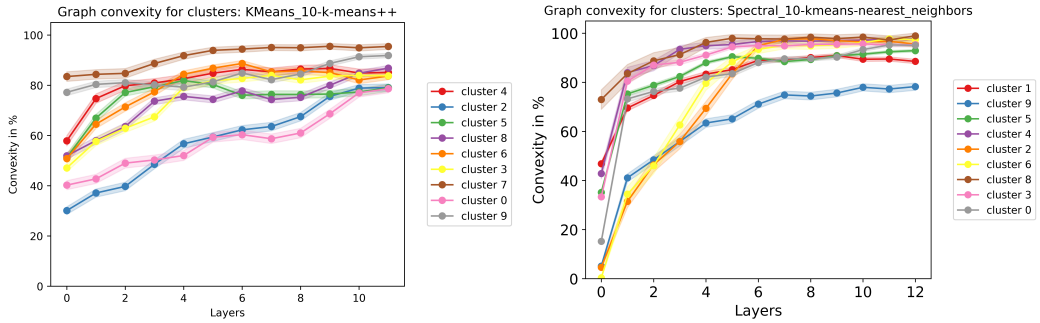
Supplementary Table 2: Pearson correlations for convexity and clustering evaluation metrics for 24 clustering methods on representations of the data2vec pretrained model. More details in Section 2.3. Note that the negative correlations with Davies-Bouldin are caused by the fact that lower Davies-Bouldin score indicates better clustering, whereas it is the opposite for the rest of the scores.

of within-cluster distances to between-cluster distances). A better clustering gets lower Davies-Bouldin score, whereas in the other score higher values mean better separation between the clusters. A potential drawback of these three scores is that they in general assign higher values to convex clusters, hence, are likely to be highly correlated with our convexity score.

We compute Pearson correlation for these scores (over all the clustering methods). The results can be found in Supplementary Table 2. We see that the convexity score is strongly correlated to other metrics estimating the quality of clustering. However, the correlations are not perfect, indicating that convexity can be used as an additional metric picking up different qualities.

Supplementary Figure 3 shows how the graph convexity of clusters evolves throughout the network. Analyses like this can be used for a thorough investigation of the quality of clustering and identification of different types of clusters.

In the case of lower number of clusters than the original 100 ImageNet classes, these clusters can be seen as natural "superclasses" of the selected ImageNet classes. Convexity is a good criterion also in this case because it can give additional information on the coherence of the "superclasses".

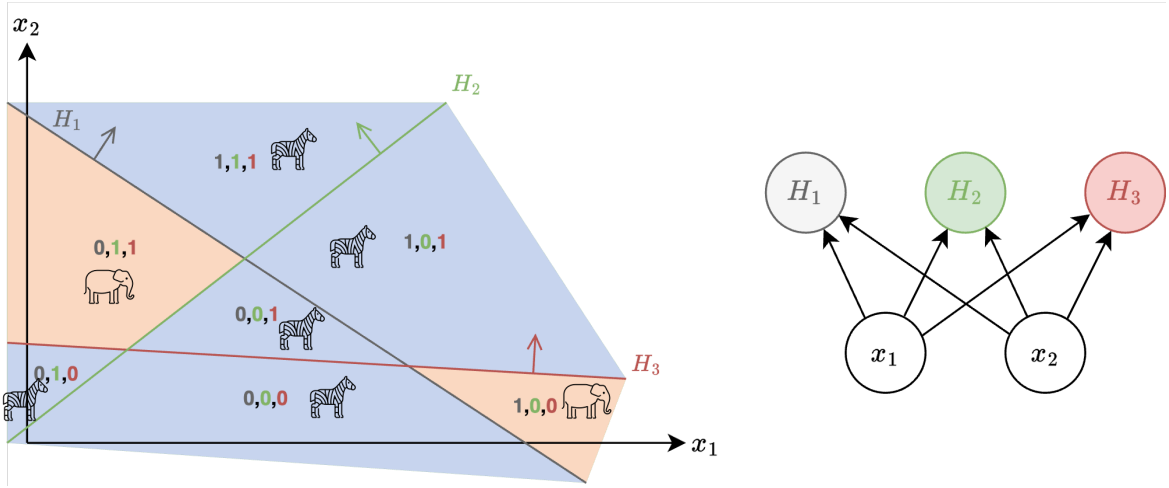


Supplementary Figure 3: Evolution of graph convexity scores of each cluster for two clustering methods: k-means (left) and spectral clustering (right).

2.4 Comparison of graph convexity to the Polytope lens

We will discuss how measuring the graph convexity captures different properties compared to the existing analysis tool, the Polytope lens [16]. Both our work and the polytope lens compute distances between points of the same class in representation space. The polytype lens computes the Hamming Distance (HD) between activations of two points of the same class, whereas the graph convexity score (GCS) computes the proportion of the shortest path (on the graph) that is contained within the same class. The two methods capture different properties and are complementary methods.

To illustrate, consider the two-class problem depicted on Supplementary Figure 4, where an MLP with two activations in the input layer, denoted x_1 and x_2, and three activations in the subsequent layer, denoted H_1, H_2, and H_3, is used. For simplicity, we assume that each of the seven regions has the same number of data points and that the data points are uniformly distributed within the region. Analyzing this network with the polytope lens will yield a HD between 0 (class points are



Supplementary Figure 4: The hyperplanes and the decision regions created by an MLP with two input activations and three output activations. The code in each region denotes whether the given activation H_i is activated.

compact within one of the seven regions) and 3 (class points are distributed so that all 3 neurons flip activation). Analyzing the network with the convexity score will yield a GCS between 0 (no paths will be in the class region) and 1 (the entire path will be within the class region).

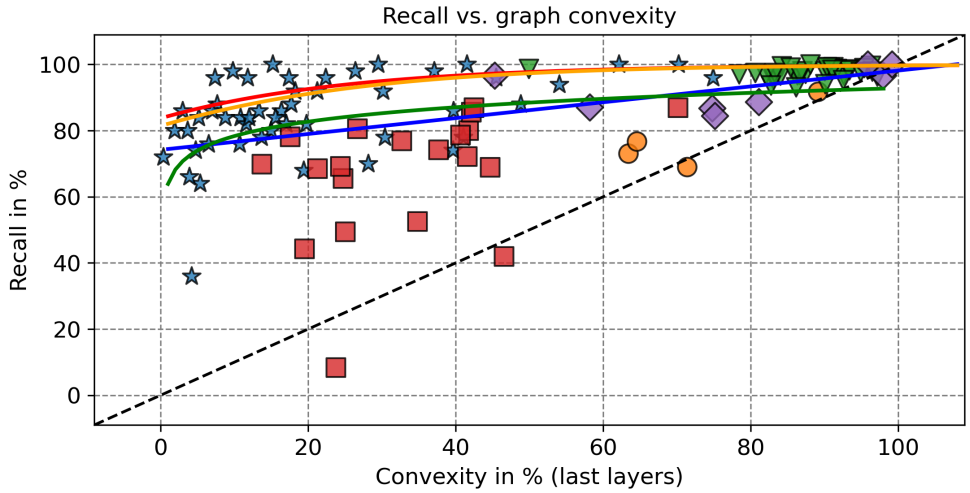
For the Elephant class, we get HD(Elephant) = 3 and GCS(Elephant) = 0.5, whereas for the Zebra class, we get HD(Zebra) = 1.8 and GCS(Zebra) = 1. Thus both methods find that the Elephant class is widely distributed compared to the Zebra class. However, GCS(Zebra) = 1 finds that the Zebra class is fully connected, whereas the HD(Zebra) = 1.8 indicates that the Zebra class is widely distributed. Considering the analysis of both the polytope lens and graph convexity, we can thereby conclude that the Zebra class is fully connected but still distributed. This conclusion can only be made by using both methods, thereby demonstrating HD and GCS are complementary analysis tools.

2.5 Relation between graph convexity and accuracy after fine-tuning

We used the data from Figure 3c from the manuscript (per-class convexity scores and accuracies) to fit three models to capture the relationship between graph convexity and accuracy after fine-tuning:

1. linear (M1): recall $= a \cdot \text{conv} + b$;
2. logarithmic (M2): recall $= a \cdot \log(\text{conv}) + b$;
3. log-linear (M3): $\log(\text{1-recall}) = a \cdot \text{conv} + b$;
4. log-linear transformed (M4): $\log\left(\frac{1-\text{recall}}{\text{recall}}\right) = a \cdot \text{conv} + b$;

Supplementary Figure 5 shows the fitted models for all modalities. Supplementary Table 3 shows a statistical analysis for these models. The only significant conclusion (with $\alpha = 0.05$) is that the fitted linear model explains the data better than a constant (mean of recalls). Even though the data visually suggest a log-linear relationship, the statistical analysis does not support this hypothesis.



Supplementary Figure 5: Graph convexity of a subset of classes in the pretrained models (last layer) vs. recall rate of these individual classes in the fine-tuned models for all data domains. The blue, green, red, and orange lines are the best M1, M2, M3, and M4 models for all modalities combined, respectively.

	all modalities	images	human activity	text	audio	medical
R^2 M1	0.268	0.072	0.690	0.121	0.008	0.183
R^2 M2	0.188	0.041	0.657	0.099	0.002	0.104
R^2 M3	-0.211	-0.257	0.533	0.079	-0.397	-109.684
R^2 M4	-0.136	-0.198	0.582	0.113	-0.386	-14.963
p M1 vs. constant	0.952	0.882	0.763	0.606	0.509	0.594
p M1 vs. M2	0.711	0.700	0.526	0.520	0.506	0.543
p M1 vs. M3	0.996	1.000	0.601	0.539	0.835	1.000
p M1 vs. M4	0.990	1.000	0.574	0.507	0.829	0.999

Supplementary Table 3: Coefficients of determination (R^2) for each model and p-values in F-test between the linear model and the others (*constant* denotes the mean of the recalls). In columns, we have the individual modalites and all combined. Note that *all modalities* includes only 50 image classes (as shown in Figure 3c in the manuscript and in Supplementary Figure 5, whereas *images* takes into account all 1000 classes.

Supplementary References

[1] Boyd, S. P. & Vandenberghe, L. Convex optimization (Cambridge Univ. Press, 2010).

[2] Bishop, C. M. & Nasrabadi, N. M. Pattern recognition and machine learning (Springer, 2006).

[3] Ansuini, A., Laio, A., Macke, J. H. & Zoccolan, D. Intrinsic dimension of data representations in deep neural networks. Advances in Neural Information Processing Systems **32** (2019).

[4] Ceruti, C. et al. Danco: dimensionality from angle and norm concentration. arXiv preprint arXiv:1206.3881 (2012).

[5] Fan, M., Gu, N., Qiao, H. & Zhang, B. Intrinsic dimension estimation of data by principal component analysis. arXiv preprint arXiv:1002.2050 (2010).

[6] Carter, K. M., Raich, R. & Hero III, A. O. On local intrinsic dimension estimation and its applications. IEEE Transactions on Signal Processing **58**, 650–663 (2009).

[7] Baevski, A. et al. Data2vec: A general framework for self-supervised learning in speech, vision and language. In International Conference on Machine Learning, 1298–1312 (PMLR, 2022).

[8] Lloyd, S. Least squares quantization in pcm. IEEE transactions on information theory **28**, 129–137 (1982).

[9] Ward Jr, J. H. Hierarchical grouping to optimize an objective function. Journal of the American statistical association **58**, 236–244 (1963).

[10] Arthur, D. & Vassilvitskii, S. k-means++: The advantages of careful seeding. Tech. Rep., Stanford (2006).

[11] Hubert, L. & Arabie, P. Comparing partitions. Journal of classification **2**, 193–218 (1985).

[12] Vinh, N. X., Epps, J. & Bailey, J. Information theoretic measures for clusterings comparison: is a correction for chance necessary? In Proceedings of the 26th annual international conference on machine learning, 1073–1080 (2009).

[13] Rousseeuw, P. J. Silhouettes: a graphical aid to the interpretation and validation of cluster analysis. Journal of computational and applied mathematics **20**, 53–65 (1987).

[14] Caliński, T. & Harabasz, J. A dendrite method for cluster analysis. Communications in Statistics-theory and Methods **3**, 1–27 (1974).

[15] Davies, D. L. & Bouldin, D. W. A cluster separation measure. IEEE transactions on pattern analysis and machine intelligence 224–227 (1979).

[16] Black, S. et al. Interpreting neural networks through the polytope lens. arXiv preprint arXiv:2211.12312 (2022).