not necessarily squared, as a Wasserstein dissimilarity matrix between the Wasserstein barycenters of the latent variables and of the factored real data. Finally, the supervised metric is calculated as in Equation (14) for the leading principal submatrix of M', generated from M as before, taking the first c rows and columns.

This metric was compared to other state-of-the-art disentanglement measures on three different datasets, dSprites (Matthey et al., 2017), a dataset to specifically test disentanglement of generative models, CelebA, and Celeba-HQ (Karras et al., 2018), for ten different architectures including the aforementioned VAE and WGAN-GP, among others. In particular, it was shown that both, the unsupervised and the supervised proposed disentanglement metrics ranked models similarly to the other reference disentanglement metrics.

3.3 Internal representations and activations

Most of the research presented in this survey belongs to this section. Here, we mainly focus on neural network weights and activations. Weights $\theta(\mathcal{N})$ are arguably one of the most important parts of neural networks, as they determine their functions alongside architectures $a(\mathcal{N})$. Therefore, selecting the appropriate weights when training is key to ensuring that neural networks perform effectively. On the other hand, activations are the result of the computations performed by the neural network $\mathcal{N}$ on the input data $x \in \mathcal{X}$, and therefore are key to understand the behavior of neural networks and their properties.

The use of TDA to analyze weights and activations is mainly divided into two different approaches. The first one, more qualitatively, is the use of Mapper, sometimes supported by some (persistent) homological computations, to understand the structure of the internal representations of neural networks. The second, more quantitatively, is the use of (persistent) homology to extract topological features of the internal representations of neural networks and link them with different properties of the networks, such as their generalization capabilities.

3.3.1 Mapper

This subsection is organized into two categories: first, those articles that apply Mapper to point clouds derived from weights, and second, those that utilize Mapper on point clouds resulting from activations.

Mapper on weights

Two seminal papers by Carlsson and Brüel Gabrielsson published in 2019 and a follow-up in 2020, stand as early explorations of the use of Mapper to analyze internal representations of FCFNNs, specifically CNNs. These papers focus on the topology and distribution of convolutional filter weights within the same layer of CNNs.

Consider a CNN with architecture a. Let i be a convolutional layer containing filters $C_1^{(i)}, \ldots, C_{c^{(i)}}^{(i)}$ each of dimensions $h \times w \times c^{(i-1)}$. Here, $h \times w$ denotes the size of the convolution, $c^{(i-1)}$ denotes the number of channels in layer $i-1$, and $c^{(i)}$ denotes the number of channels in layer i after convolution. Given t independently trained instances of this architecture, Mapper graphs are generated for a subset of the $t \cdot c^{(i-1)} \cdot c^{(i)}$ vectors of dimension $h \times w$ derived from the convolutions across all $c^{(i-1)}$ channels and $c^{(i)}$ filters.

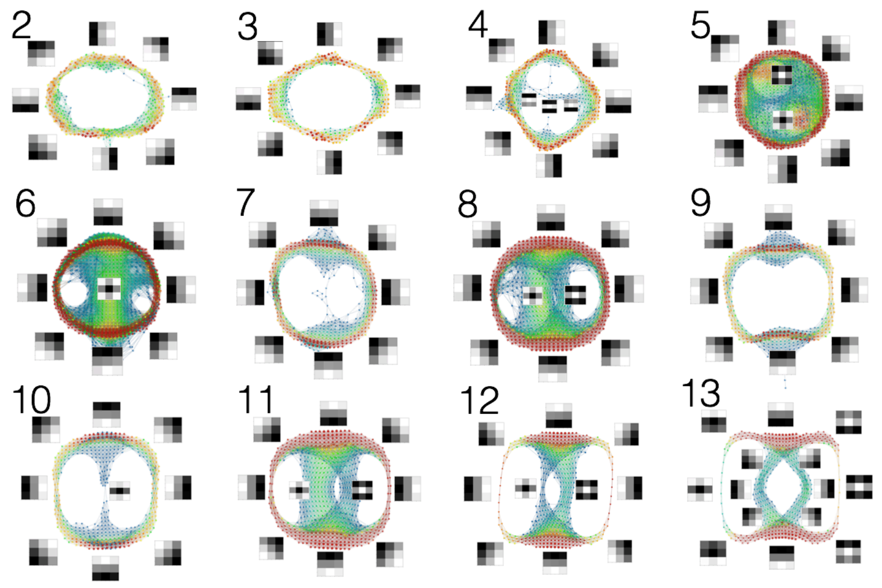

Figure 7: Mapper graphs reproduced with permission from Brüel Gabrielsson and Carlsson (2019) of the convolutional filter weights from layer 2 to layer 13 of a trained VGG-16 neural network. Vertex colors represent the size of the collection represented by each node, increasing from blue to red. Matrices surrounding the Mapper graph are averages of filter weights located at nodes near the matrix, representing filters of the corresponding area of the graph.

In simple CNN architectures, comprising two convolutions, two pooling layers, and a fully connected output layer trained on the MNIST dataset, the Mapper graphs of the first convolutional layer closely resembled topological models of 3×3 patches of natural images that were previously identified by Carlsson et al. (2008). This suggests that CNNs may capture inherent topological features of data during learning. Similar experiments were carried out on the CIFAR-10 dataset using architectures similar to those used for MNIST. Here, the results varied: some Mapper graphs resembled previously discovered topological models, while others exhibited new Mapper configurations. Mapper graphs of more advanced architectures VGG-16 and VGG-19, pretrained on ImageNet, were also analyzed. For example, in VGG-16, all convolutional layers except the first and thirteenth ones displayed a circle as the dominant topological structure, consistent with findings in natural image patches. Also, it was observed that in the first layers of the network, the topological structures were simpler than in the other ones. Figure 7 displays the Mapper graphs for all convolutional layers in VGG-16, except the first one.

These Mapper graphs were related to generalization. On the one hand, more powerful networks appeared to learn simpler topological structures in the first layers, as seen in VGG-16, suggesting a possible correlation between the topological complexity of the network's weights and its generalization capabilities. On the other hand, constraints on the parameter space of neural networks based on previously identified weight topological structures improved network generalization. These topological structures also helped preprocess the input data to obtain an increase in accuracy without modifying the network.

Building on the previous insights and the success of convolutional neural networks, Carlsson and Brüel Gabrielsson proposed three ways to generalize convolutions to layers that take into account the topology of the data. This is continued in Love et al. (2020), where topological CNNs, which encompass several topologically defined convolutional methods, are introduced.

Gabella (2021) proposed to use Mapper to study the evolution of weights through the training process for a fixed layer of a FCFNN. In this case, Mapper graphs were built using the `DBSCAN` clustering algorithm. In addition, two different filter functions were used: the l^2 norm and the projection to the three principal components of the Mapper input points. Given a FCFNN trained in n steps and a layer l with N_l neurons, Mapper simplicial complexes were computed with a point cloud of $n \cdot N_l$ points of dimension N_{l-1}, each point representing the weights of the N_{l-1} incident edges to a neuron of the layer l at a given step of the training process.

In a first experiment, Gabella trained an FCFNN in MNIST with only one hidden layer of 100 neurons, setting the bias values to zero and initializing the weights to zero. Using the l^2 function, the Mapper graph produced with the l^2 function and the output layer had ten different branches, the same number as the number of classes in the dataset. For the hidden layer, only twelve distinct branches were produced, much less than the number of neurons. This suggests that the number of branches in this Mapper graph configuration could capture the *real expressiveness* of a neural network layer —in this context, the real expressiveness of a neural network layer is a loose term. With it, we refer to a *measure* quantifying the differences between the functions induced by each neuron on the layer i. The higher the differences in activations for the different neurons in the layer, the higher this *real expressiveness*. Furthermore, the branching patterns in the Mapper graph were correlated with the accuracy of the model in each training phase, supporting the previous claim.

In a second experiment, performed with an FCFNN with two hidden layers, each containing 100 neurons, and weights initialized randomly following a normal distribution, the Mapper graph for the output layer also contained ten branches. However, for the first hidden layer, PCA projections of the weights studied seemed to distribute in a tree evolving in parallel on top of a smooth surface, and a Mapper surface generated using the first three principal components of the points further confirmed this fact. This surface was remarkably robust under variations in the training parameters and the initial weights.

Mapper on activations

Mapper has also been used to analyze the structure of neural network activations. TopoAct, by Rathore et al. (2021), was proposed as a visual exploration system to study the topology of activation vectors in fixed layers of neural networks. TopoAct uses the l^2 norms of the activation vectors and `DBSCAN` as the filter function and the clustering algorithm, respectively. The image cover of the filter function is built using an algorithm considering a fixed number of cover sets with a fixed percentage of overlap between them. Finally, each Mapper