TopoAct: Visually Exploring the Shape of Activations in Deep Learning

Archit Rathore[1], Nithin Chalapathi[1], Sourabh Palande[1], Bei Wang[1†]

[1] School of Computing, Scientific Computing and Imaging (SCI) Institute, University of Utah, USA

Abstract

Deep neural networks such as GoogLeNet and ResNet have achieved impressive performance in tasks like image classification. To understand how such performance is achieved, we can probe a trained deep neural network by studying neuron activations, i. e., combinations of neuron firings, at any layer of the network in response to a particular input. With a large number of inputs, we aim to obtain a global view of what neurons detect by studying their activations. We ask the following questions: What is the shape of the activation space? What is the organizational principle behind neuron activations? How are the activations related within a layer and across layers? Applying tools from topological data analysis, we present ***TopoAct****, a visual exploration system to study topological summaries of activation vectors for a single layer as well as the evolution of such summaries across multiple layers. We present exploration scenarios using* ***TopoAct*** *that provide valuable insights towards learned representations of an image classifier. We expect* ***TopoAct*** *to give a topological perspective that enriches the current toolbox of neural network analysis, and to provide a basis for network architecture diagnosis and data anomaly detection.*

1. Introduction

Deep convolutional neural networks (CNNs) have become ubiquitous in image classification tasks thanks to architectures such as GoogLeNet and ResNet. However, we do not quite understand how these networks achieve their impressive performance. A main challenges in deep learning is the *interpretability*: How can we make the representations learned by these networks human interpretable?

Given a trained deep neural network, we can address the interpretability issue by probing neuron activations, *i. e.*, the combinations of neuron firings, in response to a particular input image. With millions of input images, we can obtain a global view of what the neurons have learned by studying neuron activations at a particular layer and across multiple layers. We aim to address the following questions: What is the shape of the activation space? What is the organizational principle behind neuron activations? And how are the activations related within a layer and across layers?

We propose to leverage tools from topological data analysis (TDA) to capture global and local patterns of how a trained network reacts to a large number of input images. In this work:

- We present **TopoAct**, an interactive visual analytics system that uses topological summaries to explore the space of activations in deep learning classifiers for a fixed layer and across multiple layers of the network. **TopoAct** leverages the mapper construction [SMC07] from TDA to capture the overall shape of activation vectors for interactive exploration.
- We present exploration scenarios where **TopoAct** helps us discover valuable, sometimes surprising, insights into learned representations of image classifiers such as InceptionV1 [SLJ*15] and ResNets [HZRS16].
- We observe structures in the topological summaries, specifically branches and loops, which correspond to evolving activation patterns that help us understand how a neural network reacts to a large group of images. In particular, we find a correlation between semantically meaningful distinctions and topological separations among images from different classes.
- Finally, we release an open-source, web-based implementation of the exploration interface on Github[†]; the current system is also available via a public demo link [‡].

We expect **TopoAct** to benefit the analysis and visualization of neural networks by providing researchers and practitioners the ability to probe black box neural networks from a novel topological perspective.

- To the best of our knowledge, **TopoAct** is the *first* tool that focuses on exploring complex topological structures – branches and loops – within the space of activation vectors. Its exploratory

† E-mail: beiwang@sci.utah.edu. This work was partially funded by NSF DBI-1661375, NSF IIS-1513616, and NSF IIS-1910733.

† https://github.com/architrathore/TopoAct-v2.1/

‡ https://architrathore.github.io/TopoAct-v2.1/

nature helps to inform the global and local organizational principles of activation vectors across different scales.
- **TopoAct** detects if and when activations from different classes become separated, via branches at a fixed layers and across multiple layers (see Sections 6, and in particular, the deer-horse bifurcation in Section 7), which may be used to inform diagnostic or corrective actions such as selective data augmentation for misclassified inputs or network layer modification for increasing the separation between confounding classes (see Section 8).

2. Related Work

We review visual analytics systems for deep learning interpretability as well as various notions of topological summaries.

Visual analytics systems. Visual analytics systems have been used to support model explanation, interpretation, debugging, and improvement for deep learning in recent years, see [HKPC18] for a survey. Here we focus on approaches based on neuron activations for interpretability in deep learning. This line of research attempts to explain the internal operations and the. behavior of deep neural networks by visualizing the features learned by hidden units of the network. Erhan *et al.* proposed *activation maximization* [EBCV09], which uses gradient ascent to find the input image that maximizes the activation of the neuron under investigation. It was used to visualize the hidden layers of a deep belief network [EBCV09] and deep auto-encoders [Le13]. Simonyan *et al.* [SVZ14] used a similar gradient-based approach to obtain salience maps by projecting neuron activations from the fully connected layers of the CNN back on to the input space. Building on the idea of activation maximization, Zieler *et al.* [ZF14] proposed a deconvolutional network that reconstructs the input of convolutional layers of a CNN from its output. Yosinski *et al.* [YCN*15] introduced the DeepVis framework that visualizes the live activations produced on each layer of a CNN as it processes images/videos. Their framework also enabled visualizing features at each layer via regularized optimization.

These methods assumed that each neuron specializes in learning one specific type of feature. However, the same neuron can be activated in response to vastly different types of input images. Reconstructing a single feature visualization, in such cases, leads to unintelligible mix of color, scales or parts of objects. To address this issue, Nguyen *et al.* [NYC16] proposed *multifaceted feature visualization* which synthesizes a visualization of each type of input image that activates a neuron. Another issue with these visualization approaches is the assumption that neurons operate in isolation. This issue is addressed by the *model inversion* method proposed by Mahendran *et al.* [MV15, MV16]. Model inversion looks at the representations learned by the fully connected layers of a CNN, and reconstructs the input from these representations. Kim *et al.* introduced the TCAV (Testing with Concept Activation Vectors) framework, which used directional derivatives of activations to quantify the sensitivity of model predictions to an underlying high-level concept [KWG*18]. While all these techniques can help us understand how a single input or a single class of inputs is “seen" by the network, visualizing activations of neurons alone is somewhat limited in explaining the global behavior of the network. To obtain a global picture of the network, Karpathy [Kar14] used t-SNE to arrange input images that have similar CNN codes (i.e., fc7 CNN features) nearby in the embedding. Nguyen *et al.* [NYC16] projected the training set images that maximally activate a neuron into a low-dimensional space, also via t-SNE. They clustered the images using k-means in the embedded space, and computed a mean image by averaging the images nearest to the cluster centroid.

Carter *et al.* recently proposed the *activation atlas* [CAS*19], which combines feature visualization with dimensionality reduction (DR) to visualize averaged activation vectors with respect to millions of input images. For a fixed layer, the activation atlas obtains a high-dimensional activation vector corresponding to each input image. These high-dimensional vectors are then projected onto low-dimensional space via UMAP [MHM18, MHSG18] or t-SNE [MH08]. Finally, feature visualization is applied to averaged activation vectors from small patches of the low-dimensional embedding which allow users to intuitively understand how a particular layer reacts to millions of input images. Hohman *et al.* proposed *SUMMIT* [HPRC19], another framework that summarizes neuron activations of an entire layer of a deep CNN using DR. In addition to aggregated activations, *SUMMIT* also computes neuron influences to construct an *attribution graph* which captures relationships between neurons across layers.

Activation atlas computes average activation vectors in a low-dimensional embedding which may introduce errors due to neighborhood distortions. In comparison, our approach aggregates activation vectors in a different manner. Using the *mapper* construction, a tool from TDA, we obtain a topological summary of a particular layer by preserving the clusters as well as relationships between the clusters in the original high-dimensional activation space. Our approach preserves more neighborhood structures since the topological summary is obtained within the high-dimensional activation space. Not only do we study how a particular layer of the neural network reacts to a large number of images through the lens of this topological summary, but also how such summaries evolve across layers.

Various notions of topological summaries. In TDA, various notions of topological summaries have been proposed to understand and characterize the structure of a scalar function $f : \mathbb{X} \to \mathbb{R}$ defined on some topological space $\mathbb{X}$. Some of these, such as merge trees [BYM*14], contour trees [CSA03], and Reeb graphs [Ree46], capture the behavior of the (sub)level sets of a function. Fewer topological summaries are applicable for a vector-valued function, including Jacobi sets [EH02, BWN*15], Reeb spaces [EHP08, MW16] and their discrete variant, the mapper construction [SMC07]. In this paper, we apply the mapper construction to the study of the space of activations to generate topological summaries suitable for interactive visualization. The mapper construction introduced by Singh *et al.* [SMC07] has seen widespread applications in data science, including cancer research [NLC11, MNL*19], sports analytics [Ala12], gene expression analysis [JCR*19], micro-epidemiology [KGCFR*20], genomic profiling [CZJ*19], and neuroscience [GSPS19, SSGC*18], to name a few; see [PVP17] for an overview. In visualization, topological approaches such as persistent homology and mapper have been recently applied in graph visualization [HWR18, HWSR18, SHW*20].

3. Comparison With t-SNE and UMAP

Various dimensionality reduction (DR) techniques have been proposed to analyze and visualize high-dimensional point cloud data [Cay08, LMW*17]. Among these, t-SNE [MH08] and UMAP [MHM18] are most relevant to our proposed work as they have been used previously in exploring neuron activations [Kar14, NYC16, CAS*19]. In particular, Carter *et al.* [CAS*19] employ both t-SNE and UMAP to project high-dimensional activation vectors to low-dimensions for visual exploration. In comparison with t-SNE and UMAP, the benefit of **TopoAct** is two-fold.

- **TopoAct** provides a global, graph-based representation of the space of activations, which explicitly summarizes the organizational principles (clusters and cluster relations) behind neuron activations. t-SNE and UMAP detect structures visible in the *low-dimensional* embedding, while **TopoAct** captures complex topological structures in the original *high-dimensional* space.
- While t-SNE and UMAP focus on preserving proximities within local neighborhoods, **TopoAct** explicitly reveals complex topological structures – branches and loops – that are not necessarily visible via t-SNE/UMAP. These topological structures can be used to guide refined, local structural analysis (Section 6).

The kNN graph representation constructed by UMAP can be considered as a topological representation of the high-dimensional data [MHM18], however, it only approximately preserves the *connectivity* among points within local patches of the manifold. While t-SNE and UMAPP are often able to recover well-separated clusters [MH08], neither of them explicitly preserve and capture complex topological structures such as loops or branches in the high-dimensional space. **TopoAct** addresses this important challenge by utilizing the mapper construction.

In addition, several DR techniques have been developed recently [SMVJ09,WSPVJ11,YZR*18] that explicitly preserve loops and branches; however, none of them give a global summary of all such structures in a single visualization, in comparison with the mapper construction. In studying the shape of the space of images, Lee *et al.* [LPM03], Carlsson *et al.* [CISZ08], and Xia [Xia16] studied the global topological structures of patches from natural images; they used different topological tools (such as persistence homology) and focused on different data problems (such as studying the global structure of natural images from a database) in comparison to **TopoAct**. Finally, a few recent works applied topological techniques in the study of activations. Gebhart *et al.* [GSH19] computed persistent homology over the activation structure of neural networks. In particular, they characterized the topological structure of the neural network architecture when viewed as a graph with edge weights provided by activations. Gabella *et al.* [GAES19] used both persistent homology and the mapper construct to study the parameter space of neural networks (i.e., parameter matrices) during training.

4. Technical Background

We review technical background on the mapper construction, and neural network architecture. We delay the discussions on activation vectors and feature visualization until Section 5.

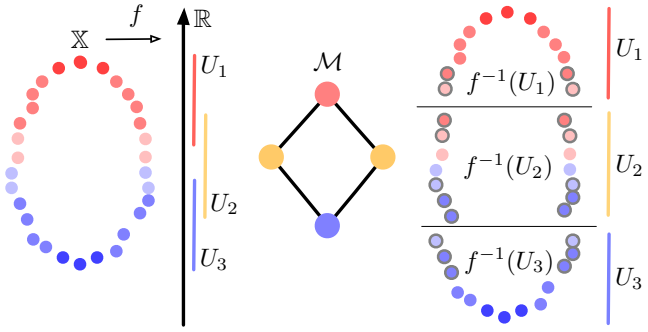

Figure 1: *A simple example of a mapper graph on a point cloud.*

Mapper graph on point cloud data. We give a high-level description of the framework by Singh *et al.* [SMC07] in a point cloud setting. Given a high-dimensional point cloud $\mathbb{X} \subset \mathbb{R}^d$ equipped with a function f on $\mathbb{X}$, $f : \mathbb{X} \to \mathbb{R}$, the mapper construction provides a topological summary of the data for compact representation and exploration. It utilizes the topological concept known as the *nerve of a covering* first introduced by Pavel Alexandrov [Ale28].

An *open cover* of $\mathbb{X}$ is a collection $\mathcal{U} = \{U_i\}_{i\in I}$ of open sets in $\mathbb{R}^d$ with an index set I such that $\mathbb{X} \subset \bigcup_{i\in I} U_i$. Given a cover $\mathcal{U}$ of $\mathbb{X}$, the 1-dimensional *nerve* of $\mathcal{U}$, denoted as $\mathcal{N}_1(\mathcal{U})$, is constructed as follows: A finite set $\{i, j\} \subset I$ (i.e., an edge) belongs to $\mathcal{N}_1(\mathcal{U})$ if and only if the intersection of U_i and U_j is non-empty; if the set $\{i, j\}$ belongs to $\mathcal{N}_1(\mathcal{U})$, then any of its subsets (i.e., the point i and the point j) is also in $\mathcal{N}_1(\mathcal{U})$.

For the mapper construction, we start with a finite cover $\mathcal{U} = \{U_i\}_{i\in I}$ of the image $f(\mathbb{X}) \subset \mathbb{R}$ of f, such that $f(\mathbb{X}) \subseteq \bigcup_i U_I$. Since f is a scalar function, U_i is an open interval in $\mathbb{R}$. Let $f^*(\mathcal{U})$ denote the cover of $\mathbb{X}$ obtained by considering the clusters induced by points in $f^{-1}(U_i)$ for each i. The 1-dimensional nerve of $f^*(\mathcal{U})$, denoted as $\mathcal{M} := \mathcal{N}_1(f^*(\mathcal{U}))$, is called the *mapper graph*. $\mathcal{M}$ is a multi-scale representation that serves as a topological summary of the point cloud $\mathbb{X}$. Its construction relies on three parameters: the function f, the cover $\mathcal{U}$, and the clustering algorithm.

The function f plays the role of a *lens*, through which we look at the data, and different lenses provide different insights [BGSF08, SMC07]. An interesting open problem for the mapper construction is how to define topological lenses beyond best practices or a rule of thumb [BGSF08, BMMP03]. In practice, functions such as height, distance from the barycenter, surface curvature, integral geodesic distances, and geodesic distances from a source point, have all been used as lenses [BGSF08]. In this paper, we use the $L2$ norm of the activation vectors as the lens; although other options are possible (see supplementary material for a detailed analysis of $L2$-norm as lens function).

The cover $\mathcal{U}$ of $f(\mathbb{X})$ consists of a finite number of open intervals as cover elements, $\mathcal{U} = \{U_i\}_{i\in I}$. A common strategy is to use uniformly sized overlapping intervals. Let n be the number of intervals and p the amount of overlap between adjacent intervals. Adjusting these parameters increases or decreases the amount of aggregation $\mathcal{M}$ provides.

We compute the clustering of the points lying within $f^{-1}(U_i)$

and connect the clusters whenever they have non-empty intersection. A typical algorithm to use is DBSCAN [EKSX96], a density-based clustering algorithm; it groups points in high-density regions together and makes points that lie alone in low-density regions outliers. The algorithm requires two input parameters: *minPts* (the number of samples in a neighborhood for a point to be considered as a core point), and ε (the maximum distance between two samples for one to be considered in the neighborhood of the other).

Figure 1 illustrates a mapper graph construction for a dataset $\mathbb{X}$ sampled from a noisy circle. The function (lens) used is $f(x) = ||x-p||_2$, where p is the lowest point in the data. X is colored by the value of the function. We divide the range of the f into 3 intervals $\{U_1, U_2, U_3\}$ with a 30% overlap. For each interval, we compute the clustering of the points lying within the domain of the lens function restricted to the interval $f^{-1}(U_i)$, and connect the clusters whenever they have nonempty intersection. $\mathcal{M}$ is the mapper graph whose vertices are colored by the index set, and it preserves the shape of the point cloud data – a global loop/circle.

InceptionV1 architecture. We give a high-level overview of InceptionV1 [SLJ*15] (GoogLeNet), the neural network architecture employed in this paper. However, our framework is not restricted by the specific architecture of a neural network. InceptionV1 is a CNN that won the ImageNet Large-Scale Visual Recognition Challenge for image classification in 2014. It was trained on ImageNet ILSVRC [DDS*09]. ImageNet consists of over 15 millions labeled high-resolution images with roughly 22,000 classes/categories while ILSVRC takes a subset of ImageNet of around 1000 images in each of 1000 classes, for a total of 1.2 million training images, 50,000 validation images, and 100,000 testing images. The highlights of InceptionV1 architecture include the use of 1×1 convolution, inception modules, and global average pooling. The 1×1 convolution from NIN (networks in networks) [LCY14] is used to reduce dimensionality (and computation) prior to expensive convolutions with larger image patches. A new level of organization is introduced in the form of the *inception module*, which combines different types of convolutions for the same input and stacking all the outputs on top of each other. InceptionV1 contains 9 *inception modules*, each composed of multiple convolution layers.

The demo version of **TopoAct** explores the activations of the last layer of each inception module. The module names such as *mixed3a*, *mixed3b* are shortened as *3a*, *3b*, etc. This choice is well-aligned with previous literature on visual exploration of InceptionV1 [OMS17, OSJ*18, HPRC19, CAS*19].

ResNet architecture. To demonstrate the generality of topological structures observed across different neural network architectures, we also apply **TopoAct** to activation vectors from a ResNet. Residual Networks or ResNets [HZRS16] were one of first neural network architectures that enabled training extremely deep neural networks with up to 1000 layers. A neural network N_D of depth D is a subnetwork of any network N_{D+K} of depth $D+k, k>0$. Theoretically, N_{D+K} should be capable of learning any function that N_D can learn by setting the k layers to an identity mapping, and thus have performance at least as good as the smaller network. In practice, however, increasing layers beyond a certain depth leads to a sharp degradation in performance (higher training error and lower classification accuracy on the test set) even when normalization schemes are used for both initialization and intermediate representations. ResNets overcome this problem by adding “shortcut" connections to a layer that adds the output from layer k to the input of layer $k+i$ where i is usually 2.

In our experiments, we have implemented ResNet-18, a residual network with 18 layers following the layer specifications in [HZRS16]. With 200 training epochs, it achieves a classification accuracy of 93.24% on CIFAR-10 and 91.87% on CIFAR-100 datasets [KH09].

5. Methods

We describe data analytic components of **TopoAct**. First, for a chosen layer of a neural network model (such as InceptionV1), we obtain activation vectors as high-dimensional point clouds for topological data analysis. Second, we construct mapper graphs from these point clouds to support interactive exploration. The nodes in the mapper graphs correspond to clusters of activation vectors in high-dimensional space and the edges capture relationships between these clusters. Third, for each node (cluster) in the mapper graph, we apply feature visualization to individual activation vectors in the cluster as well as the averaged activation vector.

Obtaining activation vectors as point clouds. The activation of a neuron is a non-linear transformation (i.e., a function) of its input. To start, we fix a pre-trained model (i.e., InceptionV1) and a particular layer (e.g., *4c*) of interest. We feed each input image to the network and collect the activations, *i. e.*, the numerical values of how much each neuron has fired with respect to the input, at a chosen layer. Since InceptionV1 is a CNN, a single neuron does not produce a single activation for an input image, but instead a collection of activations from a number of overlapping spatial patches of the image. When an entire image is passed through the network, a neuron will be evaluated multiple times, once for each patch of the image. For example, a neuron within layer *4c* outputs 14×14 activations per image (for 14×14 patches). To simplify the construction, in our setting, we randomly sample a single spatial activation from the 14×14 patches, excluding the edges to prevent boundary effects. For 300,000 images, this gives us 300,000 activation vectors for a given layer. Figure 2 illustrates what we mean by an activation vector. The dimension of an activation vector depends on the number of neurons in the layer, for instance, layers *3a*, *3b*, and *4a* have 256, 480, and 512 neurons respectively, producing point clouds of corresponding dimensions.

Constructing mapper graphs from activation vectors. Given a point cloud of activation vectors, we now compute a mapper graph as its topological summary. Each node in the mapper graph represents a cluster of activation vectors, and there is an edge between two nodes if their corresponding clusters have a nonempty intersection. All of our mapper graphs use $L2$-norm of the activation vector as the lens function; $n = 70$ cover elements with $p = 20\%, 30\%$, or 50% amount of overlap. We use DBSCAN [EKSX96] as the clustering algorithm, with minimum points per cluster $minPts = 5$. For the ε parameter of the DBSCAN algorithm, which defines core points, we use two variations in our experiments. In the first variation, we use a fixed $\varepsilon = 1,000$ estimated using the distribu-

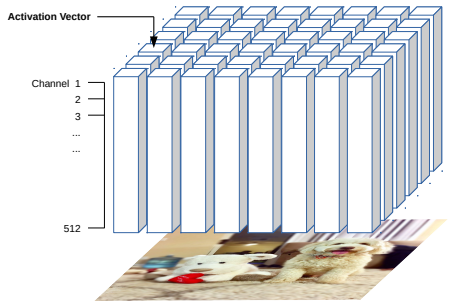

Figure 2: *Each activation vector is a vector of spatial activations across all channels.*

tion of pairwise distances at a middle layer. In the second variation, we set ε adaptively for each layer, employing the procedure proposed in [EKSX96]. Specifically, we generate an approximate kNN graph, sort the distances to the 5-th neighbor, and select an ε based on the location of a "valley" when these distances are plotted [EKSX96]. This way, the ε value is more adaptive to the activation space of each layer. As a result, our *adaptive* ε values are 830 for *3a*, 1070 for *3b*, 1750 for *4a*, 1630 for *4b*, 1330 for *4c*, 980 for *4d*, 775 for *4e*, 790 for *5a*, and 260 for *5b*.

These parameter configurations give rise to 6 datasets currently deployed in our live demo. Each dataset contains 9 mapper graphs (across 9 layers of InceptionV1) constructed by a particular set of parameters associated with the mapper construction. It is named according to these parameters. In particular, each name starts with "overlap-x" where x is either 20, 30, or 50 to denote 20%, 30%, or 50% overlap respectively. The second half of the name consists of "epsilon-x" where x is either "fixed" or "adaptive", indicating whether ε was fixed ($= 1000$) or set adaptively for each layer. For example, *overlap-50-epsilon-fixed* is the dataset containing mapper graphs of 9 layers generated using $p = 50\%$ with fixed $\varepsilon = 1000$.

Applying feature visualization to activation vectors. Activation vectors are high-dimensional abstract vectors. We employ *feature visualization* to transform them into a more meaningful semantic representation using techniques proposed by Olah *et al.* [OMS17, OSJ*18]. While the neural network transforms an input image into activation vectors, feature visualization goes in the opposite direction.

Given an activation vector $h_{x,y}$ at a spatial position (x,y), feature visualization synthesizes an idealized image that would have produced $h_{x,y}$ via an iterative optimization process. Normally, this is achieved by maximizing the dot product $h_{x,y} \cdot v$ of the vector $h_{x,y}$ with the direction v. However, the vector v that maximizes the dot product can have a large orthogonal component. To counter this, following [CAS*19], the dot product is multiplied with a cosine similarity, putting greater emphasis on the angle between vectors. The optimization process, which is similar to back propagation, begins with a random noise image. Using gradient descent, this image is iteratively tweaked to maximize the following objective: $\frac{(h_{x,y} \cdot v)^{n+1}}{(||h_{x,y}|| ||v||)^n}$. Subsequently, a *transformation robustness* regularizer [OMS17] is used, which applies small stochastic transformation (jitter, rotate or scale) to the image before applying the optimization step. Max-pooling can introduce high frequencies in the gradients. To tackle this issue, the gradient descent is performed in Fourier basis with frequencies scaled to have equal energy, which is equivalent to whitening and de-correlating the data.

We apply feature visualization to each of the 300,000 activation vectors to obtain individual *activation images*. Once we obtain a summary graph, we also apply feature visualization to the averaged activation vector per cluster (node) to obtain an *averaged activation image* for each cluster. Feature visualization is not without drawbacks; due to the optimization process and the size of each cluster, it can generate abstract images that remain hard to interpret.

6. Exploring the Shape of Activations from InceptionV1

We present the user interface of **TopoAct**, an interactive system used to explore the organizational principles of neuron activations in deep learning image classifiers. We use InceptionV1 trained on 1.2 million ImageNet images across 1,000 classes. We obtain activation vectors of 300,000 images (300 images per class) via the pre-trained model. The **TopoAct** user interface contains two exploration modes: single layer exploration and multilayer exploration; see the supplementary materials for detailed description on the interface and implementational details, including the multilayer exploration. Figure 3 illustrates the user interface under single layer exploration mode.

We present various exploration scenarios using **TopoAct** that provide valuable insights towards learned representations of InceptionV1. Visual exploration of the space of activations takes two forms: single layer exploration and multilayer exploration. For single layer exploration, the main takeaway from these scenarios is that **TopoAct** captures specific topological structures, in particular, branches and loops, in the space of activations that are hard to detect via classic DR techniques; and such features offer new perspectives on how a neural network “sees" the input images. The topological features identified by **TopoAct** can also be used to guide refined, local shape analysis of the space of activations. For multilayer exploration (see supplementary material), the mapper graphs within a single layer and across layers give interesting perspectives on the global organizational principles of the activation space.

6.1. Discovering Branches from the Space of Activations

We begin by providing examples of interesting topological structures that capture relationships between activations during single layer exploration. We notice that by replacing nodes of a mapper graph by averaged activation images, we obtain the most interpretable observations. There are two main types of topological structures that are unique to our framework, branches and loops, which differentiate **TopoAct** from prior work (e.g., [HPRC19, CAS*19]). Topologically, branches in a graph represent bifurcations, thus separations among image classes. While we observe variations of similar features along a specific branch, different branches may capture distinct, sometimes unrelated features.

Leg-face bifurcation. Our first example of a bifurcation comes from the layer *4c* of the *overlap-30-epsilon–adaptive* dataset. Figure 4 shows two branches emerging from a node in the mapper

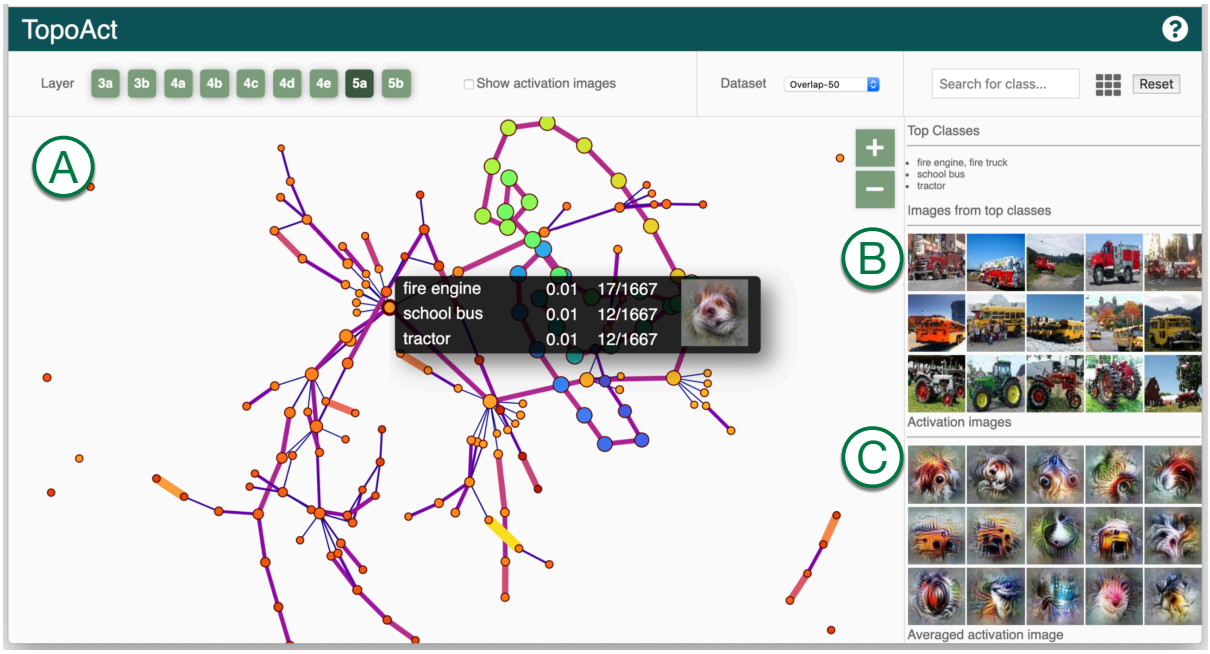

Figure 3: *With **TopoAct**, users can interactively explore topological summaries of activations in a neural network for a single layer and across multiple layers. Users can investigate activations at a particular layer under the single layer exploration mode. (A) Summary graph view provides a graph-based topological summary of the activation vectors from 300,000 images across 1,000 classes, where each node of the summary graph represents a cluster of activation vectors and each edge encodes the relationships between the clusters. For a chosen cluster in the summary graph, data example view (B) gives textual description of the top three classes within the cluster together with five image examples from each top class. The Feature visualization view (C) applies feature inversion to generate idealized images (called activation images) for individual activation vectors (obtained from data examples) as well as an averaged activation vector within a chosen cluster.*

graph; we refer to such as node as the *branching node*. It is composed of 381 activation vectors. The top three classes within the branching node are rugby ball, Indian elephant, and wig. While this appears to be random, the mapper graph coupled with averaged activation images reveals interesting insights.

The left branch appears to capture the leg of an animal. The top three classes represented in all the clusters within this branch include various breeds of dogs (Figure 4a). The right branch appears to capture features that resemble human faces, although distorted. While class names associated with clusters along the right branch may not suggest a relation to human faces; the data examples associated with these clusters reveal that all the top classes in the right branch contain images with humans, most of which include close-ups of faces (Figure 4b, 4c). Returning to the branching node, upon closer inspection (Figure 4d), we see that it contains images of rugby players and elephants which include both leg and face features, while wig images also include human faces. Therefore, the activation space bifurcates at the branching node to further differentiate between leg and face features.

We further compare **TopoAct** against t-SNE and UMAP projections. As illustrated in Figure 5, we select nodes that are involved in the leg-face bifurcation and highlight their corresponding activation vectors in the t-SNE and UMAP projections using the linked-view. In particular, neither t-SNE nor UMAP reveal a bifurcation in

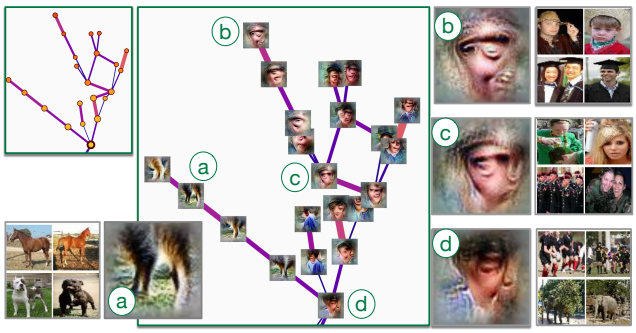

Figure 4: *Leg-face bifurcation.*

the embedding as the activation vectors in the projection are scattered over the entire 2D space (only t-SNE embeddings are shown, UMAP embedding appears similar hence omitted). Similarly scattered patterns are observed for other bifurcations as well but are not shown in interest of space.

Bird-mammal bifurcation. Our second example of a branch comes from the layer *5a* of the *overlap-30-epsilon-fixed* dataset. The branching node, in this case, is composed of 398 activation vectors. Figure 6 shows two branches emerging from this node. It contains images of both birds and dogs (Figure 6a), and the (aver-

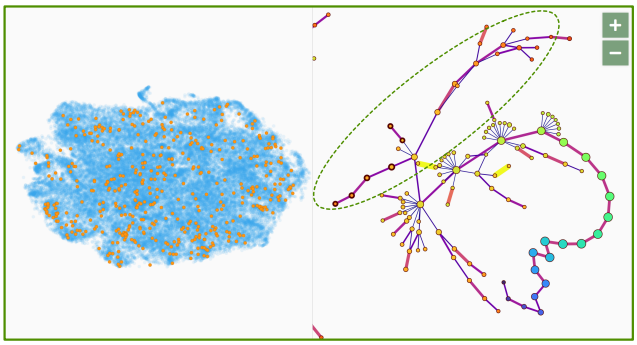

Figure 5: *Highlighting activation vectors that belong to the leg-face bifurcation as orange points in the t-SNE projection (left).*

aged) activation image of the branching node appears to be a combination of the left profile of bird faces and right profile of the dog-like faces. Upon further investigation, the bottom branch focuses on the features of bird faces: profile views composed of the left eye and beak, with variations of color and textures as we move along the branch. The clusters in this branch include mainly bird species. The variations in the captured features and corresponding data samples can be seen in Figures 6d-6f. The clusters in the top branch, on the other hand, appear to capture features of animal (mammal) faces: eyes and snout, with variations in color and texture. This branch is mainly composed of images of mammals, including various dog breeds (Figures 6b-6c), wolves, foxes and even monkeys.

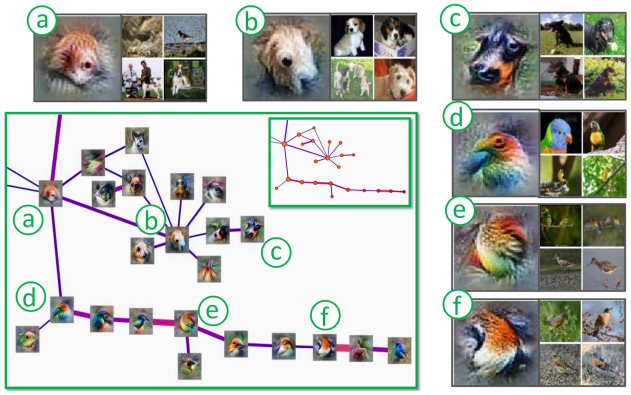

Figure 6: *Bird-mammal bifurcation.*

Geometry-text bifurcation. Our third example comes from the layer *4b* of the *overlap-30-epsilon-fixed* dataset. The branching node is a relatively large cluster with size 5126. Figure 7a shows examples of images in this cluster which include classes like bison and speedboat. Since this cluster is made up of several very different classes, the corresponding averaged activation image is less representative of any particular class. There are several branches emerging from this branching node, some of them being very short. We will focus on the three longest branches shown in Figure 7. The bottom two branches appear to capture patterns related to the text in the data images. This is somewhat obvious for the middle branch since the top classes of the clusters in this branch include things like menus. Figure 7f shows some examples of these data images. The top classes in the bottom branch include images of birds, dogs, computer mouse, etc. However, we can observe in Figures 7b and 7c that all these images contain some text. In the case of bird and animal images, this could be information such as copyright or the time and location where the images were captured. The

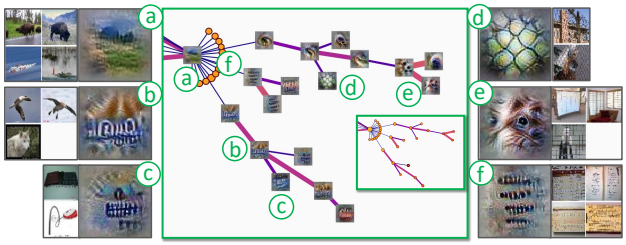

Figure 7: *Geometry-text bifurcation.*

top branch is a mixture of two types of classes - one type consists of window screens, “shoji", and chainlink fences. Clusters along the top branch seem to capture the geometric patterns found in such objects, see Figure 7d. However, clusters on this branch also contain images of dogs, wolves, etc. Indeed, in some cases, as in Figure 7e, the animal could be behind the fence. Because of such cases, the clusters in the top branch seem to capture a combination of geometric patterns from window screens and fences, as well as features related to eyes and noses of animals.

Wheel-tread bifurcation. Our fourth example comes from the layer *4c* of the *overlap-30-epsilon-adaptive* dataset. The branching node is a small cluster of size 136. All the clusters in this example contain images of various types of automobiles, for example minibus, police van, fire engine, limousine, etc. The branching node appears to capture what looks like a wheel of a vehicle - dark round shape with tread-like pattern seen in Figure 8a. The two branches of this branching node appear to focus on one of these two features. While the left branch focuses on the dark round swirling patterns of automobile wheels (Figures 8c, 8d), the right branch appears to focus more on the tread-like patterns and textures (Figure 8b).

6.2. Exploring Loops from the Space of Activations

While branches capture bifurcations in the types of features across different objects, loops seem to capture different aspects of the same underlying object.

Fur-nose-ear-eye loop of mammals. Our first example comes from layer 4d of the *overlap-30-epsilon-fixed* dataset. Figure 9 shows a loop created by six clusters. The top classes in all six clusters include various dog breeds, and a variety of foxes. All these clusters seem to capture different features (i.e., body parts) related to these animals. The left most cluster appears to capture the color patterns and the texture of the fur from the body (Figure 9a). Going clockwise, the next cluster also captures the color and texture of the fur but from a different body part, possibly the fur and hair

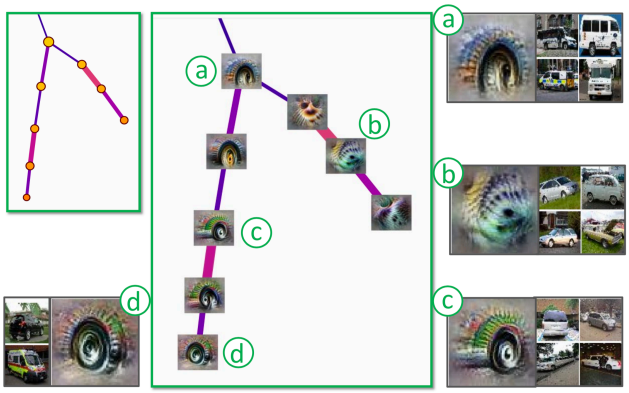

Figure 8: *Wheel-tread bifurcation.*

surrounding the nose, suggested by the dark spot and the swirling pattern (Figure 9b). The next two clusters (Figures9c, 9d) appear to capture animal ears. The averaged activation image captured by the fifth cluster (Figure 9e) is not as clearly attributable to specific part of animal body. It can be observed in Figure 9e, this cluster consists of images from a larger variety of animals, from foxes to Siamese cats and hogs. As a result, the corresponding averaged activation image is a mixture of various colors and slightly different textures. The last cluster appears to capture eyes and noses of the animals. We can observe in Figure 9f that the cluster contains front and side views of dog heads.

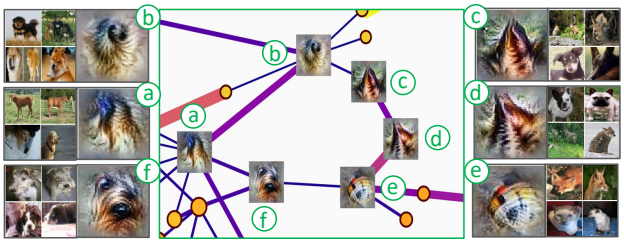

Figure 9: *Fur-nose-ear-eye loop.*

Face-body-leg loop of birds. Our next example originates from the layer 5a of the *overlap-30-epsilon-adaptive* dataset. Figure 10 shows six clusters creating the loop. The top three classes of all the clusters in the loop consist of bird species, and similar to the previous example, the averaged activation images show us different features (body parts) of the birds captured by these clusters. The three clusters (c-e) in the top part of the loop appear to capture the left profile views of the bird faces with the left eye and the beak identifiable in the averaged activation image. Figures 10c-10e show that these clusters are in fact composed of bird images. The variation in the color of birds (from red to blue, and to brown) is reflected in the corresponding activation images (b, c, d, e). The three clusters on the bottom (a, b, f) appear to capture the body and legs along with a feathered texture, although not as clearly as the other three clusters. It can be seen, in Figure 10f, that cluster f also includes images of tigers mixed with images of birds for the representation of texture.

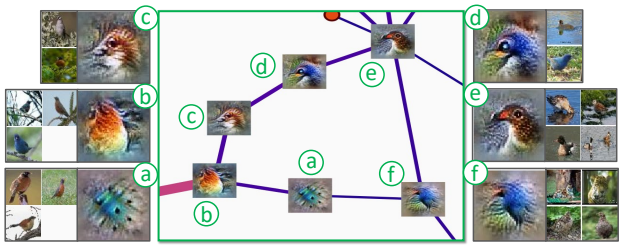

Figure 10: *Face-body-leg loop of birds.*

6.3. Studying Global View of Topological Summaries

We now explore the global view of a topological summary using the single layer exploration mode. Instead of focusing on a single type of topological structure such as loops or branches, we investigate the distribution of topological structures. As illustrated in Figure 11, we investigate the distribution of branches within the largest connected component of a mapper graph, at layer 5b of the dataset *overlap-30-epsilon-adaptive*.

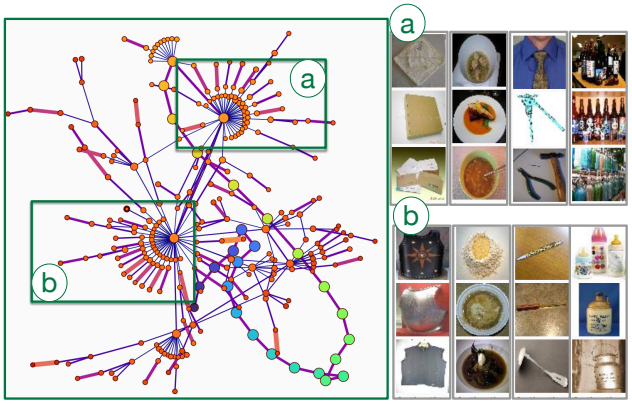

Figure 11: *A mapper graph for layer 5b,* overlap-30-epsilon-adaptive.

We pay special attention to branching nodes with high degrees (Figure 11a, 11b). These branching nodes, in some sense, serve as “anchors" or “hubs" of the underlying space of activations. There are a few interesting (however speculative) observations. For each of the two branching nodes in (a) and (b), there is a mixture of geometric and texture-based images that contribute towards the representation of the node. Nodes immediately adjacent to the branching node (a), *i. e.*, those that form branches that merge at node (a), contain geometric objects that are square-shaped (envelop, bath towel), circle-shaped (bowl, pasta), pointy-shaped (tie, hammer), and bottle-shaped (beer bottles). Nodes immediately adjacent to the branching node (b) have other objects that serve similar purposes, including square-shaped (vest, cuirass), circle-shaped (dough, mashed potato), pointy-shaped (ladle, ball point), and bottle-shaped (milk cans, whiskey jug). However, (a) and (b) seem to draw these geometric shapes from (almost completely)

different classes of images. This may indicates a level of self-similarity within the space of activations, which requires further investigation.

6.4. Refined Analysis of Topological Structures

We can utilize interesting topological structures identified by **TopoAct** – branches and loops – to obtain topologically meaningful subsets of the activation vectors for further analysis. As an example, we present some instances of **TopoAct**-guided principal component analysis (PCA). The procedure is as follows: We first identify all nodes that form a branch or a loop within a mapper graph captured by **TopoAct**. We then find all activation vectors (a subset of the 300,000 activation vectors) that map to these nodes. Next, we apply PCA to the subset by computing the first two principal components and projecting these activation vectors on a 2D plane.

For our first example, consider the leg-face bifurcation illustrated in Figure 4. Figure 12 (left) shows the 2D projection of all activation vectors that participate in the bifurcation. The red points belong to the activation vectors from the "face" branch and the blue points belong to the "leg" branch. Alhough we have projected the high-dimensional activation vectors onto just two dimensions, we could still easily observe that points from the two branches lie along two distinct directions.

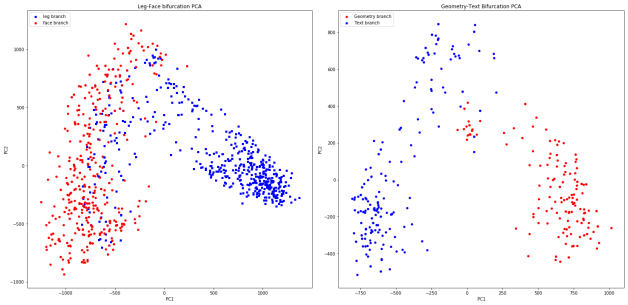

Figure 12: *PCA of the leg-face bifurcation (left) and the geometry-text bifurcation (right).*

For our second example, consider the geometry-text bifurcation scenario illustrated in Figure 7. Figure 12 (right) shows the 2D projection of the activation vectors that form this bifurcation. Once again, we can observe that the points of the "geometry" branch and the "text" branch, represented by red and blue colors respectively, lie along two distinct directions. Both examples also confirm that the branching structures detected by **TopoAct** are true features, not spurious ones.

These examples demonstrate how we can use **TopoAct** in combination with classic DR techniques (such as PCA) to perform refined shape analysis of the space of activations, which may not be revealed by performing DR (such as PCA, t-SNE, or UMAP) on the entire dataset. To illustrate this point, we apply PCA to all 300,000 activation vectors from the layer mixed4c from which we identified the leg-face bifurcation, as shown in Figure 13. Similarly to the t-SNE projection of the same layer (Figure 5 left), it is difficult to identify any particular directions along which data points concentrate. While the total number of activation vectors in our dataset

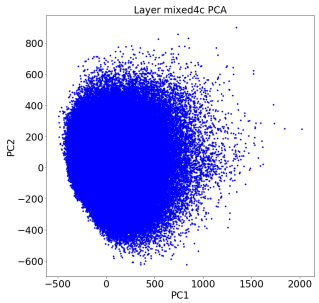

Figure 13: *PCA projection of activation vectors from mixed4c.*

is relatively large, the number of significant branches and loops is small. This leads us to hypothesize that there are certain directions in the activation space that a layer is particularly well-trained to identify. As shown in this section, **TopoAct** can help us identify these directions.

7. Applying TopoAct to ResNet Trained on CIFAR

To demonstrate the generality of our framework, we provide additional experiments using ResNet trained on the CIFAR datasets [KH09]: CIFAR-10 and CIFAR-100. Both datasets consist of the same set of $60,000$ color images of dimension 32×32. There are $50,000$ training images and $10,000$ test images. CIFAR-10 has 10 images classes with $6,000$ images per class and CIFAR-100 has 100 image classes with 600 images per class. The classes in CIFAR-10 are coarser such as "automobiles" and "mammals"; whereas classes in CIFAR-100 are finer such as "bicycle", "bus", "beaver" and "hamster". We demonstrate that the insights provided by **TopoAct** are not specific to a particular dataset or a particular network architecture. We give a few exploration scenarios involving branches by applying **TopoAct** to ResNet-18 trained on the CIFAR-10 dataset; such examples are similar to those described in Section 6. We encourage the readers to explore further with our open-sourced online demo.

Horse-deer bifurcation. Our first example is a horse-deer bifurcation from the layer *4.1.bn2* of the *CIFAR-10* dataset with number of intervals set to 40 and overlapping set to 20%. Figure 14 shows two branches emerging from a branching node in the mapper graph. The left branch that contains nodes b and c corresponding to deer images; while the right branch with nodes d, e and f contain only horse images. The branching node a contains both horse and deer images. This is an interesting example that reminds us of the brown bear vs. black bear story of SUMMIT [HPRC19].

We would also like to note that none of the earlier layers show such a clear bifurcation between the horse and deer classes. This is an example where **TopoAct** reveals the layer at which the network first begins to differentiate between the two classes. Such insights would make **TopoAct** a useful diagnostic tool for deep learning researchers (see Section 8).

Frog-cat bifurcation. Similarly, our second example is a frog-cat bifurcation from CIFAR-10 layer 4 with 100 intervals and 40% overlap, as shown in Figure 15. Here, the branching node (a) contains images of frogs and cats. It then bifurcates into a left branch

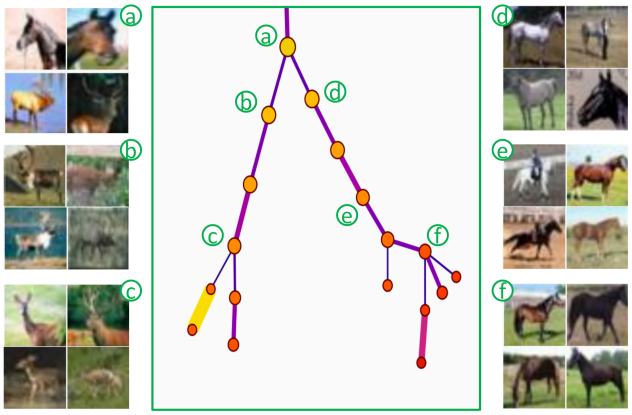

Figure 14: *Horse-deer bifurcation.*

(with nodes b and c) that contains only images of cats, and a right branch (with nodes d, e, and f) that contains only images of frogs. Even though these are very different types of animals (mammals vs. amphibians), they share similar postures.

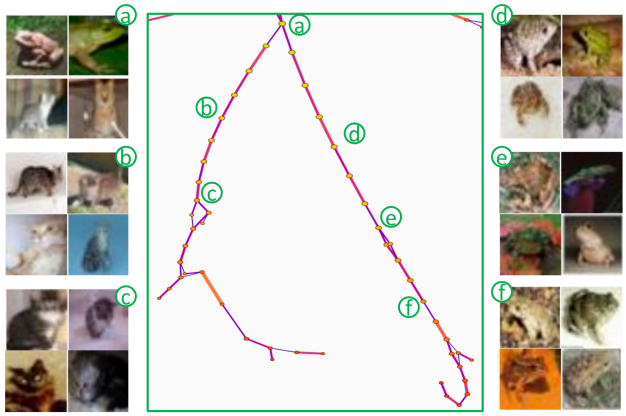

Figure 15: *Frog-cat bifurcation.*

8. Discussion

TopoAct supports exploratory analysis of numerous interesting topological structures, locally and globally, in the space of activation vectors. We encourage readers to utilize the live demo to obtain interpretable and insightful observations. Our approach has its limitations. The exploration scenarios presented here are specific to the choice of input images as well as the choice of activation vectors. Further analysis is required to determine how stable the results are with respect to these choices. However, some of these limitations are common to other recent approaches (e.g., [CAS*19,HPRC19]). We offer some topics for discussion and future work.

Generality. We focus on CNNs, specifically, InceptionV1 and ResNet-18 in this paper. However, our approach is not restricted to a particular network architecture. Mapper graphs could be generated and used as a vehicle for visual exploration whenever neuron activations are present. **TopoAct** can be generalized to explore new datasets coupled with other pre-trained network architectures, such as ZFNet [ZF12], AlexNet [KSH12], and VGGNet [SZ15].

Parameter tuning. Practical and automatic parameter tuning for the mapper construction remains a challenging open problem for the broad TDA community while active research is underway.

Carriere *et al.* [CMO18] recently provided the state-of-the-art, albeit theoretical results on mapper parameter selection under very restrictive settings. Their framework assumed that a point cloud sample taken from the underlying space has a well-behaved, parametrizable probability distribution (formally, an (a,b)-standard distribution) and that the sample is sufficiently large, so that the Hausdorff distance between the sample and the underlying space is small. However, upon careful investigation, these assumptions are not applicable in our setting. While we may assume that the activation space is a compact subset of the Euclidean space, we cannot verify that the activation vectors we sample follow the generative model of an (a,b)-standard distribution, and that $300,000$ vectors is a sufficiently large sample for approximating or possibly reconstructing the underlying space.

On the other hand, the mapper construction comes with "best practices" in terms of parameter tuning, which rely on grid search in the parameter space where good parameter combinations are those that produce stable structures. Finding a theoretically sound and yet practical parameter tuning strategy for our mapper graph construction remains open; see the supplementary materials for more discussions on this topic. For the current version of the **TopoAct**, we focus on exploring various mapper graphs with pre-determined sets of parameter combinations following best practices.

Stability. There are additional theoretical results regarding the stability of mapper construction (see [BBMW19] and references within). It is an on-going investigation into how stable the mapper graphs are with respect to different sampling techniques. Under some assumptions on the sampling condition, Brown *et al.* [BBMW19] showed that a pair of mapper graphs is close if their underlying point clouds are sampled from the same probability density function concentrated on the underlying topological space. However, similar to the situation of parameter tuning, there is still a large gap between theory and practice. Filling such a gap is beyond the scope of this paper.

Adversarial attacks. An important aspect in understanding the effectiveness of adversarial attacks on neural networks is understanding how an attack alters the intermediate representations, i.e. the activations. **TopoAct** visualizes these representations from a topological perspective and hence might be useful in analyzing the effect of adversarial attacks at different layers of the network.

Corrective actions during training. A branching point (a bifurcation) in the space of activations at a particular layer may indicate the point where the network starts distinguishing a pair of classes. This knowledge can be very useful to inform corrective actions for inputs in the test data that are being misclassified. For example, if two classes that bifurcate at a particular layer in **TopoAct** are still being misclassified into each other, an expert can choose to increase the network width at subsequent layers, or to selectively augment the training data for these classes to encourage better separation.

References

[Ala12] ALAGAPPAN M.: From 5 to 13: Redefining the positions in basketball. *MIT Sloan Sports Analytics Conference* (2012). 2

[Ale28] ALEKSANDROFF P. S.: Über den allgemeinen dimensionsbegriff und seine beziehungen zur elementaren geometrischen anschauung. *Mathematische Annalen 98*, 1 (1928), 617–635. 3

[BBMW19] BROWN A., BOBROWSKI O., MUNCH E., WANG B.: Probabilistic convergence and stability of random mapper graphs. *arXiv:1909.03488* (2019). 10

[BGSF08] BIASOTTI S., GIORGI D., SPAGNUOLO M., FALCIDIENO B.: Reeb graphs for shape analysis and applications. *Theoretical Computer Science 392* (2008), 5–22. 3

[BMMP03] BIASOTTI S., MARINI S., MORTARA M., PATANE G.: An overview on properties and efficacy of topological skeletons in shape modelling. *Shape Modeling International* (2003). 3

[BWN*15] BHATIA H., WANG B., NORGARD G., PASCUCCI V., BREMER P.-T.: Local, smooth, and consistent Jacobi set simplification. *Computational Geometry: Theory and Applications (CGTA) 48*, 4 (2015), 311–332. doi:10.1016/j.comgeo.2014.10.009. 2

[BYM*14] BEKETAYEV K., YELIUSSIZOV D., MOROZOV D., WEBER G., HAMANN B.: Measuring the distance between merge trees. *Topological Methods in Data Analysis and Visualization III: Theory, Algorithms, and Applications, Mathematics and Visualization* (2014), 151–166. 2

[CAS*19] CARTER S., ARMSTRONG Z., SCHUBERT L., JOHNSON I., OLAH C.: Activation atlas. *Distill 4*, 3 (2019), e15. 2, 3, 4, 5, 10

[Cay08] CAYTON L.: *Algorithms for manifold learning*. Tech. Rep. CS2008-0923, University of California at San Diego, 2008. 3

[CISZ08] CARLSSON G., ISHKHANOV T., SILVA V. D., ZOMORODIAN A.: On the local behavior of spaces of natural images. *International journal of computer vision 76*, 1 (2008), 1–12. 3

[CMO18] CARRIÈRE M., MICHEL B., OUDOT S.: Statistical analysis and parameter selection for mapper. *Journal of Machine Learning Research 19*, 12 (2018), 1–39. URL: http://jmlr.org/papers/v19/17-291.html. 10

[CSA03] CARR H., SNOEYINK J., AXEN U.: Computing contour trees in all dimensions. *Computational Geometry 24*, 2 (2003), 75–94. 2

[CZJ*19] CHO H. J., ZHAO J., JUNG S. W., LADEWIG E., KONG D.-S., SUH Y.-L., LEE Y., KIM D., AHN S. H., BORDYUH M., KANG H. J., SA J. K., SEO Y. J., KIM S. T., LIM D. H., DHO Y.-S., LEE J.-I., SEOL H. J., CHOI J. W., PARK W.-Y., PARK C.-K., RABADAN R., NAM D.-H.: Distinct genomic profile and specific targeted drug responses in adult cerebellar glioblastoma. *Neuro-Oncology 21*, 1 (2019), 47–58. 2

[DDS*09] DENG J., DONG W., SOCHER R., LI L.-J., LI K., FEI-FEI L.: ImageNet: A large-scale hierarchical image database. *IEEE Conference on Computer Vision and Pattern Recognition* (2009). 4

[EBCV09] ERHAN D., BENGIO Y., COURVILLE A., VINCENT P.: Visualizing higher-layer features of a deep network. *Technical Report, Univeristy of Montreal* (01 2009). 2

[EH02] EDELSBRUNNER H., HARER J.: Jacobi sets of multiple Morse functions. In *Foundations of Computational Mathematics, Minneapolis 2002* (2002), Cucker F., DeVore R., Olver P., Süli E., (Eds.), Cambridge University Press, pp. 37–57. 2

[EHP08] EDELSBRUNNER H., HARER J., PATEL A.: Reeb spaces of piecewise linear mappings. In *Proceedings of the 24th annual symposium on Computational geometry* (2008), pp. 242–250. 2

[EKSX96] ESTER M., KRIEGEL H.-P., SANDER J., XU X.: A density-based algorithm for discovering clusters in large spatial databases with noise. *International Conference on Knowledge Discovery and Data Mining* (1996), 226–231. 4, 5

[GAES19] GABELLA M., AFAMBO N., EBLI S., SPREEMANN G.: Topology of learning in artificial neural networks. arXiv:1902.08160, 2019. 3

[GSH19] GEBHART T., SCHRATER P., HYLTON A.: Characterizing the shape of activation space in deep neural networks. *Proceedings of the 18th IEEE International Conference on Machine Learning and Applications (ICMLA)* (2019). 3

[GSPS19] GENIESSE C., SPORNS O., PETRI G., SAGGAR M.: Generating dynamical neuroimaging spatiotemporal representations (DyNeuSR) using topological data analysis. *Network Neuroscience 3*, 3 (2019). 2

[HKPC18] HOHMAN F., KAHNG M., PIENTA R., CHAU D. H.: Visual analytics in deep learning: An interrogative survey for the next frontiers. *IEEE Transactions on Visualization and Computer Graphics (TVCG)* (2018). 2

[HPRC19] HOHMAN F., PARK H., ROBINSON C., CHAU D. H.: Summit: Scaling deep learning interpretability by visualizing activation and attribution summarizations. *IEEE Transactions on Visualization and Computer Graphics* (2019). 2, 4, 5, 9, 10

[HWR18] HAJIJ M., WANG B., ROSEN P.: Mapper on graphs for network visualization. *arXiv:1804.11242* (2018). 2

[HWSR18] HAJIJ M., WANG B., SCHEIDEGGER C., ROSEN P.: Visual detection of structural changes in time-varying graphs using persistent homology. *IEEE Pacific Visualization Symposium (PacificVis)* (2018). 2

[HZRS16] HE K., ZHANG X., REN S., SUN J.: Deep residual learning for image recognition. In *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)* (June 2016), pp. 770–778. doi:10.1109/CVPR.2016.90. 1, 4

[JCR*19] JEITZINER R., CARRIÉRE M., ROUGEMONT J., OUDOT S., HESS K., BRISKEN C.: Two-tier mapper, an unbiased topology-based clustering method for enhanced global gene expression analysis. *Bioinformatics 35*, 18 (2019), 3339–3347. 2

[Kar14] KARPATHY A.: t-SNE visualization of CNN codes. https://cs.stanford.edu/people/karpathy/cnnembed/, 2014. 2, 3

[KGCFR*20] KNUDSON A., GONZÁLEZ-CASABIANCA F., FEGED-RIVADENEIRA A., PEDREROS M. F., APONTE S., OLAYA A., CASTILLO C. F., MANCILLA E., PIAMBA-DORADO A., SANCHEZ-PEDRAZA R., SALAZAR-TERREROS M. J., LUCCHI N., UDHAYAKUMAR V., JACOB C., PANCE A., CARRASQUILLA M., APRÁEZ G., ANGEL J. A., RAYNER J. C., CORREDOR V.: Spatio-temporal dynamics of plasmodium falciparum transmission within a spatial unit on the colombian pacific coast. *Scientific Reports 10*, 3756 (2020). 2

[KH09] KRIZHEVSKY A., HINTON G.: *Learning multiple layers of features from tiny images*. Tech. rep., University of Toronto, 2009. 4, 9

[KSH12] KRIZHEVSKY A., SUTSKEVER I., HINTON G. E.: Imagenet classification with deep convolutional neural networks. *25th Advances in Neural Information Processing Systems 25* (2012). 10

[KWG*18] KIM B., WATTENBERG M., GILMER J., CAI C., WEXLER J., VIEGAS F., SAYRES R.: Interpretability beyond feature attribution: Quantitative testing with concept activation vectors (TCAV). *International Conference on Machine Learning* (2018). 2

[LCY14] LIN M., CHEN Q., YAN S.: Network in network. *International Conference on Learning Representations (ICIR)* (2014). 4

[Le13] LE Q. V.: Building high-level features using large scale unsupervised learning. In *2013 IEEE International Conference on Acoustics, Speech and Signal Processing* (May 2013), pp. 8595–8598. 2

[LMW*17] LIU S., MALJOVEC D., WANG B., BREMER P.-T., PASCUCCI V.: Visualizing high-dimensional data: Advances in the past decade. *IEEE Transactions on Visualization and Computer Graphics (TVCG) 23*, 3 (2017), 1249–1268. doi:10.1109/TVCG.2016.2534538. 3

[LPM03] LEE A. B., PEDERSEN K. S., MUMFORD D.: The non-linear statistics of high-contrast patches in natural images. *International Journal of Computer Vision 54* (2003), 83–103. 3

[MH08] MAATEN L. V. D., HINTON G.: Visualizing data using t-SNE. *Journal of Machine Learning Research 9* (2008), 2579–2605. 2, 3

[MHM18] McInnes L., Healy J., Melville J.: Umap: Uniform manifold approximation and projection for dimension reduction. *arXiv:1802.03426* (2018). 2, 3

[MHSG18] McInnes L., Healy J., Saul N., Grossberger L.: UMAP: Uniform manifold approximation and projection. *Journal of Open Source Software 3*, 29 (2018), 861. 2

[MNL*19] Mathews J. C., Nadeem S., Levine A. J., Pouryahya M., Deasy J. O., Tannenbaum A.: Robust and interpretable pam50 reclassification exhibits survival advantage for myoepithelial and immune phenotypes. *NPJ Breast Cancer 5*, 30 (2019). 2

[MV15] Mahendran A., Vedaldi A.: Understanding deep image representations by inverting them. In *2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)* (June 2015), pp. 5188–5196. 2

[MV16] Mahendran A., Vedaldi A.: Visualizing deep convolutional neural networks using natural pre-images. *International Journal of Computer Vision 120*, 3 (2016), 233–255. 2

[MW16] Munch E., Wang B.: Convergence between categorical representations of Reeb space and mapper. *International Symposium on Computational Geometry* (2016). 2

[NLC11] Nicolau M., Levine A. J., Carlsson G.: Topology based data analysis identifies a subgroup of breast cancers with a unique mutational profile and excellent survival. *Proceedings of the National Academy of Sciences 108*, 17 (2011), 7265–7270. 2

[NYC16] Nguyen A., Yosinski J., Clune J.: Multifaceted feature visualization: Uncovering the different types of features learned by each neuron in deep neural networks. *arXiv preprint arXiv:1602.03616* (2016). 2, 3

[OMS17] Olah C., Mordvintsev A., Schubert L.: Feature visualization. *Distill 2*, 11 (2017), e7. 4, 5

[OSJ*18] Olah C., Satyanarayan A., Johnson I., Carter S., Schubert L., Ye K., Mordvintsev A.: The building blocks of interpretability. *Distill 3*, 3 (2018), e10. 4, 5

[PVP17] Patania A., Vaccarino F., Petri G.: Topological analysis of data. *EPJ Data Science 6*, 7 (2017). 2

[Ree46] Reeb G.: Sur les points singuliers d'une forme de pfaff completement intergrable ou d'une fonction numerique (on the singular points of a complete integral pfaff form or of a numerical function). *Comptes Rendus Acad.Science Paris 222* (1946), 847–849. 2

[SHW*20] Suh A., Hajij M., Wang B., Scheidegger C., Rosen P.: Persistent homology guided force-directed graph layouts. *IEEE Transactions on Visualization and Computer Graphics 26*, 1 (2020), 697–707. 2

[SLJ*15] Szegedy C., Liu W., Jia Y., Sermanet P., Reed S., Anguelov D., Erhan D., Vanhoucke V., Rabinovich A.: Going deeper with convolutions. *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)* (2015). 1, 4

[SMC07] Singh G., Mémoli F., Carlsson G.: Topological methods for the analysis of high dimensional data sets and 3d object recognition. *Eurographics Symposium on Point-Based Graphics 22* (2007). 1, 2, 3

[SMVJ09] Silva V. D., Morozov D., Vejdemo-Johansson M.: Persistent cohomology and circular coordinates. *Proceedings of the 25th Annual Symposium on Computational Geometry* (2009), 227–236. 3

[SSGC*18] Saggar M., Sporns O., Gonzalez-Castillo J., Bandettini P. A., Carlsson G., Glover G., Reiss A. L.: Towards a new approach to reveal dynamical organization of the brain using topological data analysis. *Nature Communications 9*, 1399 (2018). 2

[SVZ14] Simonyan K., Vedaldi A., Zisserman A.: Deep inside convolutional networks: Visualising image classification models and saliency maps. *Workshop at International Conference on Learning Representations,* (2014). 2

[SZ15] Simonyan K., Zisserman A.: Very deep convolutional networks for large-scale image recognition. *3rd International Conference on Learning Representations (ICLR)* (2015). 10

[WSPVJ11] Wang B., Summa B., Pascucci V., Vejdemo-Johansson M.: Branching and circular features in high dimensional data. *IEEE Transactions on Visualization and Computer Graphics 17*, 12 (2011), 1902–1911. doi:10.1109/TVCG.2011.177. 3

[Xia16] Xia S.: A topological analysis of high-contrast patches in natural images. *Journal of Nonlinear Sciences and Applications 9* (2016), 126–138. 3

[YCN*15] Yosinski J., Clune J., Nguyen A., Fuchs T., Lipson H.: Understanding neural networks through deep visualization. *Deep Learning Workshop at the 31st International Conference on Machine Learning* (2015). 2

[YZR*18] Yan L., Zhao Y., Rosen P., Scheidegger C., Wang B.: Homology-preserving dimensionality reduction via manifold landmarking and tearing. *Symposium on Visualization in Data Science (VDS) at IEEE VIS* (2018). 3

[ZF12] Zeiler M. D., Fergus R.: Visualizing and understanding convolutional networks. *The 26th Annual Conference on Neural Information Processing Systems* (2012). 10

[ZF14] Zeiler M. D., Fergus R.: Visualizing and understanding convolutional networks. In *Computer Vision – ECCV 2014* (2014), Springer International Publishing, pp. 818–833. 2