doi:10.3233/FAIA240783

Be Persistent: Towards a Unified Solution for Mitigating Shortcuts in Deep Learning

Hadi M. Dolatabadi, Sarah M. Erfani[*] and Christopher Leckie

School of Computing and Information Systems, The University of Melbourne, Parkville, Victoria, Australia.

Abstract. Deep neural networks (DNNs) are vulnerable to shortcut learning: rather than learning the intended task, they tend to draw inconclusive relationships between their inputs and outputs. Shortcut learning is ubiquitous among many failure cases of neural networks, and traces of this phenomenon can be seen in their generalizability issues, domain shift, adversarial vulnerability, and even bias towards majority groups. In this paper, we argue that this commonality in the cause of various DNN issues creates a significant opportunity that should be leveraged to find a unified solution for shortcut learning. To this end, we outline the recent advances in topological data analysis (TDA), and persistent homology (PH) in particular, to sketch a unified roadmap for detecting shortcuts in deep learning. We demonstrate our arguments by investigating the topological features of computational graphs in DNNs using two cases of unlearnable examples and bias in decision-making as our test studies. Our analysis of these two failure cases of DNNs reveals that finding a unified solution for shortcut learning in DNNs is not out of reach, and TDA can play a significant role in forming such a framework.

1 Introduction

Deep neural networks (DNNs) have reshaped the means of data processing and generation in the past decade. Many of the recent advances in different areas of machine learning, whether in natural language processing [44, 33] or computer vision [17, 39], can be attributed to the representational power of DNNs. For instance, it has been shown that neural networks can surpass human-level accuracy in object detection [34].

In contrast to humans, DNNs notoriously rely on spurious features and draw meaningless conclusions during decision-making [3, 38, 23]. For instance, consider a DNN trained over a dataset that always depicts cows on the grass. Such a model considers grass as a spurious feature, and only detects cows where the background includes greenery (see Figure 1). This simple example indicates a serious flaw in DNNs' decision-making, rendering it dangerous for sensitive applications to be at the mercy of neural networks [47].

Spurious correlations are argued to be the result of a broader phenomenon known as *shortcut learning* [14]. Shortcuts are unintended decision rules that are learned by neural networks (e.g., detecting grass instead of cows). These shortcuts are argued to be the result of following the "Principle of Least Effort" [51] in which learning the unintended solution takes less effort compared to the actual task [14] (e.g., detecting greenery would be much easier than detecting cows in different colors and shapes). Various failure cases of neural networks, from domain shift [46] and adversarial examples [40] to even bias and fairness [2], can be traced back to shortcut learning [14]. This surprising alignment in the cause of these failure cases is a great opportunity that should be leveraged towards finding a unified solution.

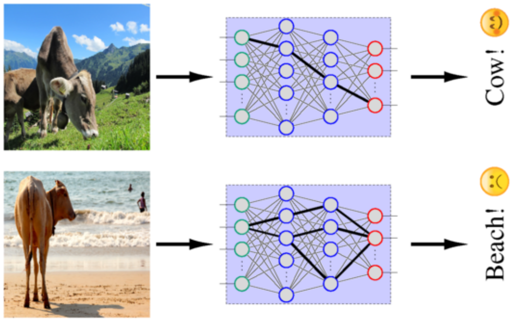


Figure 1. Neural networks exhibit spurious correlations, which is a special case of shortcut learning. For instance, DNN trained over data that depicted cows exclusively over grass has created a spurious correlation where cows are only detected in the presence of grass [3].

In this paper, we argue that topological data analysis (TDA) [6] can play an invaluable role in detecting and mitigating shortcuts in deep learning. In particular, we demonstrate that shortcut learning leaves tractable paths in the computational graph of a neural network. To unveil shortcuts in such a high-dimensional space containing thousands of neurons, we leverage the representational power of persistent homology (PH) [9] in revealing connected components between neurons. We empirically show that neural networks that learn such shortcuts leave a signature in the topology of their computational graph. Our experiments reveal that these features are statistically different from those of benign neural networks.

To demonstrate our point, we choose two failure cases of DNNs in down-stream tasks: unlearnable examples [22, 48] and bias [5]. Unlearnable examples are imperceptible perturbations added to the training data to prevent a DNN from learning meaningful patterns from the data [22]. On the other hand, bias in neural networks can happen when a DNN misuses certain sensitive attributes from the data (such as skin color) to make its final decision [5]. As seen, the origins of these two issues are quite different. However, we show that TDA can provide a unified means to shed light on both misbehaviors in DNNs. Our promising findings call for more research into the topological analysis of neural networks for treating shortcuts, with the hope that such a solution could mitigate various failures of DNNs once and for all.

Our contributions can be summarized as follows:

- We target shortcut learning as the main culprit among many

[*] Corresponding Author. Email: sarah.erfani@unimelb.edu.au

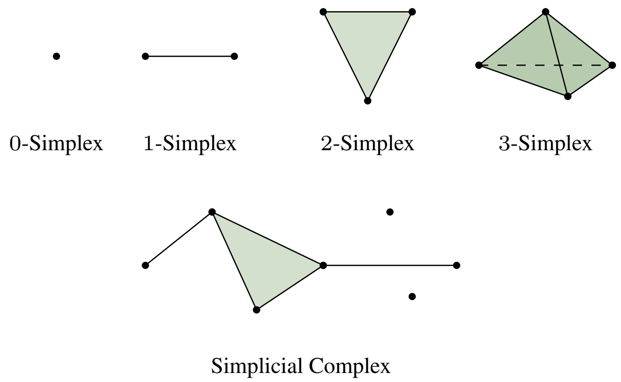


Figure 2. A k-Simplex can be regarded as the convex hull of $k+1$ points. A simplicial complex is a union of such simplices.

failure cases of neural networks [14] and argue that the research community should pay more attention to persistent homology to mitigate this issue. To the best of our knowledge, this is the first paper that proposes a unified roadmap for mitigating shortcut learning.

- We use two emerging issues in neural networks, namely unlearnable examples and bias in decision-making, as our case-studies to demonstrate the applicability of the same persistent homology framework to completely different issues in neural networks. Training more than a dozen models in each case, our experimental results demonstrate that persistent homology can easily reveal the differences between benign and affected models.
- Finally, we pinpoint some of the most important future research directions in this area that could lead to a unified solution for shortcut learning using persistent homology.

2 Background

This section reviews the background related to persistent homology. Rather than an in-depth mathematical discussion, we focus on covering the intuition and a general understanding of persistent homology. We refer the interested reader to Edelsbrunner and Harer [10] for a formal introduction to the topics discussed here.

2.1 Simplicial Homology

The field of computational topology [10] is concerned with designing the mathematical tools required for finding topological features of low- and high-dimensional manifolds. To this end, the notion of *homology groups* in a *simplicial complex* is used. Informally speaking, a simplicial complex is a discretized graph representation of the data that contains information about nodes, edges, triangles, and their higher order equivalents (see Figure 2). A homology group of rank d, denoted by $\mathcal{H}_d$, is a d-dimensional topological descriptor for simplicial complexes. In this paper, we mainly work with 0- and 1-dimensional homology groups, which encode the number of connected components and cycles/loops in a simplicial complex.

2.2 Persistent Homology

Persistent homology (PH) [1, 9] extends the notion of simplicial homology to data observations. Rather than treating the data as samples from their true underlying manifold, PH aims to identify the most *persistent* topological structures within the data points, assuming noisy observations. To this end, the evolution of topological features at multiple granularities using a *filtration* is observed. A filtration creates a sequence of simplicial complexes at growing granularities, which enables us to track the evolution of topological features between consecutive levels of granularity. Each topological structure is created at a particular granularity and destroyed at another one. We refer to these as the birth and death times of our topological features, respectively. We can then collect all these times as points in a 2-dimensional plane, creating a so-called *persistent diagram* (PD). Naturally, we would be interested in structures that have a long life-time/persistence (which is the subtraction of the death and birth time). This is because such structures are unlikely to have been generated by the noisy observations, and as such, are representing the true data manifold.

2.3 Vietoris-Rips Filtration

Vietoris-Rips (VR) [45] complex is a common method used for building a filtration of data points $\{\boldsymbol{a}_1, \boldsymbol{a}_2, \dots, \boldsymbol{a}_m\}$. To construct such a filtration, we start by defining a metric space over which we compare our data points using some notion of distance $\mathrm{d}(\cdot,\cdot)$. At each scale ϵ, we connect all the data points that are within the ϵ distance of each other, i.e., data points that satisfy $\mathrm{d}(\boldsymbol{a}_i, \boldsymbol{a}_j) \leq \epsilon$. We then sweep over all the possible ϵ's from $-\infty$ to $+\infty$ and record the birth and death of topological features of interest in a PD. As mentioned earlier, in this paper we are mostly interested in capturing the 0D and 1D topological features. Thus, we would have a separate PD for each dimension. An illustrated example of finding the VR filtration for a set of point clouds is given in Figure 3.

3 Framework

This section presents our framework for the topological analysis of shortcut learning in neural networks. First, we formally introduce our problem setting and notation. Then, we go over the details of the topological analysis of neural networks for shortcut learning.

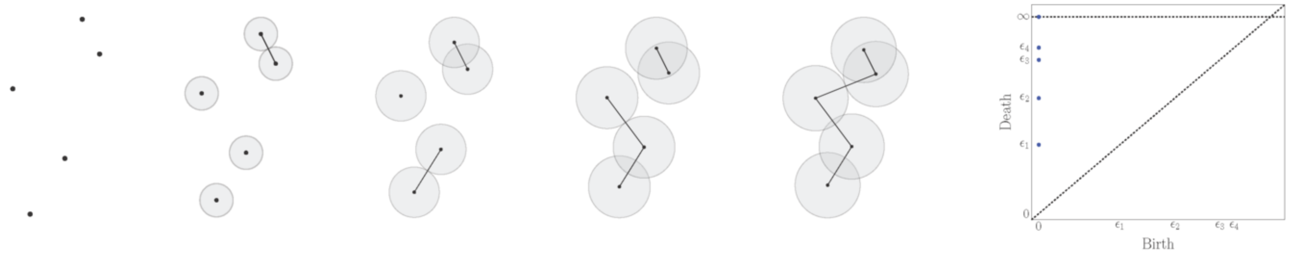
3.1 Problem Setting

Consider a labelled dataset $\mathcal{D} = \{(\boldsymbol{x}_i, y_i)\}_{i=1}^{n}$ where $\boldsymbol{x}_i \in \mathbb{R}^d$ and y_i denote a data sample and its associated label. Without loss of generality, in this paper, we consider a multi-class classification problem where y_i belongs to exactly one class from the set $\{1, 2, \dots, k\}$. Also, let $f_{\boldsymbol{\theta}} : \mathbb{R}^d \rightarrow \mathbb{R}^k$ denote a neural network classifier with parameters $\boldsymbol{\theta}$ that maps the inputs $\boldsymbol{x}$ to a real-valued vector $\boldsymbol{z} \in \mathbb{R}^k$, commonly known as the *logit* vector. The final prediction of the classifier is obtained via solving $\hat{y} = \arg\max_c \boldsymbol{z}[c]$. To train the classifier $f_{\boldsymbol{\theta}}$, we usually optimize a relevant objective over the dataset:

$$\boldsymbol{\theta}^* = \arg\min_{\boldsymbol{\theta}} \sum_{i=1}^{n} \mathcal{L}\left(f_{\boldsymbol{\theta}}\left(\boldsymbol{x}_i\right), y_i\right), \tag{1}$$

where $\mathcal{L}(\cdot,\cdot)$, called the loss function, is a measure of discrepancy between the prediction of the classifier and the ground-truth label. In this paper, we assume that our loss function is a cross-entropy function.

To pave the way for our discussion of topological analysis of DNNs, we need to define a few extra notations. Let m be the total number of neurons in a DNN. During inference, each input $\boldsymbol{x}_i$ is sent through the neural network and processed by its neurons. Let $a_j^{(i)}$ denote the value of the jth neuron for the ith data sample $\boldsymbol{x}_i$. We gather the



(a) ϵ_0 (b) ϵ_1 (c) ϵ_2 (d) ϵ_3 (e) ϵ_4 (f) Persistence Diagram

Figure 3. An example of computing the VR complex for a set of points in the Euclidean space. As the threshold increases, the 0-simplices gradually vanish. In the end, we would have only one connected component that lives forever.

values of the jth neuron for N data samples in a so-called activation vector, written as:

$$\boldsymbol{a}_j = [a_j^{(1)}, a_j^{(2)}, \dots, a_j^{(N)}]^\intercal. \tag{2}$$

Using this notation, if we get the activation values of all the m neurons of the DNN for N data samples, we end up having a collection of m activation vectors $\mathcal{A} = \{\boldsymbol{a}_1, \boldsymbol{a}_2, \dots, \boldsymbol{a}_m\}$. Next, we will see how we can create a topological map of a neural network computation graph using the set $\mathcal{A}$.

3.2 Topological Analysis of Shortcut Learning

As mentioned before, shortcut learning is commonplace in neural networks. Intuitively, this phenomenon happens when a DNN learns an unusually direct path from its input to its output. We aim to reveal this unusual behavior by analyzing the traversal of input information through the computational graph of the neural network. We resort to PH for identifying this abnormal behavior in DNNs. To this end, we construct a VR filtration of the neural network computation graph following a method introduced in Zheng et al. [50].

Recall from Section 2.3 that the first step during the construction of a VR filtration is to determine a meaningful distance measure between the data points. In our analysis, we are dealing with neurons as nodes in a computational graph. A shortcut happens when there are *multiple* neurons throughout the graph that are activated together. For this reason, we can use a standard correlation measure between the activation vectors of two different neurons as our distance metric in building a VR complex.

In other words, let us assume that $\boldsymbol{a}_i$ and $\boldsymbol{a}_j$ are two different activation vectors ($i \neq j$) constructed according to Equation (2). In addition, let $-1 \leq \rho(\boldsymbol{a}_i, \boldsymbol{a}_j) \leq 1$ be a normalized correlation metric between these two vectors. Then, we define the distance between these two activation vectors as:

$$\mathrm{d}(\boldsymbol{a}_i, \boldsymbol{a}_j) = 1 - \rho(\boldsymbol{a}_i, \boldsymbol{a}_j). \tag{3}$$

This means that if the activation vectors $\boldsymbol{a}_i$ and $\boldsymbol{a}_j$ are highly correlated, their distance is smaller and vice versa.

Using this distance metric, we next build a VR filtration for the set of all neurons $\mathcal{A}$. To this end, we assume that each neuron is denoted by a node in a graph. Then, we sweep over ϵ from $-\infty$ to $+\infty$ and connect all the neurons whose activation vectors fall within an ϵ distance of each other. This process would result in a filtration $\emptyset \subseteq \mathcal{G}_{\epsilon_1} \subseteq \mathcal{G}_{\epsilon_2} \subseteq \cdots \subseteq \mathcal{G}_{\epsilon_\infty}$. Here, $\mathcal{G}_{\epsilon_i}$ is a graph representation of neurons, where the edges are connected to each other if they have a distance less than ϵ_i. We then use existing TDA tools to extract the 0D and 1D persistence diagrams of this filtration.

We are particularly interested in 1D topological structures, or cycles, in the computational graphs. This is because such structures capture a subset of neurons that are highly correlated with each other, meaning that they are often activated together. Remember that our activation vectors have been computed for N different inputs, and if N is large enough, such activation vectors should give us a fairly comprehensive overview of input traversal through neurons in a DNN. As such, finding a subset of neurons that are activated together (which could create a cycle in the topological features of computational graphs) can signal a shortcut path.

4 Case Studies

Now that we have established a framework for extracting the topological features of DNNs, in this section we show how the same framework is applicable to two different issues that both arise as a result of shortcut learning. We also present an extra case study on backdoor attacks in the Appendix [8].

4.1 Case Study I: Unlearnable Examples

4.1.1 Overview

Unlearnable examples [22], also known as availability attacks, are a family of data poisoning attacks [4] aiming to protect personal data from being exploited. With the unauthorized access of third-party crawlers over the web, users might lose control over how their personal data is used for training deep neural networks such as automated facial recognition [19, 18]. The goal of unlearnable examples is to passively prevent this misuse by adding an imperceptible perturbation to the user's data before sharing it. This perturbation should be powerful enough not to let any DNN learn meaningful patterns from its underlying data, but imperceptible enough so it does not interfere with the utility of the data to be shared with its audience [22].

There are various ways to construct such data-protecting perturbations [22, 49, 11, 12, 48, 35]. In general, the goal of all these different approaches is to find a set of perturbations $\boldsymbol{\delta}_i$ for each training data point $\boldsymbol{x}_i$ such that after optimizing the neural network weights over the perturbed data, the trained network yields a high error rate over the test set.[1] We can write this objective using our notations as:

$$\begin{aligned} &\underset{\{\boldsymbol{\delta}_i \,\mid\, i \in \mathrm{train}\}}{\arg\max} \sum_{j \in \mathrm{test}} \mathcal{L}\left(f_{\boldsymbol{\theta}^*}(\boldsymbol{x}_j), y_j\right) \\ &\qquad \text{s.t. } \boldsymbol{\theta}^* = \underset{\boldsymbol{\theta}}{\arg\min} \sum_{i \in \mathrm{train}} \mathcal{L}\left(f_{\boldsymbol{\theta}}\left(\boldsymbol{x}_i + \boldsymbol{\delta}_i\right), y_i\right). \end{aligned} \tag{4}$$

[1] Note that the data protector might opt to protect only a subset of all the training data. In this case, we can simply set $\boldsymbol{\delta}_i = \boldsymbol{0}$ for data samples that are not going to be protected.

Different methods opt to achieve this goal from different perspectives. One approach that is closely related to our analysis is presented in Yu et al. [48]. It is argued that the majority of existing unlearnable examples create shortcut paths within the neural network during training [48, 36]. As such, the perturbation generation process can be oversimplified, and even a set of linearly separable noisy data can act as unlearnable perturbations [48]. Therefore, since unlearnable examples are related to shortcut learning in neural networks, they can be an interesting case study for our analysis.

4.1.2 Experimental Evaluation

Settings: To empirically validate our speculations for the existence of a topological signature within DNNs trained over unlearnable datasets, we choose six state-of-the-art methods used for generating unlearnable perturbations. These approaches include Targeted Adversarial Poisoning (TAP) [11], Error-minimizing Noise (EMN) [22], Robust EMN (REM) [12], Neural Tangent Generalization Attacks (NTGA) [49], Shortcut (SHR) [14], and Autoregressive attacks (AR) [35]. Using these methods, we create unlearnable CIFAR-10 [24] datasets and train 70 ResNet-18 [17] models over each dataset independently. Furthermore, we also train 70 ResNet-18 models over the clean version of the CIFAR-10 dataset. We use the settings used in Huang et al. [22] for training our models.

Evaluation: After training each model until convergence, we use our framework outlined in Section 3 to extract topological features of the DNN computational graph. We use the `ripser` package [42] to obtain the 1D persistence diagram (PD) of the computational graph using a VR filtration. Then, we operate over this PD for a statistical analysis of the topological features. To this end, we use the average life-span/persistence of the PD (AVG PD_1) as well as the Wasserstein distance (WSD) between a pair of PDs as our evaluation metrics [41].

Findings: We can summarize our key findings as below:

- As shown in Figure 4, models trained on unlearnable datasets exhibit a different average persistence compared to clean models. The difference is statistically significant for all the tested unlearnable models, where the reported p-value is almost zero. This result demonstrates a clear distinguishing factor between benign and unlearnable models in their topological features, showing that **unlearnable datasets create alternative trajectories within the DNN for processing the inputs**. These paths have a higher life-span which indicates the over-confidence of the model in utilizing them.
- Glancing at the highly persistent cycles within each model as shown in Figure 5, we can see that **unlearnable models block the flow of information from the clean test inputs**. Therefore, the models trained over unlearnable datasets should indeed exhibit a lower accuracy compared to the clean model. This observation can be verified by looking at the feature maps of the ResNet-18 models for a sample input. As shown in Figure 6, for unlearnable models we do not see any trace of the input during the intermediate layers, while this information is easily transferred to deeper layers of the clean DNN.
- Sandoval-Segura et al. [36] argued that even unlearnable models capture useful features that can be utilized for a better classification. To this end, they used a simple linear classifier to train over the frozen feature extractor of such models with a small validation set. Kirichenko et al. [23] earlier showed that this approach is useful in combating spurious correlations. Using the same approach, we train a linear classifier head to obtain a linear probe accuracy (LP ACC) for all our models. We also compute the Wasserstein distance between the PD of each model and a randomly sampled clean model. As can be seen in Figure 7, **there is a correspondence between the power of the unlearnable dataset and its ability to create alternative trajectories within the DNN**. In particular, a higher Wasserstein distance indicates that the trajectories of the trained model differ more from a baseline clean model. Therefore, models with a higher Wasserstein distance tend to have a lower LP accuracy, which shows that the underlying unlearnable dataset is more resilient. For an extended version of our results, please see the Appendix [8].

4.2 Case Study II: Bias and Fairness

4.2.1 Overview

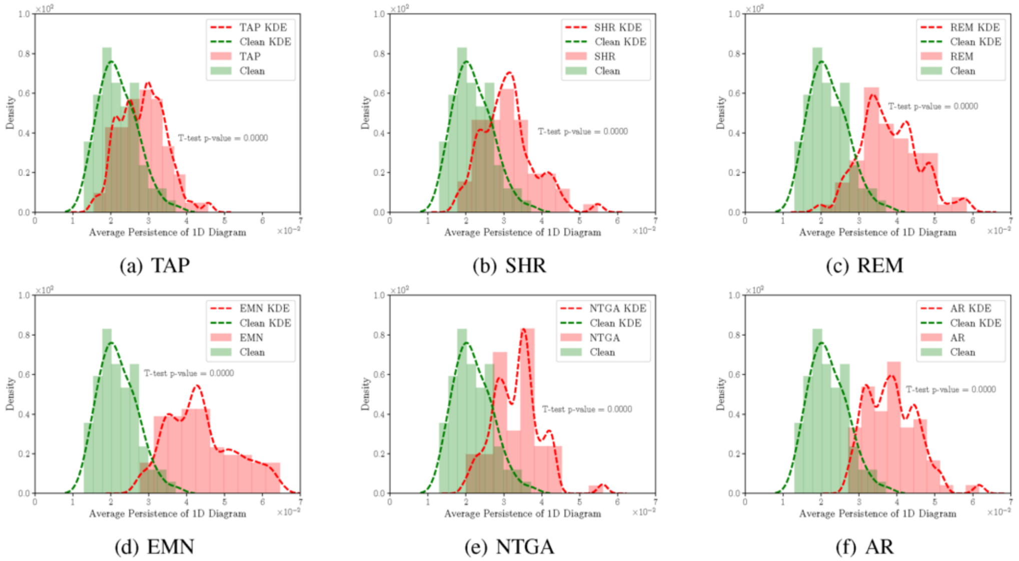
Another interesting case of shortcut learning happens during the training of biased neural networks [14]. It is widely known that bias usually takes place when a model focuses on one or a group of sensitive attributes to make its final prediction while ignoring the rest of input features [2]. Take the loan approval application as an example. In this case, bias happens when the model puts extra focus on a certain sensitive attribute, like ZIP code, to decide whether a loan request should be approved or not. Even though this notion is easy to comprehend, it is usually hard to track how the features are being processed within a machine learning model, especially when it comes to DNN models in the vision domain.

We can see traces of shortcut learning in the issue of bias and fairness in neural networks. Going back to our example of loan approvals, we can see that the ZIP code can act as a spurious feature: there exists a shortcut within the neural network which is activated when certain sensitive ZIP codes appear as the input. For these ZIP codes, the model ignores the rest of the input features and outputs a negative decision. Even though this issue is quite different from the previous case study, we demonstrate that similar observations can be made through the lens of persistent homology.

4.2.2 Experimental Evaluation

Settings: To conduct our empirical study, we use the state-of-the-art approach of Seo et al. [37] to learn unbiased ResNet-18 models over the CelebA dataset [30]. Using *gender* as our sensitive attribute, we train DNN models to predict various target attributes such as blonde hair, pale skin, etc. Naturally, these attributes have a skewed distribution depending on the subject's gender. For instance, the CelebA dataset contains a higher number of blonde females in comparison to males, making it a great dataset to study fairness in DNNs. For each target attribute, we train 10 regular (biased) and fair (unbiased) models. The only exception is the "Blonde Hair", for which we train 100 models.

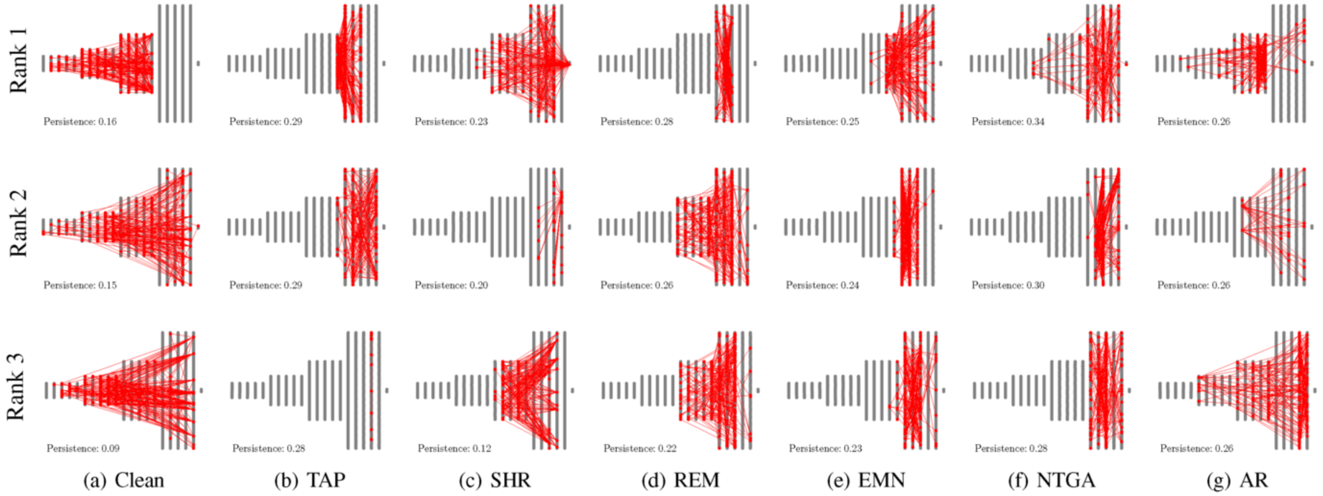
Evaluation: To quantitatively evaluate bias in each case, we use common performance metrics such as unbiased accuracy, worst group accuracy, standard deviation of unbiased accuracy over groups, Equalized Odds (EO), and Average Odds (AO) [2]. Moreover, we use the framework outlined in Section 3 to calculate the topological features of each model. In particular, we compute the 1D persistence diagram of the model and obtain the mean of the top-5 persistent features.



(a) TAP (b) SHR (c) REM

(d) EMN (e) NTGA (f) AR

Figure 4. The distribution of average persistence of 1D homology groups for ResNet-18 models trained with different versions of unlearnable CIFAR-10 datasets. Each histogram summarizes the result of 70 independent training runs for each dataset. The p-value of the T-test has also been shown in each figure.



(a) Clean (b) TAP (c) SHR (d) REM (e) EMN (f) NTGA (g) AR

Figure 5. Top-3 persistent cycles for one instance of ResNet-18 models trained with different versions of unlearnable CIFAR-10 datasets. Each node denotes a single neuron within the model. Note that due to the high number of neurons, we show a downsampled version of the original model.

Findings: We outline our key findings as below:

- As shown in Figure 8, **biased models exhibit a higher 1D persistence compared to unbiased models**. This has been likely caused by the discrepancies in the shortcut trajectories that biased models create within the DNN. Again, this difference is statistically significant to distinguish unbiased from biased models.
- We can see from Figure 9 that **models with higher top-5 persistence are more likely to be biased**. This is also evident from the worst group accuracy of these models which are significantly lower than unbiased ones.
- In Table 1, we provide a quantitative comparison between biased and unbiased models for 10 target attributes from the CelebA dataset. As seen, all **unbiased models have a lower top-5 persistence compared to their biased counterparts**. Interestingly, a lower top-5 persistence signals a lower standard deviation for unbiased accuracy across sensitive groups. Therefore, **there exists a topological signature within DNN models that indicates whether they are biased or unbiased**.

5 Conclusion and Future Directions

We demonstrated that the trajectories passed by inputs through neural networks contain invaluable insights about shortcuts within the model. To this end, we used two novel cases of shortcut learning in DNNs: unlearnable examples and fairness. We saw that persistent homology can play an important role in revealing these failure cases within a neural network. Interestingly, we showed that these differences are statistically significant.

While we used these two specific case studies, the application of persistent homology to such failure cases is not limited to these cases only. Previously, PH-based solutions have been independently discovered for other failures of neural networks related to shortcut learning, such as backdoor attacks [50], adversarial examples [13, 15], and out-of-distribution detection [25]. In this paper, we aimed to argue that all such solutions can be combined to mitigate shortcut learning at a broader level. This is due to the fact that the cause of most issues with DNNs lie within shortcut learning, which could be traced by

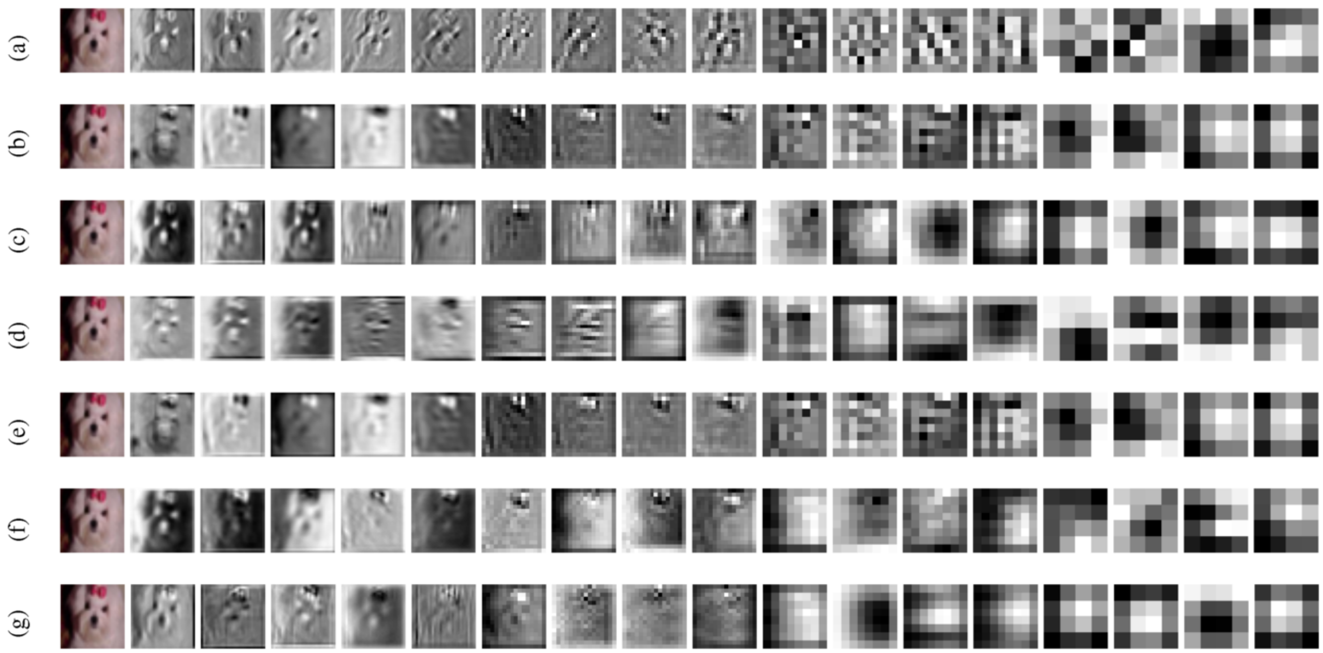
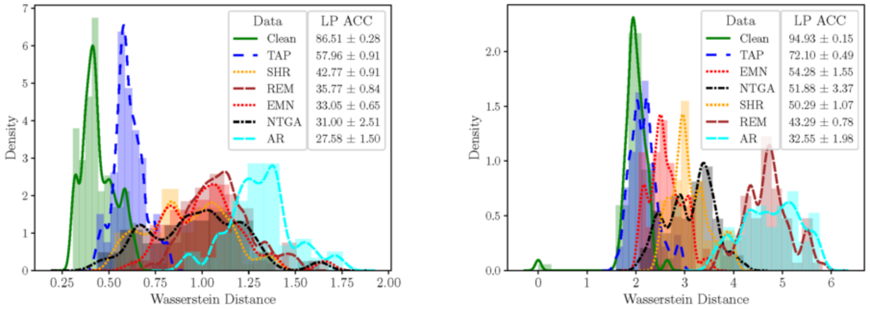


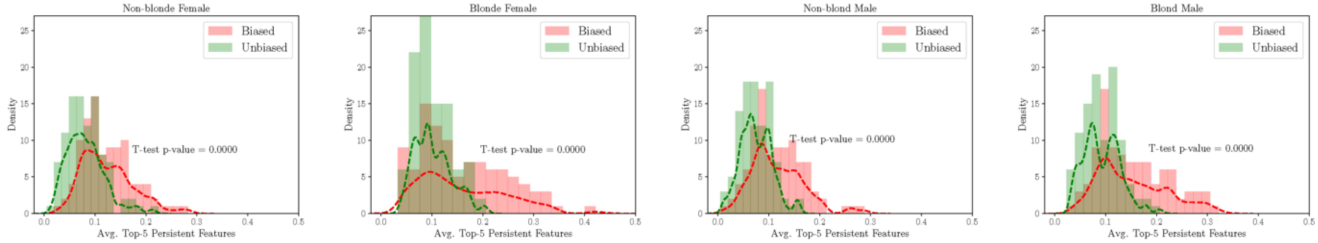
Figure 6. Feature space representation of one instance of ResNet-18 models trained with different versions of unlearnable CIFAR-10 datasets. The features are shown from the input space (leftmost) to the last layer of the classifier (rightmost). The models are trained using (a) clean, (b) TAP [11], (c) SHR [48], (d) REM [12] (e) EMN [22], (f) NTGA [49], and (g) AR [35] data.



(a) ResNet-18

(b) DenseNet-121

Figure 7. The Wasserstein distance between 1D persistence diagrams of clean versus unlearnable (a) ResNet-18 and (b) DenseNet-121 models. For clean models, we compute the distance between two randomly selected models, while for unlearnable models we use a clean baseline.



(a) Non-blonde Female

(b) Blonde Female

(c) Non-blond Male

(d) Blond Male

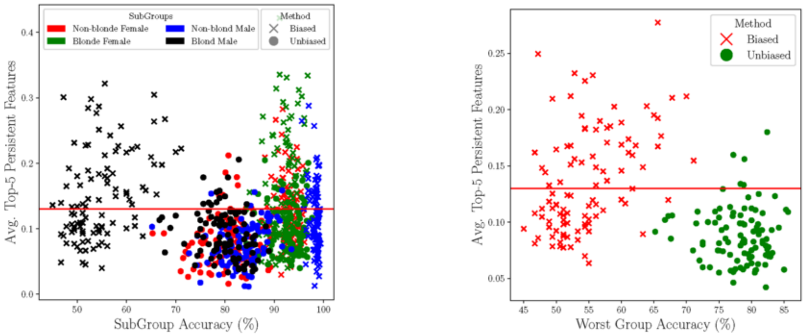
Figure 8. The distribution of average top-5 persistence of 1D homology groups for ResNet-18 models trained with biased and unbiased objectives. Each sub-figure represent a different protected group. The p-value of the T-test has also been shown in each figure.

exploring the computational graphs trajectories. Below, we highlight interesting and challenging future directions:

- **Using Persistent Homology to Create Powerful Unlearnable Datasets:** As we see in Figure 7, the most resilient unlearnable examples are those that deviate the most from the usual trajectories of clean models. Proposing a novel unlearnable example generation objective that reflects such a phenomenon directly could lead to better unlearnable examples.
- **Proposing a New Fairness Measure based on Persistent Homology:** As we saw in Table 1, PH-based measures such as the top-5 persistent features have a tendency towards revealing bias

Table 1. Performance of ResNet-18 models trained over CelebA dataset with different target attributes. The sensitive attribute for all models is gender. In each case, we train a regular and an unbiased model using [37]. The results are averaged over 10 models.

Target Attribute	Model	Performance Metrics					
		Unbiased Acc (%)	Worst Group Acc (%)	Unbiased Acc Std. (%)	EO Disparity (%)	Average Odds (%)	Top-5 1D Pers. ($\times 10^{-1}$)
Blonde Hair	Biased	84.50 ± 1.35	54.81 ± 5.76	17.39 ± 2.56	37.13 ± 4.47	21.31 ± 2.23	1.38 ± 0.45
	Unbiased	84.08 ± 2.26	78.20 ± 4.17	5.84 ± 2.31	12.89 ± 5.83	8.83 ± 4.39	0.88 ± 0.24
Chubby	Biased	67.22 ± 1.73	22.24 ± 1.64	31.35 ± 1.81	32.13 ± 7.28	19.50 ± 4.59	1.06 ± 0.41
	Unbiased	73.42 ± 2.24	61.53 ± 5.01	11.39 ± 2.85	19.53 ± 6.88	22.11 ± 6.28	0.70 ± 0.22
Double Chin	Biased	66.76 ± 2.96	21.02 ± 6.04	32.04 ± 3.55	32.10 ± 6.39	19.19 ± 4.06	1.05 ± 0.27
	Unbiased	75.81 ± 1.49	65.58 ± 3.97	8.86 ± 2.76	14.65 ± 5.84	17.16 ± 5.66	0.74 ± 0.18
Heavy Makeup	Biased	74.58 ± 1.36	49.29 ± 4.05	22.05 ± 1.66	43.71 ± 5.29	43.59 ± 3.39	1.13 ± 0.27
	Unbiased	74.69 ± 1.93	56.81 ± 2.35	15.62 ± 1.92	23.87 ± 6.02	29.47 ± 4.89	0.59 ± 0.19
Oval Face	Biased	60.32 ± 1.20	20.03 ± 5.55	28.42 ± 4.38	29.51 ± 9.60	20.49 ± 8.19	0.77 ± 0.23
	Unbiased	56.51 ± 3.97	48.84 ± 2.20	6.58 ± 2.37	11.24 ± 7.27	9.39 ± 6.29	0.66 ± 0.40
Pale Skin	Biased	80.49 ± 2.03	55.91 ± 5.39	17.83 ± 2.69	15.34 ± 3.84	8.65 ± 1.92	1.15 ± 0.17
	Unbiased	86.96 ± 0.83	83.37 ± 1.61	2.99 ± 0.84	5.47 ± 2.73	4.94 ± 2.14	0.81 ± 0.19
Pointy Nose	Biased	63.96 ± 0.98	31.87 ± 4.24	22.12 ± 2.69	27.17 ± 3.82	22.79 ± 3.84	1.12 ± 0.41
	Unbiased	62.27 ± 3.09	48.13 ± 6.07	12.69 ± 3.29	22.32 ± 8.46	24.15 ± 7.24	0.50 ± 0.19
Straight Hair	Biased	70.04 ± 1.46	52.58 ± 5.27	14.75 ± 4.82	7.18 ± 6.37	6.54 ± 3.02	0.88 ± 0.31
	Unbiased	66.92 ± 1.84	57.43 ± 3.76	8.73 ± 2.96	12.05 ± 6.28	14.77 ± 6.31	0.65 ± 0.39
Wearing Lipstick	Biased	77.79 ± 0.64	57.05 ± 1.79	18.73 ± 0.87	33.87 ± 2.74	37.20 ± 1.78	1.01 ± 0.21
	Unbiased	73.59 ± 2.74	57.83 ± 7.36	14.82 ± 4.04	24.95 ± 11.55	28.47 ± 9.16	0.83 ± 0.19
Young	Biased	77.21 ± 0.58	55.16 ± 3.58	14.09 ± 1.91	12.59 ± 2.83	18.29 ± 2.74	0.95 ± 0.29
	Unbiased	71.20 ± 1.75	62.67 ± 5.69	7.42 ± 4.12	14.42 ± 9.57	13.04 ± 9.02	0.89 ± 0.27



(a) (b)

Figure 9. The scatter-plot of average top-5 persistence of 1D homology groups against (a) sub-group accuracy and (b) worst-group accuracy. The red line indicates a preset threshold that can separate the biased from unbiased models.

in DNN models. However, these raw measures are not calibrated: even though their relative magnitude could be an indication of bias, we cannot directly compare different models based on this measure. Introducing a new calibrated measure of fairness using persistent homology could be helpful in a better, more universal quantification of fairness in DNN models.

- **Incorporating Topological Measures in Decision-Making:** In a broader sense, quantification of uncertainty using topological measures could become the cornerstone of decision-making using DNNs. This is because apparently normal models should guide the inputs in certain trajectories within the network, and as such, proposing a new measure that covers this could help us detect issues within a neural network in a straightforward manner. Such solutions have also the potential of explainability, as they could reveal the paths involved with a given input during inference.
- **Introducing a Universal Regularizer for DNN Training:** Once we can quantify a universal measure for quantification of uncertainty, we could potentially use these measures to enforce certain behavior within the neural network computational graph. Unfortunately, existing tools for integrating topological measures during neural network training lag behind their counterparts in terms of their computational speed [20], rendering it challenging to use for large-scale models.
- **Testing Solutions under Different Shortcut Learning Scenarios:** Lastly, we encourage researchers in this area to test their solutions not only for a specific type of shortcuts, but for a broad range of them. The creation of a standard benchmark along this line could help researchers to think about the broader issue of shortcut learning rather than focusing on a specific use case.

Acknowledgements

Sarah Erfani is in part supported by Australian Research Council (ARC) Discovery Early Career Researcher Award (DECRA) DE220100680. Moreover, this research was partially supported by the ARC Centre of Excellence for Automated Decision-Making and Society (CE200100005), and funded partially by the Australian Government through the Australian Research Council.

References

[1] S. Barannikov. The framed morse complex and its invariants. *Advances in Soviet Mathematics*, 21:93–116, 1994.

[2] S. Barocas, M. Hardt, and A. Narayanan. *Fairness and Machine Learning: Limitations and Opportunities*. MIT Press, Cambridge, Massachusetts, United States, 2023.

[3] S. Beery, G. V. Horn, and P. Perona. Recognition in terra incognita. In *European Conference on Computer Vision (ECCV)*, pages 472–489, 2018.

[4] B. Biggio and F. Roli. Wild patterns: Ten years after the rise of adversarial machine learning. *Pattern Recognition*, 84:317–331, 2018.

[5] J. Buolamwini and T. Gebru. Gender shades: Intersectional accuracy disparities in commercial gender classification. In *Proceedings of the Conference on Fairness, Accountability and Transparency (FAccT)*, 2018.

[6] G. Carlsson. Topology and data. *Bulletin of the American Mathematical Society (AMS)*, 46(2):255–308, 2009.

[7] X. Chen, C. Liu, B. Li, K. Lu, and D. Song. Targeted backdoor attacks on deep learning systems using data poisoning. *CoRR*, abs/1712.05526, 2017.

[8] H. M. Dolatabadi, S. M. Erfani, and C. Leckie. Be persistent: Towards a unified solution for mitigating shortcuts in deep learning. *CoRR*, abs/2402.11237, 2024. Full version of this paper.

[9] H. Edelsbrunner and J. Harer. Persistent homology-a survey. *Contemporary mathematics*, 453(26):257–282, 2008.

[10] H. Edelsbrunner and J. Harer. *Computational Topology - an Introduction*. American Mathematical Society, Providence, Rhode Island, United States, 2010.

[11] L. Fowl, M. Goldblum, P. Chiang, J. Geiping, W. Czaja, and T. Goldstein. Adversarial examples make strong poisons. In *Proceedings of the Advances in Neural Information Processing Systems (NeurIPS)*, pages 30339–30351, 2021.

[12] S. Fu, F. He, Y. Liu, L. Shen, and D. Tao. Robust unlearnable examples: Protecting data privacy against adversarial learning. In *Proceedings of the 10th International Conference on Learning Representations (ICLR)*, 2022.

[13] T. Gebhart and P. Schrater. Adversary detection in neural networks via persistent homology. *CoRR*, abs/1711.10056, 2017.

[14] R. Geirhos, J. Jacobsen, C. Michaelis, R. S. Zemel, W. Brendel, M. Bethge, and F. A. Wichmann. Shortcut learning in deep neural networks. *Nature Machine Intelligence*, 2(11):665–673, 2020.

[15] M. Goibert, T. Ricatte, and E. Dohmatob. An adversarial robustness perspective on the topology of neural networks. In *Proceedings of the Machine Learning Safety Workshop, Conference on Neural Information Processing Systems (NeurIPS)*, 2022.

[16] T. Gu, B. Dolan-Gavitt, and S. Garg. BadNets: Identifying vulnerabilities in the machine learning model supply chain. *CoRR*, abs/1708.06733, 2017.

[17] K. He, X. Zhang, S. Ren, and J. Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 770–778, 2016.

[18] K. Hill. The secretive company that might end privacy as we know it. *The New York Times*, 2020.

[19] K. Hill and A. Krolik. How photos of your kids are powering surveillance technology. *The New York Times*, 2019.

[20] C. D. Hofer, F. Graf, M. Niethammer, and R. Kwitt. Topologically densified distributions. In *Proceedings of the 37th International Conference on Machine Learning (ICML)*, pages 4304–4313, 2020.

[21] G. Huang, Z. Liu, L. van der Maaten, and K. Q. Weinberger. Densely connected convolutional networks. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 2261–2269, 2017.

[22] H. Huang, X. Ma, S. M. Erfani, J. Bailey, and Y. Wang. Unlearnable examples: Making personal data unexploitable. In *Proceedings of the 9th International Conference on Learning Representations (ICLR)*, 2021.

[23] P. Kirichenko, P. Izmailov, and A. G. Wilson. Last layer re-training is sufficient for robustness to spurious correlations. In *The 11th International Conference on Learning Representations (ICLR)*, 2023.

[24] A. Krizhevsky and G. Hinton. Learning multiple layers of features from tiny images. Master's thesis, Department of Computer Science, University of Toronto, 2009.

[25] T. Lacombe, Y. Ike, M. Carrière, F. Chazal, M. Glisse, and Y. Umeda. Topological uncertainty: Monitoring trained neural networks through persistence of activation graphs. In *Proceedings of the 13th International Joint Conference on Artificial Intelligence (IJCAI)*, pages 2666–2672, 2021.

[26] Y. Li, Y. Li, B. Wu, L. Li, R. He, and S. Lyu. Invisible backdoor attack with sample-specific triggers. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 16443–16452, 2021.

[27] Y. Li, T. Zhai, Y. Jiang, Z. Li, and S. Xia. Backdoor attack in the physical world. *CoRR*, abs/2104.02361, 2021.

[28] Y. Li, M. Ya, Y. Bai, Y. Jiang, and S.-T. Xia. BackdoorBox: A python toolbox for backdoor learning. In *Proceedings of the ICLR Workshop on Backdoor Attacks and Defenses in Machine Learning*, 2023.

[29] Y. Liu, X. Ma, J. Bailey, and F. Lu. Reflection backdoor: A natural backdoor attack on deep neural networks. In *Proceedings of the 16th European Conference on Computer Vision (ECCV)*, 2020.

[30] Z. Liu, P. Luo, X. Wang, and X. Tang. Deep learning face attributes in the wild. In *Proceedings of International Conference on Computer Vision (ICCV)*, 2015.

[31] T. A. Nguyen and A. T. Tran. Input-aware dynamic backdoor attack. In *Proceedings of the Advances in Neural Information Processing Systems (NeurIPS)*, 2020.

[32] T. A. Nguyen and A. T. Tran. WaNet - imperceptible warping-based backdoor attack. In *Proceedings of the 9th International Conference on Learning Representations (ICLR)*, 2021.

[33] A. Radford, K. Narasimhan, T. Salimans, I. Sutskever, et al. Improving language understanding by generative pre-training. 2018.

[34] O. Russakovsky, J. Deng, H. Su, J. Krause, S. Satheesh, S. Ma, Z. Huang, A. Karpathy, A. Khosla, M. S. Bernstein, A. C. Berg, and F. Li. ImageNet large scale visual recognition challenge. *International Journal of Computer Vision (IJCV)*, 115(3):211–252, 2015.

[35] P. Sandoval-Segura, V. Singla, J. Geiping, M. Goldblum, T. Goldstein, and D. W. Jacobs. Autoregressive perturbations for data poisoning. In *Proceedings of the Advances in Neural Information Processing Systems (NeurIPS)*, 2022.

[36] P. Sandoval-Segura, V. Singla, J. Geiping, M. Goldblum, and T. Goldstein. What can we learn from unlearnable datasets? In *Proceedings of the Advances in Neural Information Processing Systems (NeurIPS)*, 2023.

[37] S. Seo, J. Lee, and B. Han. Unsupervised learning of debiased representations with pseudo-attributes. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 16721–16730, 2022.

[38] S. Singla and S. Feizi. Salient ImageNet: How to discover spurious features in deep learning? In *The 10th International Conference on Learning Representations (ICLR)*, 2022.

[39] Y. Song and S. Ermon. Generative modeling by estimating gradients of the data distribution. In *Proceedings of the Advances in Neural Information Processing Systems (NeurIPS)*, pages 11895–11907, 2019.

[40] C. Szegedy, W. Zaremba, I. Sutskever, J. Bruna, D. Erhan, I. J. Goodfellow, and R. Fergus. Intriguing properties of neural networks. In *Proceedings of the 2nd International Conference on Learning Representations (ICLR)*, 2014.

[41] The GUDHI Project. *GUDHI User and Reference Manual*. GUDHI Editorial Board, 2015. URL http://gudhi.gforge.inria.fr/doc/latest/.

[42] C. Tralie, N. Saul, and R. Bar-On. Ripser.py: A lean persistent homology library for python. *The Journal of Open Source Software*, 3(29):925, Sep 2018.

[43] A. Turner, D. Tsipras, and A. Madry. Label-consistent backdoor attacks. *CoRR*, abs/1912.02771, 2019.

[44] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, L. Kaiser, and I. Polosukhin. Attention is all you need. In *Proceedings of the Advances in Neural Information Processing Systems (NeurIPS)*, pages 5998–6008, 2017.

[45] L. Vietoris. Über den höheren zusammenhang kompakter räume und eine klasse von zusammenhangstreuen abbildungen. *Mathematische Annalen*, 97(1):454–472, 1927.

[46] M. Wang and W. Deng. Deep visual domain adaptation: A survey. *Neurocomputing*, 312:135–153, 2018.

[47] R. Wexler. When a computer program keeps you in jail: How computers are harming criminal justice. *New York Times*, 13, 2017.

[48] D. Yu, H. Zhang, W. Chen, J. Yin, and T. Liu. Availability attacks create shortcuts. In *Proceedings of the 28th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, pages 2367–2376, 2022.

[49] C. Yuan and S. Wu. Neural tangent generalization attacks. In *Proceedings of the 38th International Conference on Machine Learning (ICML)*, pages 12230–12240, 2021.

[50] S. Zheng, Y. Zhang, H. Wagner, M. Goswami, and C. Chen. Topological detection of trojaned neural networks. In *Proceedings of the Advances in Neural Information Processing Systems (NeurIPS)*, pages 17258–17272, 2021.

[51] G. K. Zipf. *Human behavior and the principle of least effort*. Addison-Wesley Press, Boston, Massachusetts, United States, 1949.