A Appendix

A.1 Topological Definitions

Here we include the precise definitions of the required notions – this should not be considered as an exhaustive introduction, but rather our goal is to give precise definitions to the concepts used in the paper.

A cell complex can be defined inductively by dimension. Let $\mathcal{X}_0$ be the 0-skeleton of the complex consisting of points, which can be thought of as 0-dimensional balls (cells). The k-skeleton $\mathcal{X}_k$ is obtained by attaching some number of k-dimensional cells by their boundaries to the $(k-1)$-skeleton $\mathcal{X}_{k-1}$ via continuous maps. This process can be repeated indefinitely to form the cell complex $\mathcal{X}$, but in practice we terminate at some finite dimension. While cell complexes can be used fairly flexibly in encoding topological spaces, simplicial complexes are typically preferred in computational settings for their combinatorial description.

Definition A.1. *A k-dimensional simplex $(v_0, \dots, v_k)$ is the convex combination of $k+1$ vertices, $\{(v_0), \dots, (v_k)\}$.*

We only consider the situation where the vertices correspond to point in a sufficiently nice ambient space, usually $\mathbb{R}^d$ so that a geometric realization of simplices exists as a subset of the ambient space. Simplices are used to represent a space, but must satisfy some additional constraints so that they form a *simplicial complex*.

Definition A.2. *A simplicial complex $\mathcal{X}$ is a collection of simplices such that*

1. *For every simplex σ in $\mathcal{X}$, every face $\tau \subseteq \sigma$ is also in $\mathcal{X}$.*
2. *For any two simplices σ_1 and σ_2, $\tau = \sigma_1 \cap \sigma_2$ is a face of both σ_1 and σ_2.*

These can be thought of as higher dimensional analogs of graphs or triangular meshes – conversely, a graph is a 1-dimensional simplicial complex.

To define homology, we first construct a *chain group*. In our setting, the k-th chain group $C_k(\mathcal{X})$ is the freely generated group generated by k-dimensional simplices. Importantly, there exists a boundary homomorphism

$$\partial_k : C_k(\mathcal{X}) \to C_{k-1}(\mathcal{X})$$

such that $\partial \circ \partial = 0$. Explicitly,

$$\partial_k(v_0, \dots, v_k) = \sum_{i=0}^{k} (-1)^i (v_0, \dots, \hat{v}_i, \dots, v_k)$$

where $\hat{v}_i$ indicates that the i-th vertex has been removed. Again, we take $\partial_0 = 0$. Homology can then be defined as

$$H_k(\mathcal{X}) = \frac{\ker \partial_k}{\text{im } \partial_{k+1}}$$

We consider homology computed over fields, in which case the homology groups are vector spaces. Rather than only consider one simplicial complex $\mathcal{X}$, we can consider an increasing sequence of simplicial complexes, called a *filtration* $\mathcal{X}_0 \subseteq \mathcal{X}_1 \subseteq \dots \subseteq \mathcal{X}_N$. The requirement is that each $\mathcal{X}_i$ is itself a simplicial complex. We consider the filtration function $f : \mathcal{X} \to \mathbb{R}$ such that each sub/super-level set $f^{-1}(-\infty, \alpha]$, resp. $f^{-1}[\alpha, \infty)$ form a filtration in parameterized by α. Under appropriate finiteness conditions, which are always satisfied for finite simplicial complexes (Edelsbrunner and Harer, 2010).

Theorem A.3. *(Zomorodian and Carlsson, 2005) If homology is computed over a field, then the homology of a filtration admits an* interval decomposition*. The interval decomposition is direct sum of rank 1 elements which exists over an interval $[b, d)$.*

In our setting, whether the intervals are open or closed is not important so we supress this aspect of the notation. This collection of intervals is the *persistence diagram*.

Definition A.4. *A persistence diagram is a collection of points (b_i, d_i), possibly with repetition, along with the diagonal Δ, i.e. all points such that $b = d$.*

The diagonal plays an important part in the definition of distances between diagrams. The main distance we consider is the p-Wasserstein distance between two diagrams

$$W_p(\text{PD}_k(f), \text{PD}_k(g)) = \inf_{\varphi} \left(\sum_{p \in \text{PD}_k(f)} ||p - \varphi(p)||^p \right)^{1/p}$$

where φ is a bijection. This can be viewed as an optimal transport problem, points must either be matched to the other diagram or the diagonal. This can be formulated as a classical linear program if we add to each diagram the projection of the other diagram to the diagonal

$$\text{proj}(d, b) = \left(\frac{d+b}{2}, \frac{d+b}{2} \right)$$

In other words, we can consider the optimal transport problem between $\text{PD}(f) \cup \text{proj}(\text{PD}(g))$ and $\text{PD}(g) \cup \text{proj}(\text{PD}(f))$ where the distance between any two points on the diagonal is 0.

One of the core techniques in this paper is to compute the gradient of an energy function with respect to some

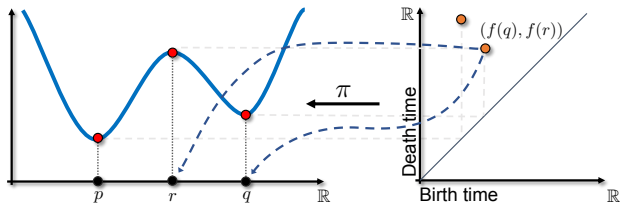

Figure 14: A one dimensional example of a persistence diagram and the inverse map π. The function on the left has critical points at points p, r and q. The local minima create components in the sub-level sets and so represent birth times (x-axis), while the maxima kills one of the components (the younger one) and so is a death time (y-axis). The inverse map for a point in the diagram returns the corresponding critical points/simplicies.

parameters. This is possible through the definition of an inverse map

$$\pi_f(k) : \{b_i, d_i\}_{i \in \mathcal{I}_k} \to (\sigma, \tau)$$

This map is unique and well defined in the case of when a filtration is a strict order, i.e. the simplices are totally ordered in the filtration. This is never the case in our scenarios, where the simplices only form a total order. However this can be resolved by extending the total order to a strict order either deterministically or randomly, see (Skraba, Thoppe, and Yogeshwaran, 2017) for a formal proof and description of how this can be done.

As is the case in our setting, the filtration is not defined directly on the simplices but rather derived from either functions on the vertices or depend on some property of the vertices, e.g. the coordinates of the points that the vertices correspond to. In this case, we require an additional inverse map from simplices to a collection of vertices. This is described in the main paper for two scenarios which we use in the applications. Once both inverse maps are defined, the gradient can be defined in the standard way using the chain rule.

A.2 Comparison

In this paper we use both super-level set filtrations as well as Vietoris-Rips and weak-Alpha filtrations. Figure 15 provides an illustrative comparison of super-level set filtration (on the left) and a Rips filtration (on the right). The digit "9" viewed in terms of the super-level set of its pixel values has one clear connected component and one clear ring, which can be read from the corresponding persistence diagrams. However, the super-level set filtration does not take into consideration distance along the grid and a small (in the 2D sense) hole can be very large from the superlevel set perspective. In the Rips case we lay out all the pixel values in a 3D space and use the euclidean distance. Here there are many more connected components at small to medium filtration values, and the same is true for the number of rings. The Rips does not merely pick up the obvious ring in the "9" but also rings that are formed with respect to the vertical axis.

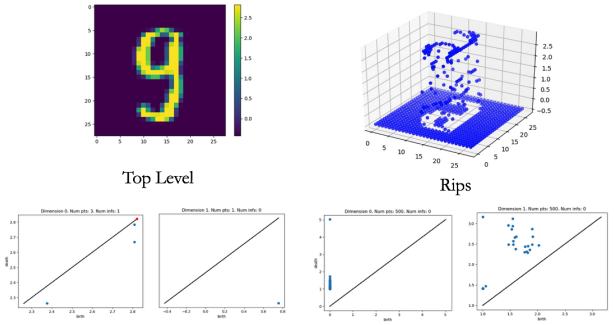

Figure 15: Compare top level set with Rips filtration

A.3 Explore latent space

In this paper we backpropagate to the latent space of generative models (query the latent space) in order to improve on them. However, the ability to topologically query the latent space may not only be used to give us more topologically desirable output but also to explore the nature of the latent space. Consider Figure 16 that consists of images produced by a trained InfoGAN-generator (Chen, Duan, et al., 2016). The type of digit closest to a "2" with a loop on the bottom is arguably a digit "2" without such a loop. However, when we backpropagate to remove the ring we get something that does not look like any MNIST digit. Similarly, if we remove the ring in the digit "6" we get something that does not look very much like any MNIST digit. This gives some indications that the latent space learned by the InfoGAN is relatively topologically flexible.

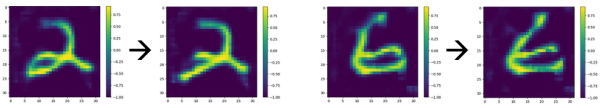

Figure 16: Infogan query

A.4 Topological Mapping

In this paper we train and experiment with networks that use persistence as input. However, with a differentiable layer we can put the persistence computations anywhere in the network. Sometimes it is useful to put certain layers in the middle of a deep network. For example in domain transfer we often want to create a

mapping where features will be more useful. The setup might look like in the Figure 17. We do a simpler setup

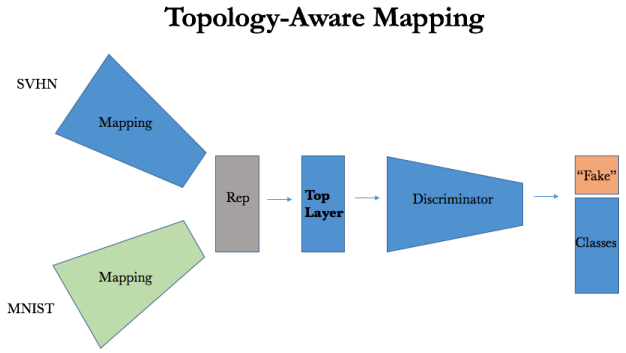

Figure 17: Topological mapping

where we train a network to use superlevel persistence features to classify the naive PD_1-features of MNIST which we define for "1", "2", "3", "4", "5", "7" as zero, for "0", "6", "9" as one, and for "8" as two. We are able to get a 84% accuracy. However, by putting two convolutional layers (the mapping) which maintain the dimensionality of the input before our Topology Layer, we get the results shown in Figure 18 where we ultimate improve our accuracy to 86%. In this case,

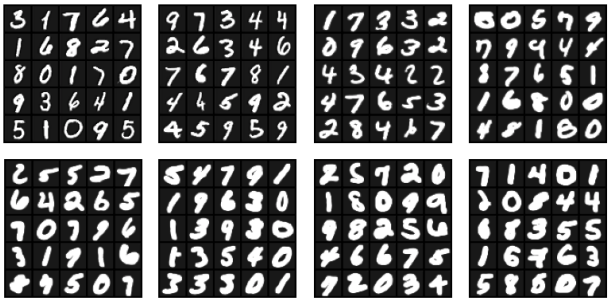

Figure 18: Intermediate Representation in topological mapping

it seems as if the mapping thickens the digits in the images in order to remove topological noise.

A.5 Topological Adversarial Attacks

A.5.1 Features for classification model

In order to train a classification model using only topological features we must include orientation and directional information into the persistence homology features. This can be achieved by using custom filtration constructed in the following way. First we define 8 directional functions

$$g_\theta(x,y) = \cos(\theta)x + \sin(\theta)y \qquad \theta = 0, \pi/4, \pi/2, \ldots, 7\pi/4$$

These functions are shifted and scaled so that in the domain of the image, they range from 0 to 1. If $I(x,y)$ is the input image, then the filtrations are given by

$$f_{\theta_i}(x,y) = I(x,y)g_{\theta_i}(x,y)$$

Persistence diagrams of dimensions 0 and 1 are computed for each filtration. We then compute 25 features on each persistence diagram given by $\mathcal{E}(p,q,0;\mathrm{PD})$, for p and q ranging between 0 and 4, thus totalling 400 features.

A.6 Regularization Sparsity Visualization

In Figure 19 we see the Rips PD_0 diagram for the weights of a logistic regression model without (left) and with (right) L1 regularization. As the test accuracy improves, we see that the persistence diagram changes as well.

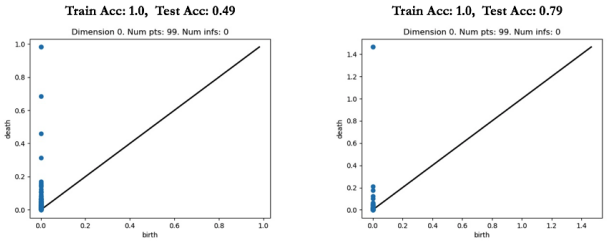

Figure 19: Rips and generalization