Centroid Affinity: How Deep Networks Represent Features

Thomas Walker[*] Rice University

Ahmed Imtiaz Humayun[†] Rice University

Randall Balestriero Brown University

Richard Baraniuk Rice University

Abstract

Understanding and identifying the features of a deep network (DN) is a focal point of interpretability research. A common characterisation of the features of a DN is that of directions in their latent spaces, known as the linear representation hypothesis (LRH). However, there are increasingly apparent limitations of the LRH and calls for strategies for understanding the *functional behaviours* of a DN's features. In this work, we explore the connection between a DN's *functional geometry* and its features. We demonstrate how a vector-summarisation of a DN's Jacobians – called centroids – possesses a semantically coherent affine structure that arises from the linear *separability* of latent activations. Thus, we introduce *centroid affinity* as a complementary perspective to the LRH that is grounded in the functional properties of the DN. Importantly, we can continue to utilise LRH-leveraging tools, such as sparse autoencoders, to study the features of a DN through centroid affinity; with centroid affinity also facilitating the introduction of novel measures for exploring the features and circuits of DNs. Indeed, we demonstrate how centroid affinity can effectively and robustly interpret the features of the DINOv2 and GPT2 models. The corresponding code for this work can be found here.

1 Introduction

What are the features of a deep network (DN)? Fundamentally, DNs systematically intertwine linear and nonlinear operations in a layer-wise fashion to induce expressive function approximators [2]. Thus, features should refer to the regions of the input space that have a determined influence on the functional behaviour of the DN. With this, the practical question becomes *how do we identify the features of a DN?*

In this paper, we propose an approach for identifying features using the *functional geometry* of a DN. Using the spline theory of deep learning [3], we demonstrate that a parametrisation of a DN's functional geometry represents *meaningful* features as affine structures, a notion we formalise as *centroid affinity*. Since this parametrisation is accessible through Jacobian vector products, it follows that centroid affinity offers an appealing approach for identifying the features of a DN.

To arrive at this, we first utilise prior works to introduce the notion of a feature and circuit of a DN, which we then explore through the perspective of a DN's nonlinearities. Using the spline theory of deep learning, we contextualise this within the power diagram parametrisation of a DN's functional

[*]Corresponding author: `thomas.walker@rice.edu`

[†]Now at Google Research.

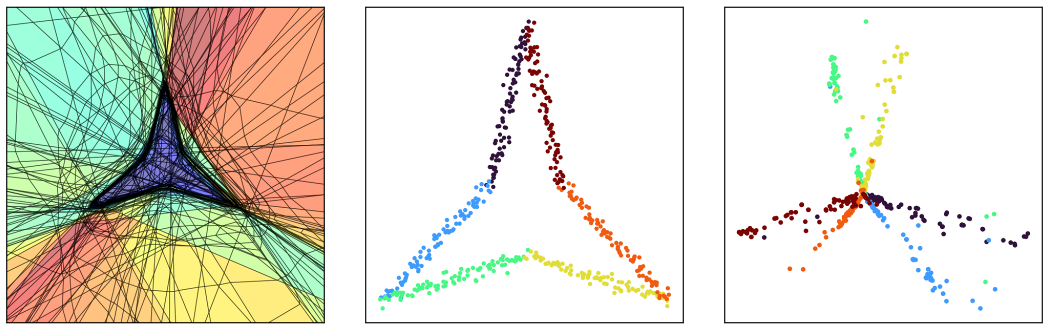


Figure 1: Centroid affinity identifies the meaningful features of a DN. Here we visualise the centroids of a DN trained to classify the interior of a star-shaped polygon in the two-dimensional plane (see Appendix A). More specifically, we sample points in the input space, **centre** plot, and visualise their corresponding centroids, **right** plot. Centroids are computed using Proposition 2.3. The corresponding functional geometry of the DN is visualised in the **left** plot (using SplineCam [1]).

geometry [4]. Allowing us to introduce centroid affinity as a complementary perspective to the prevailing linear representation hypothesis (LRH) [5, 6], which posits that features are represented as linear structures in the latent spaces of a DN.

Evidence for the LRH was first teased out using text semantics in word-embedding models [7], before being supported in transformers [8]. The utility of this perspective is that it is simple and intuitive; which facilitated the development of a variety of computationally efficient tools to perform feature extraction [9–13] and understand the behaviours of DNs [14–19]. Furthermore, it is amenable to theoretical characterisation, allowing for the systematic study of DN *features* [6, 20, 21]. However, there is a growing consensus that the LRH is limited in its capacity to provide a meaningful understanding of the DN's function [22] that is consistent [23]. Fundamentally, it is agnostic to the input-output mapping [17, 24, 25] and is not easily contextualised within the DN's input space [24]. Thus, it is difficult to interpret the dictionaries of the so-called features extracted by these methods, which are often on the order of millions [26].

Thus, secondly, we demonstrate that centroid affinity overcomes these limitations of the LRH to facilitate a function-aware study into the features of a DN. In particular, by considering a simple example, we demonstrate explicitly how centroid affinity ignores functionally meaningless features of a DN that are instead identified under the LRH.

Although more functionally aware approaches for identifying the features of a DN are being proposed [27, 28], often only an approximation of their theoretical construction can be implemented in practice [29, 30]. Despite this, these approaches have offered a range of intuitive and interpretable insights [31, 32].

So thirdly, we highlight how centroid affinity can efficiently explore the features of the DINOv2 [33] and GPT2 [34] models in a similarly insightful way. For the DINOv2 model, by training sparse autoencoders on centroids, we identify sparser, more meaningful, and more functionally relevant features compared to the features of sparse autoencoders trained on latent activations. On GPT2, we utilise novel centroid-based metrics to corroborate prior circuit analysis work [35].

2 Background

Features and Circuits. As articulated in Section 1, the features of a DN should relate the input space to the functional mapping.

Definition 2.1. A feature of a DN is a region of the input space that predictably influences the functional behaviour of the DN.

Influencing the functional behaviour of the DN could refer to anything from inducing a particular activation pattern in its nonlinearities, influencing the output logits in a consistent manner, or triggering an attention head, for example. We formalise the precise functional influence of a feature with circuits.

Definition 2.2. A circuit of a DN constitutes a sub-component of the DN's computational graph that is influenced by a feature.

In Appendix B, we explain how this formalisation of features and circuits is analogous to prior work.

Deep Networks and their Approximations. An L layer DN $f : \mathbb{R}^d \to \mathbb{R}^{d^{(L)}}$, with the convention that $d^{(0)} = d$, is a composition of L functions $f^{(\ell)} : \mathbb{R}^{d^{(\ell-1)}} \to \mathbb{R}^{d^{(\ell)}}$ which usually constitute some sort of affine transformation followed by a nonlinearity. Using the spline approximation theory [36, 37], DNs can be approximated to arbitrary precision or even characterised exactly [3] using continuous piecewise affine splines. More specifically, one can write $f(\mathbf{x}) \approx \mathbf{A}_{\omega_{\boldsymbol{\nu}(\mathbf{x})}}\mathbf{x} + \mathbf{b}_{\omega_{\boldsymbol{\nu}(\mathbf{x})}}$, where $\mathbf{A}_{\omega_{\boldsymbol{\nu}(\mathbf{x})}} \in \mathbb{R}^{d^{(L)} \times d}$ and $\mathbf{b}_{\omega_{\boldsymbol{\nu}(\mathbf{x})}} \in \mathbb{R}^{d^{(L)}}$ are affine parameters specific to the *linear region* $\omega_{\boldsymbol{\nu}(\mathbf{x})} \subseteq \mathbb{R}^d$ encompassing $\mathbf{x}$. Here $\boldsymbol{\nu}(\mathbf{x})$ identifies the equivalence class of $\mathbf{x}$ under the collection of all equivalence classes $\mathcal{V}$ constructed by $\sim$ where $\mathbf{x}_1 \sim \mathbf{x}_2$ if and only if $\mathbf{x}_1$ and $\mathbf{x}_2$ are in the same linear region. In particular, one can construct these approximations for each function $f^{(\ell)}$ to obtain

$$f(\mathbf{x}) \approx \mathbf{A}^{(L)}_{\omega^{(L)}_{\boldsymbol{\nu}(\mathbf{x})}} \left(\dots \left(\mathbf{A}^{(1)}_{\omega^{(1)}_{\boldsymbol{\nu}(\mathbf{x})}} \mathbf{x} + \mathbf{b}^{(1)}_{\omega^{(1)}_{\boldsymbol{\nu}(\mathbf{x})}} \right) \dots \right) + \mathbf{b}^{(L)}_{\omega^{(L)}_{\boldsymbol{\nu}(\mathbf{x})}},$$

where $\omega^{(\ell)}_{\boldsymbol{\nu}(\mathbf{x})} \subseteq \mathbb{R}^{d^{(\ell-1)}}$ denotes the linear region for the mapping $f^{(\ell)}$ encompassing $f^{(1\leftarrow\ell-1)}(\mathbf{x}) \in \mathbb{R}^{d^{(\ell-1)}}$.[3] Similarly, one can compute continuous piecewise affine approximations for sub-components of the DN, say $f^{(\ell_1\leftarrow\ell_2)} : \mathbb{R}^{d^{(\ell_1-1)}} \to \mathbb{R}^{d^{(\ell_2)}}$ for $1 \leq \ell_1 < \ell_2 \leq L$, for which we adopt a similar notation. When the DN employs continuous piecewise nonlinearities (e.g. ReLU), these spline approximations are exact [3].

Henceforth, when we speak in terms of DN sub-components, we do so with the understanding that this covers everything from a single layer to the entire DN.

Functional Geometry. The *functional geometry* of a DN sub-component refers to the arrangement of the linear regions of its continuous piecewise affine approximation. Namely, the functional geometry of the DN sub-component $f^{(\ell_1\leftarrow\ell_2)}$ is the disjoint union of $\left\{\omega^{(\ell_1\leftarrow\ell_2)}_{\boldsymbol{\nu}}\right\}_{\boldsymbol{\nu}\in\mathcal{V}}$, which is a partition of $\mathbb{R}^{d^{(\ell_1-1)}}$ into convex polytopes [4]. The particular arrangement of the linear regions characterises surprisingly many properties of the sub-component [38–40].

On the one hand, the linear regions can be thought of as being bounded by the level-sets of the nonlinearities of the DN sub-component. Each nonlinearity has a level-set[4] which is a hyperplane in its input space. As the hyperplane is projected back to the input space of the sub-component, it bends at the point of intersection with the level-sets of the preceding nonlinearities (see Figure 7). It is the intersection of these planes that forms the regions which constitute the DN sub-component's functional geometry [1]. In particular, through this process one can contextualise the functional geometry of a component $f^{(\ell_1\leftarrow\ell_2)}$ into the input space of the DN, $\mathbb{R}^d$.

On the other hand, it was shown in Balestriero et al. [4] that the functional geometry of $f^{(\ell_1\leftarrow\ell_2)}$ can be parametrised by a power diagram subdivision using a collection of centroid-radius pairs $\left\{\left(\mu^{(\ell_1\leftarrow\ell_2)}_{\boldsymbol{\nu}}, \tau^{(\ell_1\leftarrow\ell_2)}_{\boldsymbol{\nu}}\right)\right\}_{\boldsymbol{\nu}\in\mathcal{V}} \subseteq \mathbb{R}^{d^{(\ell_1-1)}} \times \mathbb{R}$, such that

$$\omega^{(\ell_1\leftarrow\ell_2)}_{\boldsymbol{\nu}} = \left\{ \mathbf{x} \in \mathbb{R}^{d^{(\ell_1-1)}} : \boldsymbol{\nu} = \arg\min_{\boldsymbol{\nu}'\in\mathcal{V}} \left(\left\| \mathbf{x} - \mu^{(\ell_1\leftarrow\ell_2)}_{\boldsymbol{\nu}'} \right\|_2^2 - \tau^{(\ell_1\leftarrow\ell_2)}_{\boldsymbol{\nu}'} \right) \right\}.$$

Proposition 2.3 (Balestriero et al. 4)**.** *Let* $\mathbf{J}_{\mathbf{x}}\left(f^{(\ell_1\leftarrow\ell_2)}\right)$ *denote the Jacobian of* $f^{(\ell_1\leftarrow\ell_2)}$ *at* $f^{(1\leftarrow\ell_1-1)}(\mathbf{x})$*. Then,* $\mu^{(\ell_1\leftarrow\ell_2)}_{\boldsymbol{\nu}(\mathbf{x})} = \left(\mathbf{J}_{\mathbf{x}}\left(f^{(\ell_1\leftarrow\ell_2)}\right)\right)^\top \mathbf{1}$.

[3] It should be understood that in this context, $\boldsymbol{\nu}(\mathbf{x})$ identifies the equivalence class on the linear regions in $\mathbb{R}^{d^{(\ell-1)}}$.

[4] By level-set, we refer to the points in space – whether that be in the input space of the DN or the input space of the nonlinearity – that activate the nonlinearity at its knots. For example, for the ReLU nonlinearity, the level set would refer to the points that are zero when fed into the nonlinearity.

From Proposition 2.3 it follows that the functional geometry can be parametrised with a vector that is computationally accessible through a Jacobian vector product.

3 Centroid Affinity Identifies Features

When the spline approximation is exact, say in the context of continuous piecewise affine DNs, clearly linear regions satisfy Definition 2.1. Indeed, the inputs within a linear region, by construction, induce the same activation pattern within the DN (see Figure 7). However, since DNs contain exponentially many linear regions [41], for purposes of interpretability, we need to refine the scope of Definition 2.1 to *meaningful* features.

Meaningful Features as Aligned Linear Regions. Humayun et al. [42] characterised the grokking phenomenon in DNs [43] – the observation that test accuracy can take substantially longer to saturate compared to the train accuracy – as the migration of linear regions from the training data to the decision boundary; which was compared with the phenomenon of circuit clean-up [44] – the observation that redundant circuits are discarded as the DN generalises. Since Humayun et al. [42] demonstrated that the region migration phenomenon – alignment of linear regions along a boundary in the input space – is a *universal* phenomenon of DN training dynamics and has a link to grokking, we will use this as an indicator of the *meaningful* features of a DN.

Definition 3.1 (*Informal*)**.** A feature of a DN is a collection of *aligned* linear regions in the input space. (Formalised in Appendix C).

We will now further support and articulate Definition 3.1 with a simple example. Consider a fully connected ReLU network – with three hidden layers – of the form $f : \mathbb{R}^2 \to \mathbb{R}$, trained to perform the binary classification of a polygon in a two-dimensional plane (see Appendix A). More specifically, points within the polygon have label zero and points outside the polygon have label one. In this instance, we would expect a feature of the DN to be the polygon.

Let us consider the last component of the DN, namely $g(\mathbf{z}) = \mathbf{W}\sigma(\mathbf{z})$,[5] and focus on the k^{th} nonlinearity. Suppose $\epsilon\mathbf{e}_k$ perturbs the input point $\mathbf{z}$ such that the nonlinearity becomes active.[6] Then $g(\mathbf{z} + \epsilon\mathbf{e}_k) = g(\mathbf{z}) + ([\mathbf{z}]_k + \epsilon)\mathbf{W}\mathbf{e}_k$. That is, such a perturbation is going to influence the confidence of the predicted class. Importantly, this contribution will not be present when the nonlinearity is inactive. Thus, regardless of the nature of this contribution, the DN is *incentivised* to place the level-set of the nonlinearity in a region where the output should change from one class to another. Therefore, the level-set of the nonlinearities should align themselves along the boundary of the polygon, which subsequently supports the characterisations of features as per Definition 3.1.

Connection to the LRH. The level-set of a nonlinearity is a hyperplane in *its* input space, meaning it can only identify boundaries in the input space of the DN when they are linearly identifiable in its input space. For example, the k^{th} nonlinearity of f can only utilise the polygon as a feature when the points inside and outside the polygon are linearly separable in the input space of the last layer of the DN. Indeed, in Figure 2 we see that the entire shape of the polygon is captured in the third layer, with the corresponding activations of the interior and exterior of the polygon being linearly separable.

By generalising these arguments, we have that the features of the ℓ^{th} layer of a DN are the regions of the input space whose boundary is linearly identifiable within the input space of the ℓ^{th} layer and populated with the level-sets of the nonlinearities of the ℓ^{th} layer.

This highlights a subtle, although complementary, difference from the LRH. The LRH posits that the latent activations of features form affine structures, which is a stronger condition than being linearly identifiable. Furthermore, the way the LRH is utilised and interpreted implies that any set of activations that form affine structures corresponds to features of the DN [45]. However, from the perspective of hyperplanes, even if a collection of latent activations forms a linearly identifiable boundary, we also require that the nonlinearities align themselves along this boundary to form a feature. For if no nonlinearities align themselves along this boundary, then moving along these affine structures would not induce a distinct difference in the functional behaviour of the DN, and thus not a feature as per Definition 2.1.

[5]For simplicity, we have removed the bias term, although in practice this bias term is present.

[6]Since we are using the ReLU nonlinearity, active refers to the state of the activation being positive.

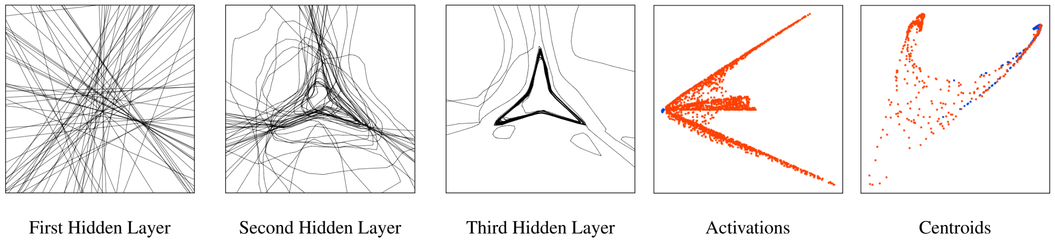


Figure 2: A DN migrates the level-sets of its nonlinearities to meaningful regions of the input space, with earlier layers capturing coarser features which propagate back from the deeper layers. The corresponding centroids form affine subspaces with the latent activations being linearly separable. Here we train a fully connected ReLU DN with three hidden layers on the polygon classification task of Figure 7. The DN has a width of 64 at each hidden layer. We visualise the functional geometry of the first hidden layer component, the second hidden layer component, and the third hidden layer component with the **far left**, **left** and **centre** plots, respectively. In the **right** and **far right** plots, we project the latent activations and centroids in the input space of the third hidden layer onto their first two principal components, respectively. Orange points correspond to input samples from outside of the polygon, and blue points correspond to input samples from inside the polygon.

Therefore, linearity in latent activations is a sufficient representation of features of a DN, but it is not a necessary representation, as seen in Figure 2, where the activations of the interior of the polygon do not correspond to a direction in the latent space. Consequently, our framework provides a strategy for mitigating the identification of spurious features of a DN. For example, in Figure 2 we see that the activations of the external points of the polygon cluster into three directions; however, these will not correspond to different functional behaviours in the last hidden layer of the DN since its nonlinearities do not meaningfully partition the external regions of the polygon. Interestingly, the centroids align along two distinct linear directions. We will now formalise this observation.

Centroid Affinity Identifies Meaningful Features. Above we supported Definition 3.1, by considering the hyperplanes of the nonlinearities forming the regions. Fortunately, for practical purposes, we can reformulate this within the power diagram perspective of the functional geometry.

Proposition 3.2 (*Informal*)**.** *The features of the ℓ^{th} layer of a DN are the collections of regions in its input space, $\mathbb{R}^{d^{(\ell-1)}}$, whose corresponding centroids form an affine subspace in $\mathbb{R}^{d^{(\ell-1)}}$. (Formalised and Proved in Appendix C).*

To contextualise the features of the ℓ^{th} layer in a larger sub-component, it suffices to propagate the regions back into the input space of the sub-component. As mentioned previously, this is done by iteratively taking the intersection with the regions of the previous layers of the sub-component, resulting in the refinement of the regions. For the corresponding centroids, it follows from Proposition 2.3 that we perform a linear projection based on the activation pattern of the points in the region at the previous layers of the sub-component, which refines the original centroids into multiple centroids. Importantly, this refinement maintains the affine structure of the original centroids, although now a single feature at the ℓ^{th} layer may partition into multiple refined features in the input space of the sub-component. Thus, feature analyses of larger sub-components will often identify more granular features than feature analyses on smaller components. We formalise this reasoning with Theorem 3.3.

Theorem 3.3 (*Informal*)**.** *The features of a DN sub-component are the collections of regions in its input space whose centroids form affine subspaces. (Formalised in Appendix C).*

To make this concrete, recall Figure 2 where the nonlinearities aligning along the boundary of the polygon identify the polygon as a feature of the last layer. Similarly, the second hidden layer identifies the three sides of the polygon as features, as the nonlinearities align along these boundaries (evidenced by the extending hyperplanes emanating from each of the three tips of the polygon). Consequently, with the far right plot of Figure 2, we support Proposition 3.2, since the centroids of the interior and exterior of the polygon form affine subspaces in the input space of the last layer. Whereas with Figure 1 we support Theorem 3.3 since the centroids of the linear regions bounding the edges of the polygon form affine subspaces segmented according to which edge they identify; vividly demonstrating the hierarchical feature extraction capabilities arising due to their compositional nature of DNs. In particular, it becomes apparent from analysing the centroids that the features of the first hidden layer ought to be further dividing the three sectors of the polygon identified in the second hidden layer into the individual edges of the polygon – note this observation is challenging to make by just analysing the nonlinearities in Figure 2.

In light of this and Theorem 3.3, we characterise this formalisation of the features of a DN as *centroid affinity*. In Section 4, we will explore how centroid affinity can be used to identify the features of a DN.

Centroid Stability Identifies Circuits. To identify the circuits of a DN, one needs to quantify the relationship between components of a DN and features, which is usually done through attribution methods [46–48]. Based on our analyses, we propose to use the sensitivity of centroids to manipulations in the components of a DN as an attribution method.

Formally, let f be a DN and $f^{(i,\ell)}$ be the same DN but with the i^{th} neuron of the ℓ^{th} manipulated. Then we can quantify the attribution of neuron i to the features of a collection of samples $\mathcal{N}$ as

$$s_{\mathcal{N}}^{(i,\ell)} := \frac{1}{|\mathcal{N}|} \sum_{\mathbf{x} \in \mathcal{N}} \frac{\left\| \mu_{\mathbf{x}}^{f} - \mu_{\mathbf{x}}^{f^{(i,\ell)}} \right\|_2}{\left\| \mu_{\mathbf{x}}^{f} \right\|_2}, \tag{1}$$

where $\mu_{\mathbf{x}}^{f}$ and $\mu_{\mathbf{x}}^{f^{(i,\ell)}}$ are the centroids of f and $f^{(i,\ell)}$ at $\mathbf{x}$ respectively. By taking $\mathcal{N}$ to be $\mathcal{B}_\epsilon(\mathbf{x}) = \left\{\mathbf{x}' \in \mathbb{R}^d : \|\mathbf{x} - \mathbf{x}'\|_2 < \epsilon\right\}$, we can quantify the attribution of a neuron to the local features of a sample point $\mathbf{x}$.[7]

4 Re-evaluating Interpretability Tools

Based on centroid affinity, we ought to re-evaluate the application of interpretability tools. Here we consider the utilisation of sparse autoencoders [12, 13] and transcoders [49] for feature and circuit extraction, and the analysis of point clouds.

Our development of Theorem 3.3 considered a DN utilising piecewise affine nonlinearities. However, in practice, one often utilises smooth nonlinearities (e.g. GELU [50]). Furthermore, for practical reasons, we found it to be useful to *soften* continuous piecewise affine nonlinearities to smooth nonlinearities when studying centroids. Henceforth, when performing centroid analyses, we will be computing centroids of the softened DN, for example, the one where ReLUs are replaced with GELUs. We justify this in Appendix E.

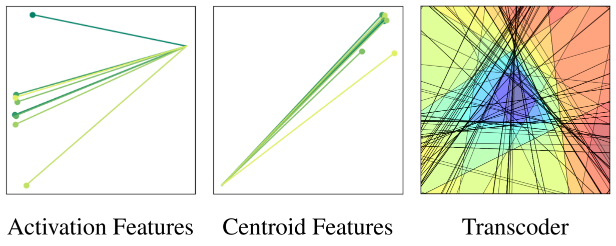


Figure 3: Sparse autoencoders identify the meaningful affine structures present in DN centroids, and are susceptible to identifying linear structures in latent activations that have no functional relevance to the DN. Transcoders are limited in their capacity to reconstruct the features of a DN. In the **left** and **centre** plots, we train a sparse autoencoder to reconstruct the latent activations and softened centroids of the input sample from Figure 9 at the last hidden layer of the DN, respectively. In the **right** plot, we train a transcoder with twice as many features as neurons in the hidden layers of the DN of Section 3 to reconstruct the input-output mapping of this DN. The functional geometry of the DN sub-component is visualised in the left plot of Figure 1.

4.1 Sparse Autoencoders

Sparse autoencoders are a method for extracting an over-complete basis for a set of vectors [12, 13], with the aim of identifying *meaningful* directions. A sparse autoencoder has an architecture of the form $g(\mathbf{z}) = \mathbf{W}_{\text{dec}}\sigma\left(\mathbf{W}_{\text{enc}}\mathbf{z} + \mathbf{b}_{\text{enc}}\right)$, with $\mathbf{W}_{\text{enc}} \in \mathbb{R}^{d^{\text{feat}} \times d^{\text{act}}}$, $\mathbf{b}_{\text{enc}} \in \mathbb{R}^{d^{\text{feat}}}$, $\mathbf{W}_{\text{dec}} \in \mathbb{R}^{d^{\text{act}} \times d^{\text{feat}}}$, where d^{act} is the dimension of the set of vectors and d^{feat} is the size of the over-complete basis which

[7]Henceforth, we will use $s^{(i,\ell)}$ to denote $s^{(i,\ell)}_{\mathcal{B}_\epsilon(\mathbf{x})}$ unless stated otherwise.

is to be constructed. In the context of a DN, a sparse autoencoder is trained to reconstruct its latent activations with an added sparsity regularisation term. The idea is that the rows of $\mathbf{W}_{\text{dec}}$ would then constitute a dictionary of features, with the term $\sigma\left(\mathbf{W}_{\text{enc}}\mathbf{z} + \mathbf{b}_{\text{enc}}\right)$ giving the decomposition of the activation $\mathbf{z}$ in terms of these features.

Under centroid affinity, it is not necessarily the case that the latent activations of features correspond to linear directions. This means that sparse autoencoders may not extract every feature, and may extract spurious features instead. For example, in Figure 3, the sparse autoencoder trained on latent activations identifies directions that are not functionally relevant (see Figure 2). Another problem with this approach is the neglect of any functional information.

These problems can be mitigated by applying sparse autoencoders to reconstruct centroids rather than latent activations. In Figure 3, we see that sparse autoencoders trained on centroids only recover the affine structures of the centroids that correspond to the features of the DN.

4.2 Transcoders

Transcoders have a similar architecture to sparse autoencoders, except they are trained to reconstruct the input-output mapping of a DN sub-component rather than its activations [49]. Although this approach is more faithful to the *function* of the DN sub-component, it is inherently limited since the transcoder only has one hidden layer and so its functional geometry is not very expressive. In Figure 3, we see that the transcoder's functional geometry is not faithful to that of the DN sub-component, meaning it has not captured its underlying features.

Transcoders were primarily constructed to overcome the apparent poly-semanticity of neurons in the multi-layer perceptron components of transformers [51], and facilitate circuit discovery. For this, we can instead use (1) to effectively attribute the influence of a neuron to features, which directly accounts for this poly-semanticity since centroids are necessarily influenced by the behaviour of multiple neurons.

In our experiments, we consider the sensitivity of centroids to neuron pruning as a method for quantifying the influence between neurons and features. In Figure 4, we see that the neurons of the third hidden layer of the DN of Section 3 have a more encompassing effect on the entire boundary of the polygon, whereas, the neurons of the second layer are more directly linked to the external sectors of the polygon – as expected from our prior analyses.

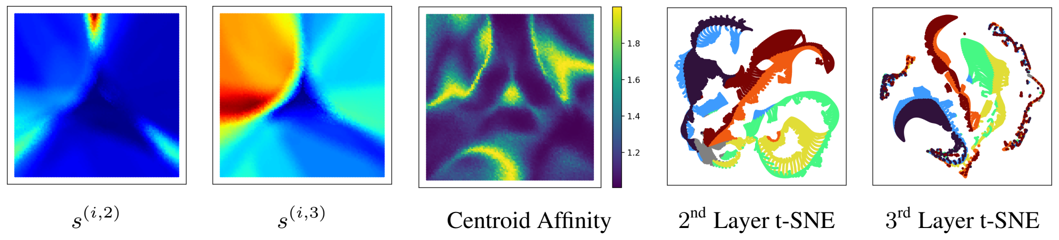
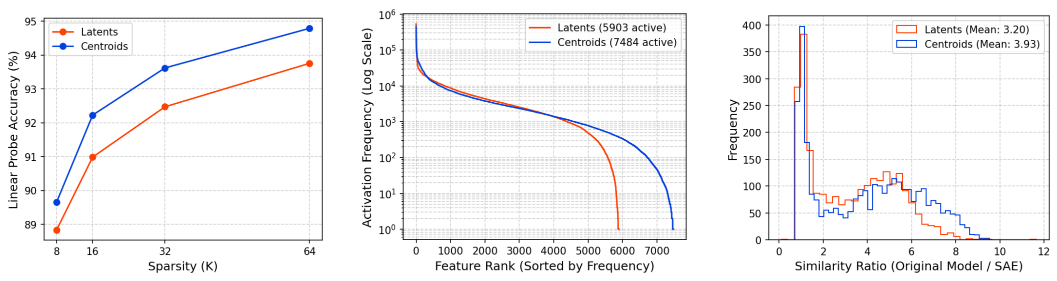


Figure 4: Here we study the centroids of the softened DN of Section 3. In the **far left** and **left** plot, we visualise the influence of neurons from the second and third hidden layer on the centroids of points sampled in the input space. In the **centre** plot, we took individual points in the input space, obtained a 128 sized sample of points within a 0.4 radius of this point, and computed the effective dimension of the corresponding centroids. In the **right** and **far right** plot, we took the input sample of Figure 9 and embedded – using t-SNE – their corresponding centroids obtained at the second and third hidden layer. Points within the polygon are coloured grey, whilst the other points are coloured depending on what sector of the input space they came from.

4.3 Point-Cloud Analysis

Another standard practice in interpretability is studying point clouds of latent activations or feature directions using embedding methods [52], topological descriptors [53] or linear probes [8, 10]. Although centroid affinity only posits centroids having an affine structure, from our simple example, we have seen that this structure is *coherent*. Namely, in Figure 1, the centroids possess an angular structure that is relevant to their structure in the input space.



(a) Linear Probe Accuracy (b) Feature Firing Distribution (c) Activation Pattern Similarity

Figure 5: For the **left** and **centre** plots, we train sparse autoencoders on the latent activations and centroids extracted from all the tokens of the DINOv2 feature extractor. More specifically, we train a TopK sparse autoencoder on the extracted vectors from the Imagenette train dataset with an expansion factor of 10 and sparsity values in the range $\{8, 16, 32, 64\}$. In the **left** plot, we measure the accuracy of the corresponding linear probe, and in the **centre** plot, we measure the firing distribution of the 32-K sparse autoencoder. In the **right** plot, we train a 32-K TopK sparse autoencoder with an expansion factor of 4 just on latent activations and centroids extracted from the class token of the DINOv2 feature extractor. We then record the activation similarity ratios for an input sample.

On the one hand, we can consider directly measuring the distribution of centroid affinity, using a notion of effective dimension,[8] to identify feature boundaries. In Figure 4, we see that this identifies the external sectors and interior of the polygon as features of the input space.

On the other hand, one can consider the embeddings of centroids to determine different clusters of features. Again, in Figure 4, we consider embedding the centroids at the second and last layer obtained from the samples of Figure 3. Before, we determined that the last layer of this DN captures the inside and outside of the polygon, whereas the previous layers coarsen this representation to the different sectors of the polygon. These observations are corroborated in the t-SNE embeddings [54] of Figure 4, where there is not much sector segmentation in the last layer centroids; whereas, the centroids of the second layer cluster more saliently depending on which sector of the input space the sample used to compute the centroid came from.

We reproduce these findings for DNs trained to classify polygons of other shapes in Appendix F and extend the analysis of centroid structure to a convolutional neural network trained on MNIST in Appendix G.

5 Experiments

We now explore some of these ideas on larger models and datasets, including DINOv2 [33] – a Vision Transformer with registers [55] backbone pre-trained on ImageNet [56] – on Imagenette [57], and GPT2 [34].

DINOv2 Feature Extraction with Sparse Autoencoders. Here we demonstrate that sparse autoencoders trained on the centroids from a DINOv2 with registers model yield a sparser, more meaningful and more functionally relevant dictionary of features. Similar to Hindupur et al. [58], we extract the features of Imagenette using this model, and train a TopK sparse autoencoder [59] on these features. However, we additionally consider training a TopK sparse autoencoder on the centroids extracted from the last multi-layer perceptron block of the model.

We compare the features of these sparse autoencoders in the following ways:[9] train linear probes on the decompositions of the train set of Imagenette to classify its classes and then evaluate the accuracy of the probe on the decompositions of the test set of Imagenette, record the frequency at which the features of the sparse autoencoder fire on the test set of Imagenette, record the activation pattern similarity ratios of an input sample. The activation pattern similarity ratio is computed as follows: we sample an input point and record the Jaccard similarity between its sparse decomposition and the

[8]Here we measure effective dimension using the exponential of the entropy of the normalised singular values.

[9]We additionally provide a more qualitative analysis in Appendix H.

sparse decomposition of the other points in the training distribution. We then compute the Jaccard similarity of its binarised latent activation in the feature extractor to the other points in the training distribution. The activation pattern similarity ratio is then this latter quantity divided by the former.

In Figure 5, we observe that the features from the centroid sparse autoencoders yield linear probes with higher accuracy and have a more uniform firing distribution across the feature dictionary. In particular, the activation pattern similarity ratios are generally larger, which means similar feature decompositions correspond to similar activation patterns in the feature extractor. Therefore, with Figure 5 we demonstrate that sparse autoencoders trained with centroids yield sparser, more general and functionally relevant features.

GPT2 Circuit Discovery. In Clement and Joseph [35], it was observed that in GPT2-Large, there is a neuron in the multi-layer perceptron component of the thirty-first layer, which is responsible for predicting the ”an” token. This was determined through performing ablations and observing the effect on the logits of the model. Through further manipulations, it was more generally concluded that this neuron works in concert with other neurons within the multi-layer perceptron block to capture the corresponding feature, which aligns with centroid affinity, where necessarily multiple neurons must contribute to construct regions of the input space which form the features of the DN. We support this by computing neuron attribution values with equation (1) using a neighbourhood of the last token embeddings at the input of the thirty-first layer on the prompt ”I climbed up the pear tree and picked a pear. I climbed up the apple tree and picked”.

In Figure 6, we observe that the neuron identified by Clement and Joseph [35], marked in black, has an attribution value within the 99.8^{th} percentile; with a select few other neurons also obtaining high attribution scores and the other neurons obtaining near-zero attribution scores.

6 Discussion

In this work, we have attempted to address the limitation of interpretability techniques derived under the LRH of not being aware of the *function* of the DN. To do so, we appealed to the spline theory of deep learning to explore a parametrisation of the functional geometry of a DN, where the functional geometry of a DN refers to the arrangement of the linear regions of its continuous piecewise affine approximation. We demonstrated that the affine structure of the centroids of this parametrisation identifies the features of a DN. Importantly, the emergence of this structure requires features to have linearly identifiable latent activations, which is not sufficient to also satisfy the LRH. Consequently, *centroid affinity* provides a perspective for identifying DN features that is less susceptible to identifying spurious or over-fitted features, something we demonstrated with experiments on a DINOv2 model. Importantly, to explore centroid affinity, we can continue to utilise interpretability techniques developed under the LRH; however, with novel centroid-based metrics, we can also interpret DNs such as GPT2.

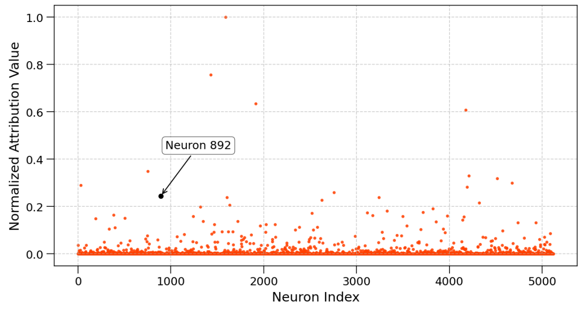


Figure 6: We prompt GPT2-Large to predict the ”an” token, using the prompt ”I climbed up the pear tree and picked a pear. I climbed up the apple tree and picked”. In a neighbourhood of the embedding at the input of the multi-layer perceptron block at the thirty-first layer of the last token of this prompt, we compute neuron attribution values for each neuron in the multi-layer perceptron using (1). The neighbourhood is constructed by sampling 256 points within a radius of 0.25 of the embedding. We normalise these values to be between zero and one. In black we indicate the 892^{nd} neuron in the multi-layer perceptron.

Although centroid affinity provides a principled and robust mechanism to extract the features of a DN, it has some limitations, and thus, we consider it a complementary perspective to the LRH. For example, features extracted using centroid affinity cannot be directly utilised for feature steering, unlike features extracted using the LRH. Moreover, exploring centroid affinity requires more computation compared to exploring the LRH; we quantify this for the Section 5 experiments in Appendix I.

Acknowledgments and Disclosure of Funding

This work was supported by ONR grant N00014-23-1-2714, ONR MURI N00014-20-1-2787, DOE grant DE-SC0020345, and DOI grant 140D0423C0076.

References

[1] Ahmed Imtiaz Humayun, Randall Balestriero, Guha Balakrishnan, and Richard Baraniuk. SplineCam: Exact Visualization and Characterization of Deep Network Geometry and Decision Boundaries. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, June 2023.

[2] A.R. Barron. Universal approximation bounds for superpositions of a sigmoidal function. *IEEE Transactions on Information Theory*, 39(3), 1993.

[3] Randall Balestriero and Richard Baraniuk. A Spline Theory of Deep Learning. In *Proceedings of the 35th International Conference on Machine Learning*. PMLR, July 2018.

[4] Randall Balestriero, Romain Cosentino, Behnaam Aazhang, and Richard Baraniuk. The Geometry of Deep Networks: Power Diagram Subdivision. In *Neural Information Processing Systems*, May 2019.

[5] Nelson Elhage, Tristan Hume, Catherine Olsson, Nicholas Schiefer, Tom Henighan, Shauna Kravec, Zac Hatfield-Dodds, Robert Lasenby, Dawn Drain, Carol Chen, Roger Grosse, Sam McCandlish, Jared Kaplan, Dario Amodei, Martin Wattenberg, and Christopher Olah. Toy models of superposition. *Transformer Circuits Thread*, 2022.

[6] Kiho Park, Yo Joong Choe, and Victor Veitch. The linear representation hypothesis and the geometry of large language models. In *Forty-First International Conference on Machine Learning*, 2024.

[7] Tomás Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. Efficient Estimation of Word Representations in Vector Space. In *1st International Conference on Learning Representations*, 2013.

[8] Neel Nanda, Andrew Lee, and Martin Wattenberg. Emergent linear representations in world models of self-supervised sequence models. In *Proceedings of the 6th BlackboxNLP Workshop: Analyzing and Interpreting Neural Networks for NLP*. Association for Computational Linguistics, December 2023.

[9] Marco Tulio Ribeiro, Sameer Singh, and Carlos Guestrin. "Why should I trust you?": Explaining the predictions of any classifier. In *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. Association for Computing Machinery, 2016.

[10] Been Kim, Martin Wattenberg, Justin Gilmer, Carrie Cai, James Wexler, Fernanda Viegas, and Rory sayres. Interpretability beyond feature attribution: Quantitative testing with concept activation vectors (TCAV). In *Proceedings of the 35th International Conference on Machine Learning*, July 2018.

[11] Zhi Chen, Yijie Bei, and Cynthia Rudin. Concept whitening for interpretable image recognition. *Nature Machine Intelligence*, 2(12), December 2020.

[12] Trenton Bricken, Adly Templeton, Joshua Batson, Brian Chen, Adam Jermyn, Tom Conerly, Nick Turner, Cem Anil, Carson Denison, Amnda Askell, Robert Lasenby, Yifan Wu, Shauna Kravec, Nicholas Schiefer, Tim Maxwell, Nicholas Joseph, Zac Dodds-Hatfield, Alex Tamkin, Karina Nguyen, Brayden McLean, Josiah E Burke, Tristan Hume, Shan Carter, Tom Henighan, and Christopher Olah. Towards Monosemanticity: Decomposing Language Models With Dictionary Learning. *Transformer Circuits Thread*, 2023.

[13] Robert Huben, Hoagy Cunningham, Logan Riggs Smith, Aidan Ewart, and Lee Sharkey. Sparse autoencoders find highly interpretable features in language models. In *The Twelfth International Conference on Learning Representations*, 2024.

[14] Tolga Bolukbasi, Kai-Wei Chang, James Zou, Venkatesh Saligrama, and Adam Kalai. Man is to computer programmer as woman is to homemaker? debiasing word embeddings. In *Proceedings of the 30th International Conference on Neural Information Processing Systems*, 2016.

[15] Nelson F. Liu, Matt Gardner, Yonatan Belinkov, Matthew E. Peters, and Noah A. Smith. Linguistic knowledge and transferability of contextual representations. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*. Association for Computational Linguistics, June 2019.

[16] John Hewitt and Christopher D. Manning. A structural probe for finding syntax in word representations. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*. Association for Computational Linguistics, June 2019.

[17] Adly Templeton, Tom Conerly, Jonathan Marcus, Jack Lindsey, Trenton Bricken, Brian Chen, Adam Pearce, Craig Citro, Emmanuel Ameisen, Andy Jones, Hoagy Cunningham, Nicholas L Turner, Callum McDougall, Monte MacDiarmid, C. Daniel Freeman, Theodore R. Sumers, Edward Rees, Joshua Batson, Adam Jermyn, Shan Carter, Chris Olah, and Tom Henighan. Scaling monosemanticity: Extracting interpretable features from claude 3 sonnet. *Transformer Circuits Thread*, 2024.

[18] Andy Arditi, Oscar Balcells Obeso, Aaquib Syed, Daniel Paleka, Nina Rimsky, Wes Gurnee, and Neel Nanda. Refusal in language models is mediated by a single direction. In *The Thirty-Eighth Annual Conference on Neural Information Processing Systems*, 2024.

[19] Yuping Lin, Pengfei He, Han Xu, Yue Xing, Makoto Yamada, Hui Liu, and Jiliang Tang. Towards understanding jailbreak attacks in LLMs: A representation space analysis. In *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing*. Association for Computational Linguistics, November 2024.

[20] Sanjeev Arora, Yuanzhi Li, Yingyu Liang, Tengyu Ma, and Andrej Risteski. A latent variable model approach to PMI-based word embeddings. *Transactions of the Association for Computational Linguistics*, 4, 2016.

[21] Kiho Park, Yo Joong Choe, Yibo Jiang, and Victor Veitch. The geometry of categorical and hierarchical concepts in large language models. In *The Thirteenth International Conference on Learning Representations*, 2025.

[22] Lee Sharkey, Bilal Chughtai, Joshua Batson, Jack Lindsey, Jeff Wu, Lucius Bushnaq, Nicholas Goldowsky-Dill, Stefan Heimersheim, Alejandro Ortega, Joseph Bloom, Stella Biderman, Adria Garriga-Alonso, Arthur Conmy, Neel Nanda, Jessica Rumbelow, Martin Wattenberg, Nandi Schoots, Joseph Miller, Eric J. Michaud, Stephen Casper, Max Tegmark, William Saunders, David Bau, Eric Todd, Atticus Geiger, Mor Geva, Jesse Hoogland, Daniel Murfet, and Tom McGrath. Open Problems in Mechanistic Interpretability, January 2025.

[23] Angus Nicolson, Lisa Schut, Alison Noble, and Yarin Gal. Explaining explainability: Recommendations for effective use of concept activation vectors. *Transactions on Machine Learning Research*, 2025.

[24] Gonçalo Paulo, Alex Mallen, Caden Juang, and Nora Belrose. Automatically Interpreting Millions of Features in Large Language Models, December 2024.

[25] Nikita Balagansky, Ian Maksimov, and Daniil Gavrilov. Mechanistic permutability: Match features across layers. In *The Thirteenth International Conference on Learning Representations*, 2025.

[26] Tom Lieberum, Senthooran Rajamanoharan, Arthur Conmy, Lewis Smith, Nicolas Sonnerat, Vikrant Varma, János Kramár, Anca Dragan, Rohin Shah, and Neel Nanda. Gemma Scope: Open Sparse Autoencoders Everywhere All At Once on Gemma 2, August 2024.

[27] Juhan Bae, Nathan Hoyen Ng, Alston Lo, Marzyeh Ghassemi, and Roger Baker Grosse. If influence functions are the answer, then what is the question? In *Advances in Neural Information Processing Systems*, 2022.

[28] Daniel Murfet, Susan Wei, Mingming Gong, Hui Li, Jesse Gell-Redman, and Thomas Quella. Deep Learning is Singular, and That's Good. *IEEE Transactions on Neural Networks and Learning Systems*, 34(12), December 2023.

[29] Roger Grosse, Juhan Bae, Cem Anil, Nelson Elhage, Alex Tamkin, Amirhossein Tajdini, Benoit Steiner, Dustin Li, Esin Durmus, Ethan Perez, Evan Hubinger, Kamilė Lukošiūtė, Karina Nguyen, Nicholas Joseph, Sam McCandlish, Jared Kaplan, and Samuel R. Bowman. Studying Large Language Model Generalization with Influence Functions, August 2023.

[30] Edmund Lau, Zach Furman, George Wang, Daniel Murfet, and Susan Wei. The Local Learning Coefficient: A Singularity-Aware Complexity Measure, September 2024.

[31] Jesse Hoogland, George Wang, Matthew Farrugia-Roberts, Liam Carroll, Susan Wei, and Daniel Murfet. The Developmental Landscape of In-Context Learning, February 2024.

[32] Sang Keun Choe, Hwijeen Ahn, Juhan Bae, Kewen Zhao, Minsoo Kang, Youngseog Chung, Adithya Pratapa, Willie Neiswanger, Emma Strubell, Teruko Mitamura, Jeff Schneider, Eduard Hovy, Roger Grosse, and Eric Xing. What is Your Data Worth to GPT? LLM-Scale Data Valuation with Influence Functions, May 2024.

[33] Maxime Oquab, Timothée Darcet, Théo Moutakanni, Huy V. Vo, Marc Szafraniec, Vasil Khalidov, Pierre Fernandez, Daniel HAZIZA, Francisco Massa, Alaaeldin El-Nouby, Mido Assran, Nicolas Ballas, Wojciech Galuba, Russell Howes, Po-Yao Huang, Shang-Wen Li, Ishan Misra, Michael Rabbat, Vasu Sharma, Gabriel Synnaeve, Hu Xu, Herve Jegou, Julien Mairal, Patrick Labatut, Armand Joulin, and Piotr Bojanowski. DINOv2: Learning robust visual features without supervision. *Transactions on Machine Learning Research*, 2024.

[34] Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, and Ilya Sutskever. Language models are unsupervised multitask learners. *OpenAI*, 2019.

[35] Neo Clement and Miller Joseph. We Found An Neuron in GPT-2. https://www.lesswrong.com/posts/cgqh99SHsCv3jJYDS/we-found-an-neuron-in-gpt-2, February 2023.

[36] Tom Lyche and Larry L Schumaker. Local spline approximation methods. *Journal of Approximation Theory*, 15(4):294–325, 1975.

[37] Larry Schumaker. *Spline Functions: Basic Theory*. Cambridge Mathematical Library. Cambridge University Press, Cambridge, 3 edition, 2007.

[38] Randall Balestriero, Romain Cosentino, and Sarath Shekkizhar. Characterizing Large Language Model Geometry Helps Solve Toxicity Detection and Generation. In *International Conference on Machine Learning*, December 2023.

[39] Romain Cosentino and Sarath Shekkizhar. Reasoning in Large Language Models: A Geometric Perspective, July 2024.

[40] Ahmed Imtiaz Humayun, Ibtihel Amara, Cristina Nader Vasconcelos, Deepak Ramachandran, Candice Schumann, Junfeng He, Katherine A Heller, Golnoosh Farnadi, Negar Rostamzadeh, and Mohammad Havaei. What secrets do your manifolds hold? Understanding the local geometry of generative models. In *The Thirteenth International Conference on Learning Representations*, 2025.

[41] Guido Montúfar, Razvan Pascanu, Kyunghyun Cho, and Yoshua Bengio. On the Number of Linear Regions of Deep Neural Networks. In *Neural Information Processing Systems*, February 2014.

[42] Ahmed Imtiaz Humayun, Randall Balestriero, and Richard Baraniuk. Deep Networks Always Grok and Here is Why. In *High-Dimensional Learning Dynamics 2024: The Emergence of Structure and Reasoning*, June 2024.

[43] Alethea Power, Yuri Burda, Harri Edwards, Igor Babuschkin, and Vedant Misra. Grokking: Generalization Beyond Overfitting on Small Algorithmic Datasets, January 2022.

[44] Neel Nanda, Lawrence Chan, Tom Lieberum, Jess Smith, and Jacob Steinhardt. Progress measures for grokking via mechanistic interpretability. In *The Eleventh International Conference on Learning Representations*, September 2022.

[45] Lewis Smith. The ‘strong’ feature hypothesis could be wrong, August 2024.

[46] Kevin Meng, David Bau, Alex J Andonian, and Yonatan Belinkov. Locating and editing factual associations in GPT. In Alice H. Oh, Alekh Agarwal, Danielle Belgrave, and Kyunghyun Cho, editors, *Advances in Neural Information Processing Systems*, 2022.

[47] Kevin Ro Wang, Alexandre Variengien, Arthur Conmy, Buck Shlegeris, and Jacob Steinhardt. Interpretability in the wild: A circuit for indirect object identification in GPT-2 small. In *The Eleventh International Conference on Learning Representations*, 2023.

[48] Nicholas Goldowsky-Dill, Chris MacLeod, Lucas Sato, and Aryaman Arora. Localizing model behavior with path patching, 2023.

[49] Jacob Dunefsky, Philippe Chlenski, and Neel Nanda. Transcoders find interpretable LLM feature circuits. In *The Thirty-Eighth Annual Conference on Neural Information Processing Systems*, 2024.

[50] Dan Hendrycks and Kevin Gimpel. Gaussian Error Linear Units (GELUs), June 2023.

[51] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Lukasz Kaiser, and Illia Polosukhin. Attention is All you Need. In *Advances in Neural Information Processing Systems*, 2017.

[52] Yuxiao Li, Eric J. Michaud, David D. Baek, Joshua Engels, Xiaoqing Sun, and Max Tegmark. The geometry of concepts: Sparse autoencoder feature structure. *Entropy. An International and Interdisciplinary Journal of Entropy and Information Studies*, 27(4), 2025.

[53] Aideen Fay, Inés García-Redondo, Qiquan Wang, Haim Dubossarsky, and Anthea Monod. Holes in latent space: Topological signatures under adversarial influence, 2025.

[54] Laurens van der Maaten and Geoffrey Hinton. Visualizing Data using t-SNE. *Journal of Machine Learning Research*, 9(86), 2008.

[55] Timothée Darcet, Maxime Oquab, Julien Mairal, and Piotr Bojanowski. Vision transformers need registers. In *The Twelfth International Conference on Learning Representations*, 2024.

[56] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. ImageNet Classification with Deep Convolutional Neural Networks. In *Advances in Neural Information Processing Systems*, 2012.

[57] Jeremy Howard. Fastai/imagenette. fast.ai, July 2025.

[58] Sai Sumedh R. Hindupur, Ekdeep Singh Lubana, Thomas Fel, and Demba E. Ba. Projecting assumptions: The duality between sparse autoencoders and concept geometry. In *ICML Workshop on Methods and Opportunities at Small Scale*, 2025.

[59] Leo Gao, Tom Dupre la Tour, Henk Tillman, Gabriel Goh, Rajan Troll, Alec Radford, Ilya Sutskever, Jan Leike, and Jeffrey Wu. Scaling and evaluating sparse autoencoders. In *The Thirteenth International Conference on Learning Representations*, 2025.

[60] Chris Olah, Nick Cammarata, Ludwig Schubert, Gabriel Goh, Michael Petrov, and Shan Carter. Zoom in: An introduction to circuits. *Distill*, 2020.

[61] Prajit Ramachandran, Barret Zoph, and Quoc V. Le. Searching for activation functions, 2017.

[62] Randall Balestriero and Richard G Baraniuk. From hard to soft: Understanding deep network nonlinearities via vector quantization and statistical inference. In *International Conference on Learning Representations*, 2018.

A Motivating Example

Throughout our discussion, we utilised a simple example to gain an intuition for how centroids identify the features of a DN. In Figure 7, we illustrate this example and indicate how the nonlinearities of the corresponding DN affect its functional geometry.

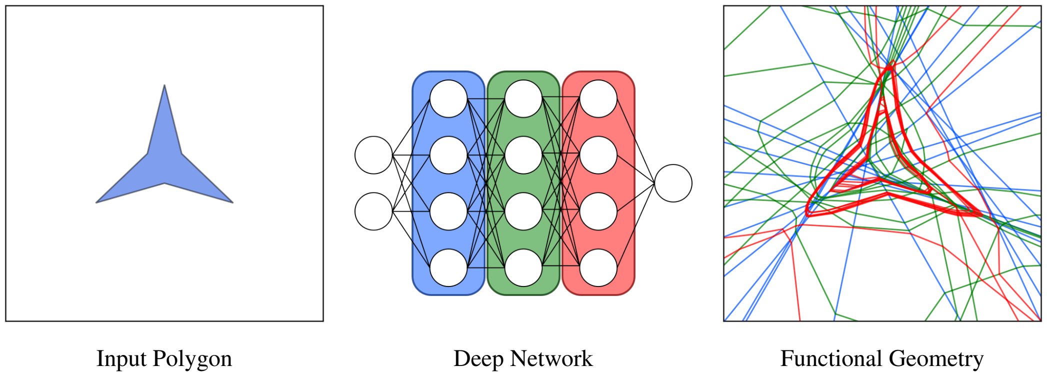


Figure 7: A DN has a functional geometry formed by its nonlinearities. Each nonlinearity identifies a plane within the input space of the DN. These planes intersect to bound regions which construct the features of the DN. On the **left**, we visualise the underlying data distribution that the DN is being trained on. In the **centre**, we visualise a simplified schematic of the architecture of the DN. In this schematic, we highlight the nonlinearities in the first, second and third hidden layers of the DN; each of which constructs a hyperplane within the input space of the DN, which we identify in the **right** plot. The **right** plot depicts the functional geometry of the DN (using SplineCam [1]), having trained on the polygon of the **left** plot.

B Comparison to Prior Work

Features. Definition 2.1 is analogous to the notion of a feature used in Park et al. [6], which provides a rigorous theoretical characterisation of the LRH. In particular, in Park et al. [6], a feature is a variable that leads to a particular output when caused by a context. In Definition 2.1, the concept would correspond to the region of the input space, the context would correspond to points within the region, and the particular output would be the induced functional behaviour.

Since an underlying assumption of the machine learning paradigm is that there exists an underlying distribution on the input space from which inputs are drawn, the probabilistic notion of a variable used in Park et al. [6] is comparable to the notion of a region used in Definition 2.1. More specifically, a region of the input space can be thought of as a variable whose probability distribution is derived from the underlying distribution of the input space.

Circuits. Circuits were initially introduced in Olah et al. [60] to deal with the apparent polysemantic nature of neurons. That is, specific neurons were observed to trigger on seemingly semantically disjoint inputs, whereas ensembles of neurons demonstrated more reliable activation patterns. Instead, our notion of a circuit arises naturally as the functional response to a feature. In particular, this functional response is likely to incorporate multiple neurons or components of a DN due to the DN's compositional construction.

C Formal Theory

Definition C.1. A feature of the ℓ^{th} layer of a DN is a collection of regions constructed by hyperplanes whose normals have a pair-wise cosine similarity bounded below by $1 - \epsilon$, and whose closest points to the origin have a pair-wise Euclidean distance bounded above by δ.

Proposition C.2. *The features in the ℓ^{th} layer of a DN are the collection of regions in its input space, $\mathbb{R}^{d^{(\ell-1)}}$, whose corresponding centroids form an affine subspace in $\mathbb{R}^{d^{(\ell-1)}}$ with deviations proportional to approximately $\sqrt{2\epsilon}$.*

Proof. Let the corresponding centroids and radii of the ℓth of the DN be $\left\{\left(\mu_{\boldsymbol{\nu}}^{(\ell)}, \tau_{\boldsymbol{\nu}}^{(\ell)}\right)\right\} \subseteq \mathbb{R}^{d^{(\ell-1)}} \times \mathbb{R}$. Then suppose Π and $\tilde{\Pi}$ are hyperplanes forming the feature. Each hyperplane, say Π, corresponds to a boundary of two regions, such that

$$\begin{aligned}\Pi &= \left\{\mathbf{x} : \left\|\mu_1^{(\ell)} - \mathbf{x}\right\|_2^2 - \tau_1^{(\ell)} = \left\|\mu_2^{(\ell)} - \mathbf{x}\right\|_2^2 - \tau_2^{(\ell)}\right\} \\ &= \left\{\mathbf{x} : \left\langle \mu_1^{(\ell)} - \mu_2^{(\ell)}, \mathbf{x}\right\rangle = c\right\},\end{aligned}$$

for some constant c. Thus, Π has normal vector $\mathbf{n}_1 := \mu_1^{(\ell)} - \mu_2^{(\ell)}$. Similarly, we can assume $\tilde{\Pi}$ is such that it has normal $\mathbf{n}_2 := \mu_2^{(\ell)} - \mu_3^{(\ell)}$. By assumption we have that

$$\frac{\mathbf{n}_1 \cdot \mathbf{n}_2}{\|\mathbf{n}_1\|_2 \|\mathbf{n}_2\|_2} \geq 1 - \epsilon.$$

Thus, by using small angle approximations, the angle between the normal vectors θ is approximately less than $\sqrt{2\epsilon}$. In particular, $\mathbf{n}_2$ can be decomposed into components parallel and orthogonal to $\mathbf{n}_1$ as

$$\mathbf{n}_2 = \|\mathbf{n}_2\|_2 \cos(\theta)\hat{\mathbf{n}}_1 + \|\mathbf{n}_2\|_2 \sin(\theta)\hat{\mathbf{u}},$$

where $\hat{\mathbf{n}}_1$ is the unit vector of $\mathbf{n}_1$ and $\hat{\mathbf{u}}$ is normal to it. Consequently, we can write

$$\begin{cases} \mu_1^{(\ell)} = \mu_1^{(\ell)} + 0 \cdot \mathbf{d} \\ \mu_2^{(\ell)} = \mu_1^{(\ell)} - \|\mathbf{n}_1\|_2 \mathbf{d} \\ \mu_3^{(\ell)} = \mu_2^{(\ell)} + \|\mathbf{n}_2\|_2 \cos(\theta)\mathbf{d} + \|\mathbf{n}_2\|_2 \sin(\theta)\hat{\mathbf{u}}, \end{cases}$$

where $\mathbf{d} = \hat{\mathbf{n}}_1$. Therefore, since $\sin(\theta)$ is of order $\sqrt{2\epsilon}$, the proof is complete. □

Definition C.3. A feature of a DN sub-component is a collection of regions in its input space formed by a feature in the ℓth layer.

Theorem C.4. *The features of a DN sub-component are the collections of regions in its input space whose centroids form approximate affine subspaces.*

Proof. Suppose that the feature corresponds to a (ϵ, δ)-feature of the ℓth layer. Then by Proposition C.2, the centroids at the ℓth form an approximate affine subspace. Thus, for sufficiently small δ, the DN sub-component corresponds to an affine transformer, meaning the corresponding centroids within the input space of the DN sub-component also form an approximate affine subspace. □

D Emergence of Centroid Structure

The centroid affinity of Figure 1 emerges gradually through training. At initialisation, the centroids have a similar arrangement to the input samples, due to the random initialisation of the DN. However, as training progresses, we observe that the centroids slowly migrate and align themselves. In particular, we can see the alignment of the centroids manifest before they arrive at their ultimate position.

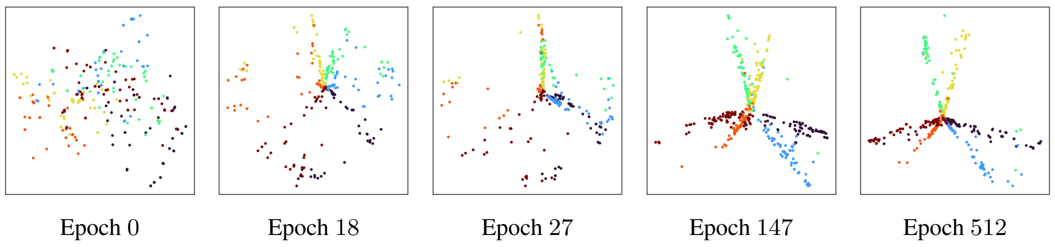


Figure 8: Throughout the training of the DN of Section 3, we tracked the DN centroids of the input samples of Figure 1.

E Softening Deep Networks

We developed centroid affinity by studying the level-sets of nonlinearities, which is a property of continuous piecewise affine DNs (i.e. those implementing continuous piecewise affine nonlinearities, like ReLU). We argued that this was valid since any DN can be approximated by such continuous piecewise affine DNs. However, for these DNs, the centroids are discrete objects since they exist uniquely for each linear region, which may present a challenge since Theorem 3.3 is a necessarily continuous utilisation of centroids. Therefore, here we consider the effect of using smooth nonlinearities on centroid affinity.

Firstly, to allow for better analyses of continuous piecewise affine DNs, we will explore the effect of relaxing their nonlinearities to smooth nonlinearities. For example, for the DN of Section 3, we consider *softening* it by replacing the ReLU nonlinearities with GELU nonlinearities [50]. The GELU nonlinearity belongs to the swish family of nonlinearities [61], which are theoretically known to provide an appropriate softening of a ReLU DN's functional geometry [62]. In Figure 9, we see that by softening the DN, we maintain and add more detail to the structure of the centroids.

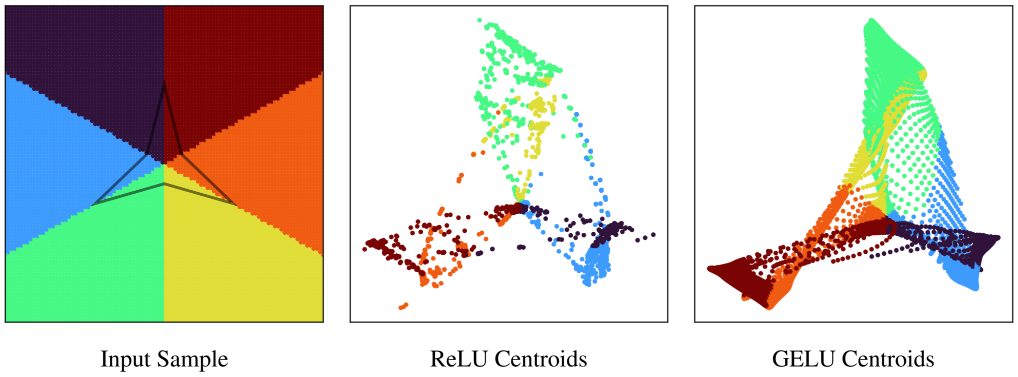


Figure 9: Softening a DN with ReLU nonlinearities by replacing them with GELU nonlinearities provides more detail to the structure of the centroids without affecting their overall structure. Here we sample a grid of points in the input space of the polygon-classifying DN of Section 3, **left** plot, and compute their corresponding centroids when the ReLU nonlinearity is maintained, **centre** plot, and when the ReLU nonlinearity is replaced by the GELU nonlinearity, **right** plot.

Secondly, we determine whether our investigations of Section 4 hold for DNs trained from scratch using continuous nonlinearities. That is, for a DN with GELU nonlinearities, we perform the exact same training procedure for the DN considered in Section 3, and then analyse the resulting centroids.

From Figure 10 we observe similar features as those identified for the ReLU DN considered in Section 4: when replacing the nonlinearity back to a ReLU we can observe its functional geometry using SplineCam [1] and we see the alignment of the nonlinearities around the polygon, when we observe the centroids of the input samples from Figure 9 we see the same affine structures that arose in the ReLU DN, computing centroid affinity for points in the input space again identifies the edges of the polygon as a feature, the influence of pruning neurons on the centroids is still effective as a neuron attribution metric.

F Other Polygons

In addition to the star-shaped polygon considered in the main text, in Figure 11 we corroborate the observed patterns when the input distribution is a bowtie-shaped and reuleaux-shaped polygon.

G MNIST Centroid Structure

Thus far, we have seen theoretically and in a simple example how the centroids of a DN have a semantically coherent structure; here, we demonstrate how this can be used to explore the *feature boundaries* of a DN trained on the MNIST classification task. For this, we train a DN with a

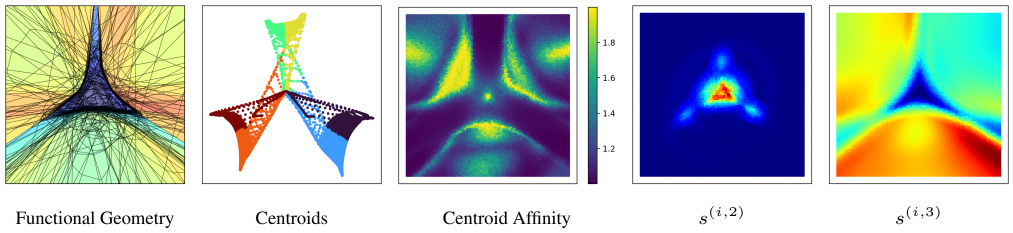


Figure 10: Here we train a DN in the same manner as the one considered in the main-text, except we use the GELU nonlinearity. In the **far left** plot, we replace the nonlinearities with ReLU such that we can use SplineCam to visualise its functional geometry. In the **left** plot, we visualise the centroids from the input samples of Figure 9. In the **centre** plot, we compute the centroid affinity of points in the input space based on a sample of radius 0.4. In the **right** and **far right** plots, we consider the sensitivities of centroids when pruning neurons from the second and third layers, respectively.

convolutional feature extractor followed by a linear layer on MNIST. After training, we sample two inputs from distinct classes and compute centroid affinity values – at the feature extractor component of the DN – along the linear interpolation between the samples. We observe in Figure 12 that there is a greater relative drop in centroid affinity between more distinct classes. More specifically, the 3 and 6 classes are intuitively more distinct than the 4 and 9 classes; consequently, centroid affinity is lower along the interpolation between the 3 and 6 inputs since the features are more distinct. Whereas, if we similarly consider the effective dimensions of the latent activations, we do not observe any contextual change.

H Qualitative Analysis of Features

In Section 5, we demonstrated quantitatively that the features extracted from a sparse autoencoder trained to reconstruct centroids were semantically- and functionally-relevant to the input distribution and feature extractor. Here, we qualitatively support this and compare them to the features extracted by the sparse autoencoder trained to reconstruct latent activations. To do so, we randomly sample a point from the input distribution and identify the other inputs from the distribution with similar feature decompositions – as measured by Jaccard similarity.

In Figure 13, we see that the sparse autoencoder trained to reconstruct centroids identifies similar features to those of the sparse autoencoder trained to reconstruct latent activations. This further supports that the centroids of a DN have a coherent structure that can be used to identify the features of the DN.

I Computational Requirements.

A valid concern with exploring centroid affinity is the computational burden it introduces into the process of interpretability, since it requires interrogating the Jacobians of a DN. Fortunately, though, this interrogation only requires considering Jacobian vector products (see Proposition 2.3), which are significantly cheaper to compute in common computational frameworks.

In this section, we empirically quantify the computational burden incurred by considering centroids rather than latents in our main experiments of Section 5.

DINO Feature Extraction. For this experiment, we compare the difference in computational time to extract the latent activations and centroids from the feature model. After these vectors are extracted, the computational pipeline is identical when using centroids or latent activations. We summarise the results in Table 1a. We see that extracting the centroid of the class token is only marginally slower than just extracting the latent activation, and that extracting the centroids of all the tokens only increases computation by 8%.

GPT2 Circuit Discovery. Although there is no direct analogue of this experiment with latent activations, we can still argue that the computational burden is relatively benign. In particular, since we only compute centroids across the multi-layer perceptron block of the thirty-first layer, we only need to consider the Jacobian vector product for this component. This can be done by storing gradients of a forward pass across this block, which, in relation to performing a forward pass across the model, is insignificant.

MNIST Centroid Structure. For this experiment, we compare the time necessary to perform the experiment with centroids or latent activations. More specifically, instead of considering the effective dimensions of the centroids of neighbourhoods of points, we compute the effective dimensions of the latent activations of neighbourhoods of points. We summarise the results in Table 1b, where we see that using centroids instead of latent activations requires 21% more computation time.

Table 1: In this table, we compare the computation times (in seconds) for using the centroids of a DN to perform interpretability to using latent activations.

(a) DINO Feature Extraction. Here we record the time taken (in seconds) to extract the centroids or latent activations from a DINOv2 model on the training distribution of Imagenette.

Experiment	With Centroids	With Latent Activations
Figures 5a and 5b	685	684
Figure 5c	727	673

(b) MNIST Centroid Structure.

Experiment	With Centroids	With Latent Activations
Figure 12	323	266

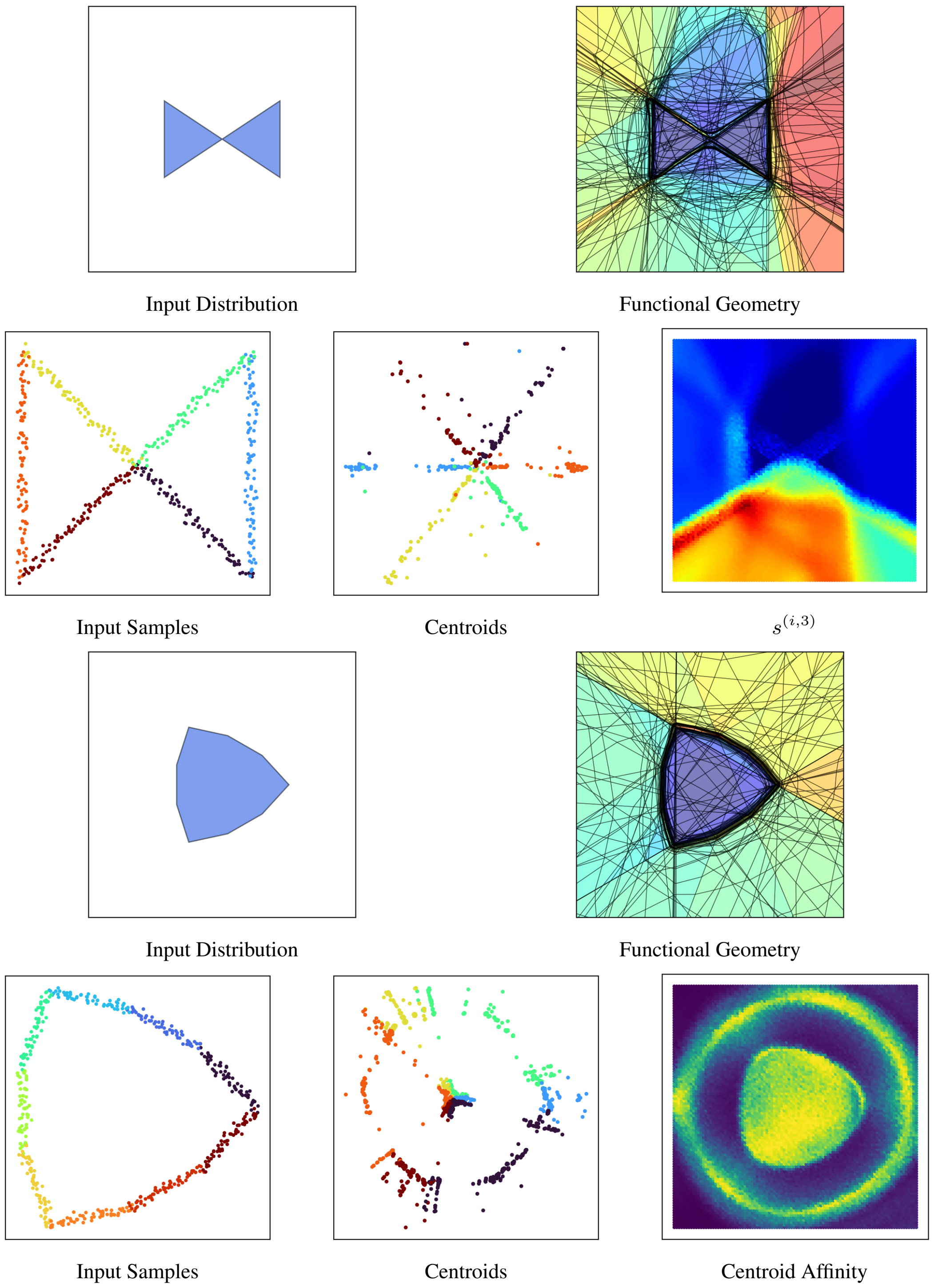
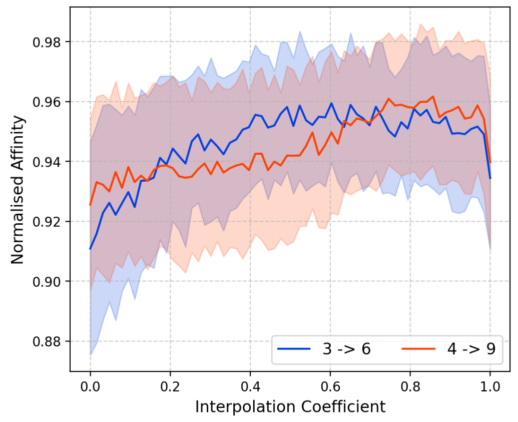


Figure 11: Here we perform some of the same analyses of Section 4 but with a DN trained on a bowtie-shape polygon, **top** two rows, and a reuleaux-shaped polygon, **bottom** two rows.



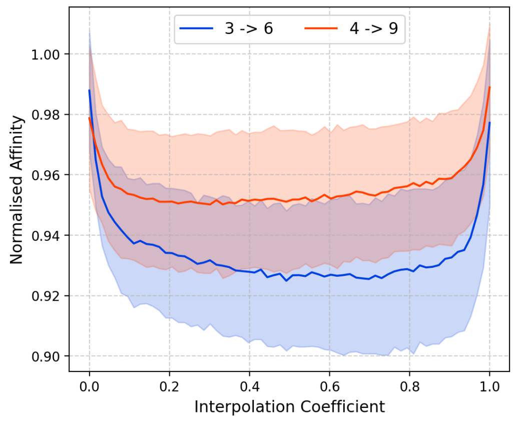
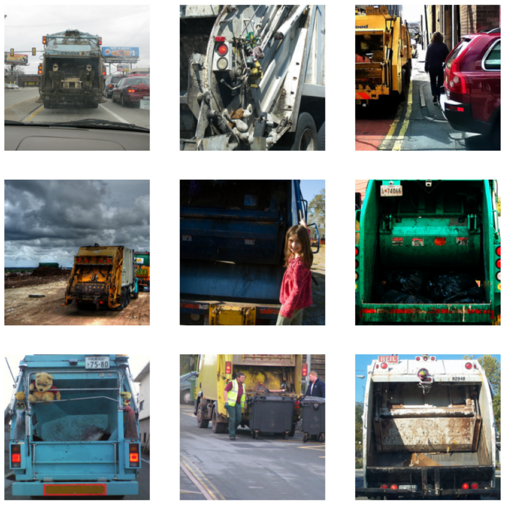
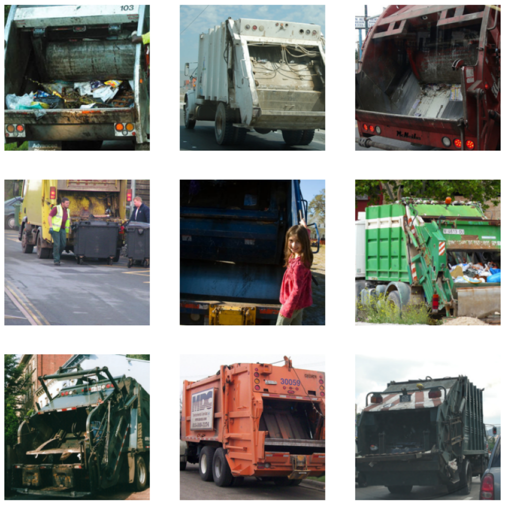
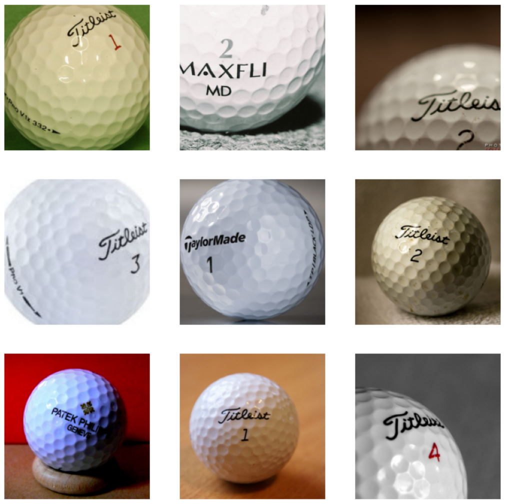


Figure 12: We train a DN on the MNIST classification. In the **right** plot, we consider the centroid affinity of centroids at the feature extractor component of the DN – which constitutes three convolutional layers. In the **left** plot, we similarly consider the effective dimensions of the latent activations. For two training points of distinct classes, we compute the centroid affinity of samples along their linear interpolation. In Figure 12 we visualise these affinities for samples from class pairs (3,6) and (4,9).


Centroid Garbage-truck Feature


Latent Garbage-truck Feature


Centroid Golf-ball Feature

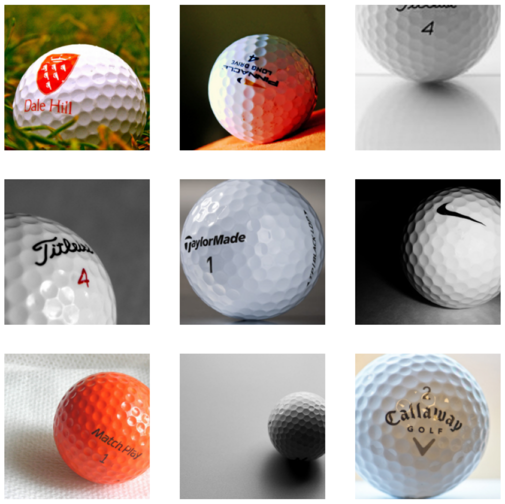

Latent Golf-ball Feature

Figure 13: For a sampled input, we compute its feature decomposition using the sparse autoencoders of Figure 5c, and then identify the other inputs whose feature decompositions are most similar to this using Jaccard similarity. More specifically, in the **left** column we consider the sparse autoencoder of Figure 5c trained using centroids, and in the **right** column we consider the sparse autoencoder of Figure 5c trained using latent activations. The central image of each plot represents the initial point that is sampled from the input distribution, and the surrounding images are the identified inputs with similar feature decompositions. Each **row** considers a specific input.