normalized version of their Betti curves $b_k(t) = |\{(b,d) \in D(\mathbb{V}_k(\mathrm{VR}(P,d))) : t \in [b,d)\}|$ to study the evolution of the topology during training again.

During the first epochs, it was observed that the highest values of these curves moved from the left part of the domain to the right for $k \in [3]$. However, when the training entered into the overfitting regime, the maximum values of the normalized Betti curves moved again to the left part. Due to the behaviour of the normalized Betti curves, Corneanu et al. proposed an early stopping of the training whenever it started moving the maximum values of Betti curves to the left again. Also, Corneanu et al. used normalized Betti curves to detect sets of adversarial examples, if any.

The observed evolution of Betti curves suggests that the topological correlation structure of the activations of neural networks is highly related with their capacity to generalize to the whole data distribution. This was further studied by Corneanu et al. (2020) and Ballester et al. (2023b), who used persistence diagrams $D(\mathbb{V}_k(\mathrm{VR}(P,d)))$ of dimensions $k \in \{0,1\}$ to predict the generalization gap, that is, the difference between training and test accuracies.

First, Corneanu et al. proposed to predict the generalization gap by performing a linear regression with independent variables chosen to be two persistence summaries, namely average persistence λ and average midlife μ, given by

$$\lambda(D) = \frac{1}{|D|} \sum_{(b,d)\in D} d - b, \qquad \mu(D) = \frac{1}{|D|} \sum_{(b,d)\in D} \frac{d+b}{2},$$

respectively. This approach seemed to work on controlled computer vision scenarios. Following these results, Ballester et al. (2023b) extended previous work by studying more networks with different generalization gaps extracted from the first and second tasks of the *Predicting Generalization in Deep Learning* NeurIPS challenge (Jiang et al., 2020a). In this study, more persistence summaries were tested, such as persistent entropy (Atienza et al., 2020), persistence pooling vectors (Bonis et al., 2016), and complex polynomials (Di Fabio and Ferri, 2015), among others.

The problem with the aforementioned approach is that it does not scale properly for the more complex networks analyzed in Ballester et al. (2023b). Due to the high number of neurons in larger neural networks and the high number of training examples in the datasets, computing persistence diagrams for the whole set of neurons and training examples was unfeasible. To alleviate this problem, Ballester et al. proposed to generate multiple persistence diagrams from uniform samples of the dataset and from neuron samples using a probability distribution where neurons with higher activations in absolute value had a higher chance of being sampled than neurons with lower activations. From the corresponding persistence diagrams, persistence summaries are computed and then bootstrapped to perform linear regression to predict the generalization gap.

The best persistence summaries to predict generalization gaps in this line of work were a combination of statistical measures —such as averages and standard deviations— of the points in persistence diagrams. In particular, fixing some sources of variability of the networks affecting their depth, Ballester et al. found a correlation between the averages and standard deviations of the second coordinates of zero- and one-dimensional persistence diagram points and the generalization gap of their associated networks. These insights and methods were further studied and used in a follow-up article by Ballester et al. (2023a) to develop regularization terms that minimize pairwise correlations between neurons.

A very similar approach was used to detect trojaned neural networks. A brief explanation of trojaned neural networks can be found in Section 2.4.4. Zheng et al. (2021) propose to build zero- and one-dimensional Vietoris–Rips persistence modules from activations as in the previous works but replacing the Pearson correlation coefficient in absolute value by a more general correlation coefficient measure. For such persistence modules, Zheng et al. found a significant difference between clean and trojaned neural network persistence diagrams for both synthetic and real-world experiments, where the real-world experiments comprised training 70 ResNet18 using a clean MNIST dataset and another 70 with a trojaned MNIST counterpart. The average death time of zero persistence diagrams was particularly significant for segregating persistence diagrams from clean and trojaned neural networks, where averages were significantly lower than those of clean models. Also, by manual inspection of cycle representatives of the points of persistence diagrams, it was observed that trojaned models contained edges connecting shallow and deep layers, a phenomenon that did not happen for clean models. With this in mind, Zheng et al. trained basic FCFNN models to detect trojaned convolutional neural networks trained on MNIST, CIFAR10 and the IARPA/NIST TrojAI competition (of Standards and Technology) datasets using persistence summaries from the persistence diagrams of the networks as input. These models resulted in higher performance than those for other state-of-the-art trojan detection algorithms and models.

Activations for each layer

While in previous works the topology of the activations was studied globally, Naitzat et al. (2020) studied the evolution of the activations layer by layer in FCFNNs for binary classification problems in an extensive set of experiments. For data samples $\mathcal{D}_i$ from one of the classes $i \in [2]$ at a time, the evolution of the (persistent) homology of $\mathcal{D}_i$ through the different layers is analyzed. In particular, for each example $x \in \mathcal{D}_i$ and each layer l in an FCFNN $\mathcal{N}$, an activation vector a_x^l is computed by taking the vector composed of the different activation values of the neurons given the example x in layer l, that is,

$$a_x^l = \left(\phi_{\mathcal{N}}^{(l)}(x)_1, \ldots, \phi_{\mathcal{N}}^{(l)}(x)_{N_l}\right).$$

Then, persistent homology is computed from Vietoris–Rips filtrations of the point clouds $\mathcal{D}_i^l = \left\{a_x^l : x \in \mathcal{D}_i\right\}$ for each layer l.

The experiments were divided into two different scenarios: synthetic and real datasets. For the first scenario, the datasets were samples from spaces with known topology. For the second scenario, the datasets were simplified and binarized classification versions of several known datasets, such as MNIST. In both cases, FCFNNs were trained with different architectures, different training parameters, and almost zero training error.

For the synthetic datasets, the evolution of Betti numbers was studied for different dimensions k of Vietoris–Rips complexes of fixed value t. The dissimilarity for these complexes was chosen to be the graph geodesic distance on the k-nearest neighbors graph applied to the point clouds $\mathcal{D}_i^l$. For a fixed sample $\mathcal{D}_i$ and a fixed neural network $\mathcal{N}$, the parameters t and k were selected in such a way that $b_0(\mathrm{VR}_t(\mathcal{D}_i))$, $b_1(\mathrm{VR}_t(\mathcal{D}_i))$, and $b_2(\mathrm{VR}_t(\mathcal{D}_i))$ were equal to the first three (known) Betti numbers of the space from which the dataset $\mathcal{D}_i$ was sampled. However, due to the difficulty in selecting good parameters t and k for the

real data, persistence diagrams of the activations for each layer were instead studied in the second scenario.

For the synthetic scenario, Naitzat et al. observed a decay of the different Betti numbers b_k through the layers. This decay was slower through different experiments for completely smooth functions than for ReLU-like (ReLU and leaky ReLU) activation functions for dimension zero. The number of layers needed to perform this topological simplification varied according to the topology of the initial data. Narrow layers, that is, those with few neurons per layer, appeared to simplify the topology faster than their wider counterparts. Bottleneck architectures, i.e., architectures with decreasing numbers of neurons per layer, seemed to force larger topological changes in the data than networks with constant number of neurons per layer. Also, depth did not seem to influence the way the topological simplifications are distributed across the layers: initial layers did not seem to perform many topological simplifications, in general. Thus, reducing depth simply concentrated the topological simplifications in the last layers. This topological simplification was also observed in the persistent diagrams of the activations for the real datasets in simpler networks than the ones used for the synthetic scenario, where the number of points and their persistence across all dimensions diminished through layers.

The previous observations were contradicted by Wheeler et al. (2021). In this work, evolution of the topology of the activations through the different layers of FCFNNs for synthetic and real data was studied again. However, this time, the activations and distances between them were preprocessed and slightly modified depending on the experiments performed to build a point cloud. From such preprocessed point clouds, Vietoris–Rips persistence diagrams were computed using the usual Euclidean distance between vectors. Finally, instead of studying Betti numbers or persistence diagrams directly, Wheeler et al. computed persistence landscapes $\lambda_k^l = \left(\lambda_{k,1}^l, \lambda_{k,2}^l, \dots\right)$ derived from the persistence diagrams for each layer for different homological degrees k.

In the previous work, Betti numbers were used to quantify the *topological complexity* of activations in a given layer. Persistence landscapes also allow one to define topological complexities over point clouds. Specifically, given a persistence landscape λ coming from the point cloud, one way to measure its complexity is to measure the area under its curves λ_i. The sum of these areas defines an inner product

$$\left\langle \lambda, \lambda' \right\rangle = \sum_{i=1}^{\infty} \int_{-\infty}^{\infty} \lambda_i(t) \lambda_i'(t)\, dt,$$

that induces a norm $\|\lambda\| = \sqrt{\langle \lambda, \lambda \rangle}$. The higher the norm, the higher the topological complexity of the point cloud. This topological complexity measure was the one used by Wheeler et al. to measure the topological evolution of the activations computing it layer by layer, fixing a homological degree k.

The results for synthetic data were similar to the previous ones: for FCFNNs with 11 layers with ReLU activation functions except for the output one, trained to perfect accuracy with 100 different initialization weights, topological complexities were observed to decay through the last layers, where the best weight configurations decayed faster on average than their worse counterparts. However, for the first layers, the topological complexities increased layer by layer. More strikingly, for FCFNNs trained to near-perfect accuracy with