Master's Paper of the Department of Statistics, the University of Chicago

Topological Regularization in Deep Learning

Deqing Fu

Advisors: Bradley J. Nelson
Lek-Heng Lim

Approved _____

Date _____

February 25, 2022

Abstract

Topological Data Analysis (TDA) has started to be used in deep learning problems, not only on analyzing the mechanisms of deep neural networks but on regularizing the learning processes. In this work, we will introduce the fundamental theories of persistent homology, followed by their applications in deep learning. Starting from classification problems, we summarize the topological viewpoints and introduce the efficacy of topological regularization on unsupervised classification. Afterward, the work will focus on applying topological regularization to dense prediction problems, such as depth perception and semantic segmentation, which are important applications in computer vision. Dense prediction problems have a concrete topological description in terms of partitioning an image into connected components or estimating a function with a small number of local extrema corresponding to objects in the image. Experimental results show that the output topology can also appear in the internal activations of trained neural networks which allow for novel use of topological regularization to the internal states of neural networks during training, reducing the computational cost of the regularization. We demonstrate that this topological regularization of internal activations leads to improved convergence and test benchmarks on several problems and architectures. At the end of this work, we will provide exploratory experiments with topological analysis on time-dependent vision problems, such as video object tracking.

Contents

List of Figures

Chapter 1

Topological Data Analysis in Deep Learning

1.1 Background

In this section, we review some mathematical backgrounds of persistent homology, following Carlsson [7, 8] and Naitzat *et al.*[51], and review ways of computing persistent homology, following Otter *et al.*[53].

1.1.1 Simplicial Complexes

The topological spaces we use for dense prediction problems are all thought of as subsets of a rectangle, discretized using the Freudenthal triangulation so that the vertex set of the triangulation is arranged to correspond to the pixel grid of an image. This gives a combinatorial representation of the rectangle as a *simplicial complex* $\mathcal{X} = \mathcal{X}_0 \cup \mathcal{X}_1 \cup \mathcal{X}_2$, consisting of a vertex set (0-simplices) $\mathcal{X}_0$, an edge set (1-simplices) $\mathcal{X}_1 \subset \mathcal{X}_0 \times \mathcal{X}_0$, and a set of triangles (2-simplices) $\mathcal{X}_2 \subset \mathcal{X}_0 \times \mathcal{X}_0 \times \mathcal{X}_0$. We will denote k-simplices as $(x_0, \ldots, x_k), x_i \in \mathcal{X}_0$ and a k-dimensional simplex is the convex hull of $k+1$ affinely independent points. The case for point clouds is also very similar, and we'll start introducing mathematical backgrounds by considering point clouds.

Let $X = \{x_0, \cdots, x_n\}$ be the point cloud data of interest. The notion of simiplical complexes corresponding to a point cloud X is given as the following.

Definition 1.1 (Simplicial Complex) *An abstract simplicial complex K is a pair $K = (V(X), \Sigma(X))$ where $V(X)$ is a finite set of* verticies *of X and $\Sigma(X)$ is a subset (simplices) of the collection of non-empty subsets of $V(X)$, such that if $\sigma \in \Sigma(X)$, and $\varnothing \neq \tau \subseteq \sigma$, then $\tau \in \Sigma(X)$.*

Though there are many ways to generate simplicial complexes given a point cloud, the Rips Complex may be the simplest and is widely used in topological data analysis.

Definition 1.2 (Rips Complex) *A* Rips Complex *(also called* Vietoris-Rips *complex) at scale ε on X is written as $\mathrm{VR}_\varepsilon(X)$. It's defined to be the simplicial complex whose vertex*

set is X *and whose* k*-simplices comprise all* $\{x_0, \cdots, x_k\}$ *satisfying* $d(x_i, x_j) \leq 2\varepsilon$*, for* $i, j = 0, \cdots, k$*. Formally, it's written as*

$$\mathrm{VR}_\varepsilon(X) = \left\{ \{x_0, \cdots, x_k\} : d(x_i, x_j) \leq \varepsilon, x_0, \cdots, x_k \in X, k = 0, 1, \cdots, n \right\} \tag{1.1}$$

We use the figure from [65] as a nice example for Rips Complex – see Figure 1.1. There are 18 0-simplices (points). If their neighborhoods (yellow circles of radius ε) intersect, then the two points form an edge; and there are 21 1-simplices (edges) in the given example. For three points, if they are pair-wise connected by 1-simplices, then they form a triangle (2-simplex); and there are 7 2-simplices. Four vertices form a 3-simplex (a tetrahedron) if they are also pair-wise connected by an edge, and there is one 3-simplex here.

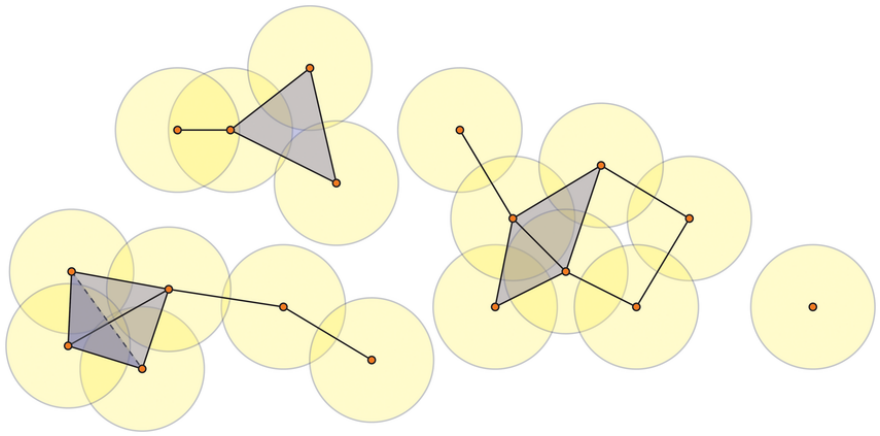


Figure 1.1. An example of *Vietoris-Rips* Complex

A *filtration* is a nested sequence of topological spaces $\{\mathcal{X}_a\}$. In the settings of Rips Complex, $\mathrm{VR}_r(X) \subset \mathrm{VR}_s(X)$ if $r \leq s$ and this naturally forms a filtration. Though Rips complex and Rips filtration are useful in analyse point cloud data [51], it's hard to generalize Rips for image data (as discussed in Chapter 2). Images can either be input or output images of a neural network (with one or more channels), or the internal activations of a neural network on a particular image (typically many channels). In the case of images, we switch to level-set filtrations. A super level-set filtration uses a real-valued function f which takes values over each pixel, in our case we take the 2-norm of the channel values over each pixel. This function can be extended from a function on pixel values (the vertex set of $\mathcal{X}$) to higher-dimensional simplices using a lower-star filtration $f(x_0, \ldots, x_k) = \min_{i=0,\ldots,k} f(x_i)$.

Super Level-set Filtration. Super-level set filtration for a function f extended to a whole simplicial complex use $\mathcal{X}_a = f^{-1}([a, \infty))$, which satisfies the inclusion condition $\mathcal{X}_a \subseteq \mathcal{X}_b$ if $a > b$. An example image with several snapshots of the associated super-level set filtration can be seen in Fig. 1.2.

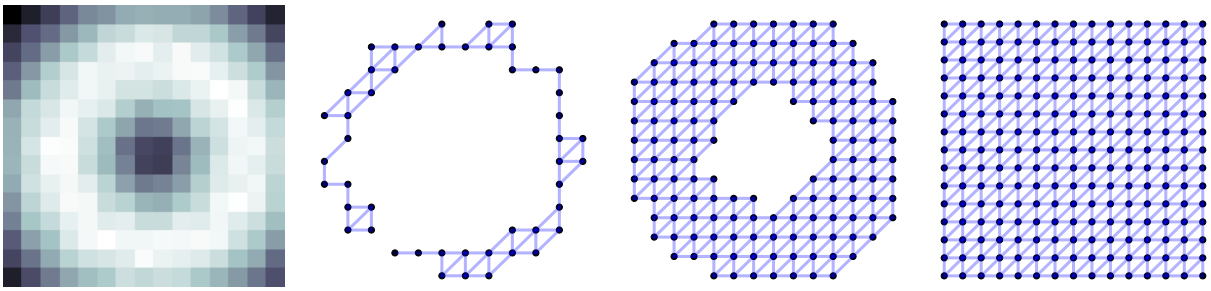


Figure 1.2. An example of super level-set filtration

Left to right: function f of pixel values over an example image where light pixels have higher value than dark. Then a sequence of simplicial complexes obtained from super-level sets of the function: $f^{-1}([1,\infty))$, $f^{-1}[0.85,\infty))$, and $f^{-1}([0,\infty))$.

1.1.2 Persistent Homology

Persistent homology is an algebraic invariant of filtrations which summarizes how topological features such as connected components and holes appear and disappear as the filtration parameter increases. Homology is computed first starting with chain complexes $\{C_k(\mathcal{X})\}_{k\geq 0}$ which are vector spaces with basis vectors for each k-simplex in $\mathcal{X}$, and connected by boundary maps $\partial_k : C_k \to C_{k-1}$ defined (for simplicial complexes) as $\partial_k(x_0,\ldots,x_k) \mapsto \sum_i (-1)^k (x_0,\ldots,\hat{x}_i,\ldots,x_k)$, where $\hat{x}_i$ denotes the removal of the vertex x_i to obtain a $k-1$ simplex from the k-simplex, and $\partial_0 = 0$. These boundary maps are differentials: $\partial_k \partial_{k+1} = 0$ for all $k \geq 0$.

Homology in dimension k is the quotient vector space $H_k = \ker \partial_k / \operatorname{img} \partial_{k+1}$. The rank of homology in dimension 0 counts the number of connected components of $\mathcal{X}$, in dimension 1 counts the number of holes, and, generally, in dimension k counts k-dimensional voids. Homology is a functor, meaning that the inclusions $\mathcal{X}_a \subseteq \mathcal{X}_b$ have associated linear maps $F_{k;a,b} : H_k(\mathcal{X}_a) \to H_k(\mathcal{X}_b)$ which are useful in determining how features in $\mathcal{X}_a$ map to features in $\mathcal{X}_b$. *Persistent homology* $PH_k(\{\mathcal{X}_a\})$ describes how vectors first appear in the co-kernel of a map and then survive the application of maps induced by inclusion until eventually entering the kernel. Persistent homology is characterized up to isomorphism [74, 9] by birth-death pairs $PH_k(\{\mathcal{X}_a\}) = \{(b_i, d_i)\}$ where the pair (b_i, d_i) describes the birth of a new vector at parameter b_i and death of its image at parameter d_i, often visualized plotted in the plane as a *persistence diagram* – see Fig. 1.3 for an example. Pairs with well-separated birth and death are robust to perturbation of function values [17], and points with nearby births and deaths are typically considered topological noise. Notice that the we denote the persistence diagram $PD_k(\{\mathcal{X}_a\}) = \{(b_i, d_i)\} \cup \Delta$ where Δ is the diagonal with infinite number of points at each location. By doing so, the persistence diagram PD_k allows for a perfect matching without changing any information in PH_k, and this is important for defining distance metrics in the next section.

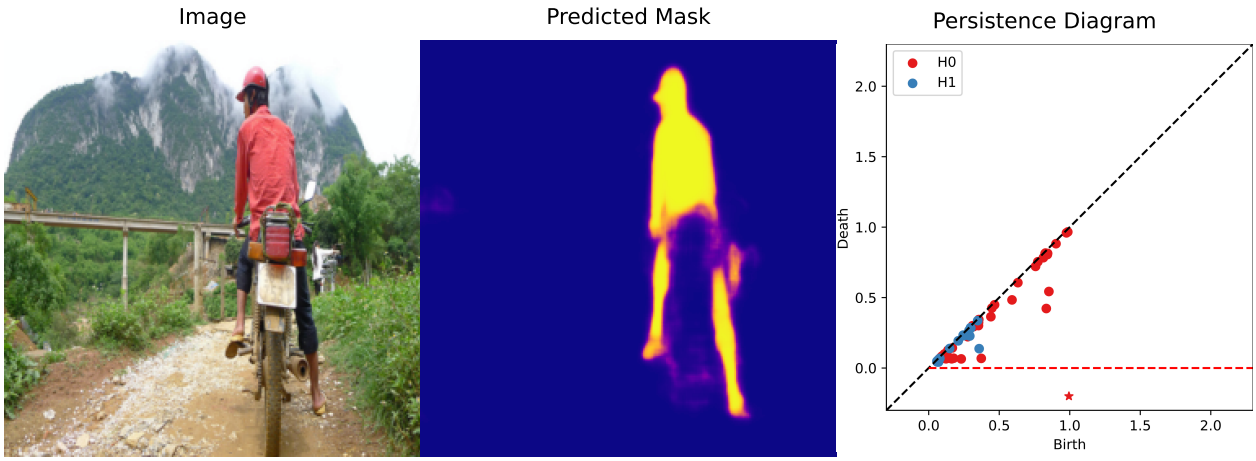


Figure 1.3. An example of computing persistent diagrams from segmentation masks

Left to right: an input image from the COCO data set [47]; semantic segmentation mask; persistence diagram of the mask. Each point in the persistence diagram is a persistence pair: red points are homology in dimension 0, and blue points are homology in dimension 1. There is a single red point below the dashed red line representing the single connected component present at the end of the filtration.

1.2 Topological Regularization

The idea of topological regularization arises naturally as people start to observe nice topological structures in learning problems, for example, one important observation by Carlsson *et al.*[10] shows that the natural image patches land near a manifold with the topology of a Klein bottle. The idea of applying topological regularization to machine learning problems, similar to other learning problems, is also divided into two fashions: the supervised, where the groundtruth persistence diagram is known, and the unsupervised, where the groundtruth persistence diagram is unknown or not required. In the following sections, we'll discuss each method and its related applications.

1.2.1 Supervised Topological Reguarlization

In the settings where we want to optimize the predicted persistence diagram to be "similar" to the ground-truth persistence diagram, we need a distance metric to quantify this "similarity". The two most popular distance metrics for persistence diagrams are Wasserstein distance and bottleneck distance.

Definition 1.3 (p-Wasserstein distance) *The Wasserstein distance between two persistence diagrams can be viewed as the optimal transport distance between two points of diagrams [11]. Let $f, g : \mathcal{X} \to \mathbb{R}$ be two filtration, and φ be a bijection map between points of persistence*

diagrams $\mathrm{PD}_k(f)$ *and* $\mathrm{PD}_k(g)$*. Then the p-Wasserstein distance is defined as*

$$\mathcal{W}_p(\mathrm{PD}_k(f), \mathrm{PD}_k(g)) = \inf_{\varphi} \left(\sum_{p \in \mathrm{PD}_k(f)} \|p - \varphi(p)\|^p \right)^{\frac{1}{p}} \tag{1.2}$$

Definition 1.4 (Bottleneck distance) *It is in fact* $d_{\text{Bottleneck}} = \lim_{p \to \infty} \mathcal{W}_p$ *and can also be written as*

$$d_{\text{Bottleneck}}(\mathrm{PD}_k(f), \mathrm{PD}_k(g)) = \mathcal{W}_\infty(\mathrm{PD}_k(f), \mathrm{PD}_k(g)) = \inf_{\varphi} \sup_{p \in \mathrm{PD}_k(f)} \|p - \varphi(p)\|_\infty \tag{1.3}$$

One nice property of persistence diagrams, under bottleneck distance, is its stability under perturbation:

Theorem 1.1 (Bottleneck Stability [16]) *Let* $\mathcal{X}$ *be a topological space (simplicial complex) and* $f, g : \mathcal{X} \to \mathbb{R}$ *be two sub/super level-set filtration. Let* $\mathrm{PD}_k(f)$ *and* $\mathrm{PD}_k(g)$ *be the corresponding* k*-dimensional persistence diagrams. Then*

$$d_{\text{Bottleneck}}(\mathrm{PD}_k(f), \mathrm{PD}_k(g)) \leq \|f - g\|_\infty = \sup_{\sigma \in \mathcal{X}} |f(\sigma) - g(\sigma)| \tag{1.4}$$

For p-Wasserstein distance when $p < \infty$, there is a similar stability theorem with more technicality. More details can be found in [16]. Edelsbrunner and Harer [21] also present an illustrative figure for Theorem 1.1 – see Figure 1.4.

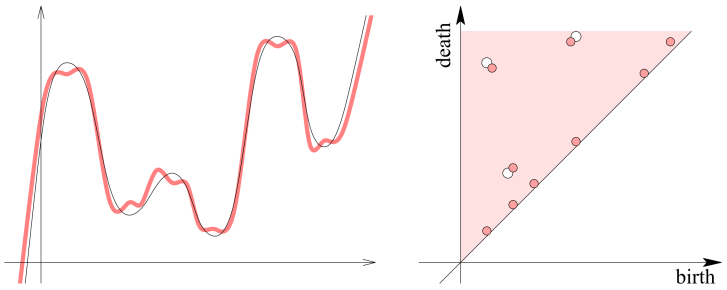


Figure 1.4. Illustration on stability of bottleneck distance

Left: small perturbation to the filtration f. Right: resulting persistence diagrams have small bottleneck distance. (From [21])

The particularly useful insight after Theorem 1.1 is that this makes the process of computing persistence diagrams robust toward noises. Hence, a small perturbation may not change the Betti numbers, which are mostly of interest in topological regularizations. A

slight modification of Definition 1.3 gives a supervised topological loss [34], given that the ground-truth $PD_k(f^*)$ is known:

$$\mathcal{L}_{\text{topology}}^{\text{supervised}}(f, f^*; k) = \inf_{\varphi} \sum_{p \in \text{PD}_k(f^*)} [b(p) - b(\varphi(p))]^2 + [d(p) - d(\varphi(p))]^2 \tag{1.5}$$

where k is the dimension of persistent homology to penalize and the birth-death pairs (b, d) are coordinates of points p in a persistence diagram. One important application of the supervised topology loss $\mathcal{L}_{\text{topology}}^{\text{supervised}}(f, f^*; k)$ is on biomedical image segmentation, where the predictions are desired to avoid unwanted breaks – which happens frequently using vanilla U-Net [58]. Hu *et al.*[34] shows the capability of optimizing the topological loss $\mathcal{L}_{\text{topology}}^{\text{supervised}}(f, f^*; k)$ in biomedical segmentation problems.

1.2.2 Unsupervised Topological Regularization

The supervised regularization discusses previously in Sec. 1.2.1, in most cases, is used more like a loss function than a regularizer, since it requires a ground-truth persistence diagram. However, if we care more about true regularization, they are mostly done in an unsupervised manner, where no ground-truth is required. Leygonie *et al.*[46] summarized many methods of constructing topological regularizations to serve different purposes. Here, we introduce one of them, developed by Brüel-Gabrielsson *et al.*[5].

Each birth or death in persistent homology has a subgradient, one element of which is obtained by associating the birth or death to a particular simplex that changed the rank of homology at the parameter the birth or death occurred. While this mapping is not generally unique, algorithms for computing persistent homology produce a choice of one-to-one mapping, as introduced as the `TopologyLayer` by Brüel-Gabrielsson *et al.*[5]. Brüel-Gabrielsson *et al.*[5] proposed the generalized topological penalty as the following

$$\mathcal{E}(p, q, i_0; \text{PD}_k) = \sum_{i=i_0}^{|\mathcal{I}_k|} |d_i - b_i|^p \left(\frac{d_i + b_i}{2} \right)^q \tag{1.6}$$

One way to compute gradients for the regularizer $\mathcal{E}(p, q, i_0; \text{PD}_k)$ is based on the following assumption: each birth-death pair can be mapped to the cells that respective created and destroyed the homology class. Denote the inverse map as

$$\pi_f(k) : \{b_i, d_i\}_{i \in \mathcal{I}_k} \mapsto (\sigma, \tau) \tag{1.7}$$

Then we can obtain the gradients by summing over sparse indicator functions. Formally, it's

$$\frac{\partial \mathcal{E}}{\partial \sigma} = \sum_{i \in \mathcal{I}_k} \frac{\partial \mathcal{E}}{\partial b_i} \mathbb{I}_{\pi_f(k)(b_i) = \sigma} + \sum_{i \in \mathcal{I}_k} \frac{\partial \mathcal{E}}{\partial d_i} \mathbb{I}_{\pi_f(k)(d_i) = \sigma} \tag{1.8}$$

Remark: One important remark about this above method on computing gradients is that, in cases rarely seen in practice, there can be a situation where there is no descent direction on some edges. For an example of Rip complex, there is a zero-length birth-death

pair which occurs when an edge and a triangle is added at the same filtration level. The inverse map Eq. (1.7) is well-defined, but the inverse map from siplices (σ, τ) to edges is not injective (since the edge and the triangle must share the same edge). Formal analysis and solutions corresponding other ways or assumptions to compute gradients are discussed in [46]. Nevertheless, this is only a theoretical failure case, that can hardly appear in reality, and the loss function Eq. (1.6) and its gradient Eq. (1.8) serve their designed purposes in real applications (see [5] and further Chapter 2 in this work).

Some interesting application of this topological regularization includes robust adversarial attacks [5] and surface reconstruction [4]. In Chapter 2, we will introduce the novel use of this type of topological regularization to dense prediction problems. The particular regularization we use for dense prediction problems and classification problems is in the family of functionals based on algebraic functions of the birth-death pairs [1]. Specifically, we penalize all but the κ longest birth-death pairs in dimension 0:

$$\mathcal{L}_{\text{Topology}}(\{b_i, d_i\}) = \mathcal{E}(2, 0, \kappa; \text{PD}_0) = \sum_{i>\kappa} |d_i - b_i|^2. \tag{1.9}$$

This encourages at most k local maxima in the function f over the image channels.

Computation. To apply our topological loss we use the TopologyLayer PyTorch package [5] modified to use the union-find algorithm [64] to compute PH_0, which runs in $O(m\alpha(m))$ time, where m is the number of edges in the simplicial complex and α is the inverse Ackermann function. To compute persistent homology in higher dimensions it is necessary to perform a factorization of boundary matrices which can be achieved in matrix multiplication time [50]. Standard implementations are asymptotically cubic in the number of simplices, but sparsity in boundary matrices often makes this bound pessimistic [53]. An advantage of our method is that the size of the spaces we use to regularize internal activations of a network is an order of magnitude smaller than the size of the output space, significantly reducing the cost of using a topological penalty.

1.3 Self-supervised Learning

The method of learning representation of images has its profound impacts on pre-training networks. Training a network with full supervision surely can learn the latent space that can be viewed as the space where the images are embedded to. But the supervised method can sometimes lead to a strong bias toward the training dataset. Thereafter, researchers turn to unsupervised, or self-supervised, ways of representation learning. Early works using automatic colorization [71, 42] and inpainting [55] validate the idea of feature learning in a self-supervised manner. Afterwards, He *et al.*[28] and Chen *et al.*[15] proposed the unsupervised contrastive representation learning methods.

Most of the representation learning work learns representations with a unit L_2 norm constraint, that is to embed images onto a unit hypersphere. One nice property of the hypersphere is that, if the features of a class are sufficiently well clustered, they should be linearly separable from samples of other classes in the feature space [68]. Wang *et al.*[68] also introduce the metrics of *alignment*, that favors encoders to map positive pairs to nearby features

on the hypersphere; they also show that the embedding space learned through contrastive learning shows better alignment than learned through supervision. A similar interpretation in the topology sense of *alignment* would be that the features of each class should only form *one* connected components on the hypersphere. In the next few paragraphs, we'll introduce the contrastive learning framework and how to apply topological regularization onto it.

We consider the following important contrastive loss (and sometimes also called InfoNCE loss) which has been popular throughout the research for contrastive learning.

Definition 1.5 (InfoNCE loss [66]) *Suppose that $p_{data}(\cdot)$ be the data distribution and $p_{pos}(\cdot,\cdot)$ be the distribution of positive pairs, and suppose they satisfy $p_{pos}(x,y) = p_{pos}(y,x)$ and $\int p_{pos}(x,y)dy = p_{data}(x)$. Let $f : \mathbb{R}^n \to \mathcal{S}^{m-1} \subset \mathbb{R}^m$ be the hypersphere encoder/embedding function to be learned. The InfoNCE/Contrastive loss can be formulated as*

$$\mathcal{L}_{\text{contrastive}}(f;\tau,K) = \mathbb{E}_{\substack{(x_q,x_k)\sim p_{pos} \\ \{x_{k^-}^{(i)}\}_{i=1}^K \sim p_{data}}} \left[-\log \frac{e^{f(x_q)^\top f(x_k)/\tau}}{e^{f(x_q)^\top f(x_k)/\tau} + \sum_{i=1}^K e^{f(x_q)^\top f(x_{k^-}^{(i)})/\tau}} \right] \tag{1.10}$$

The InfoNCE loss can be thought as the log loss of $K+1$-class classifier that tries to classify x_q as x_k [28]. The encoder function $f : \mathbb{R}^n \to \mathcal{S}^{m-1}$ can be different for query image and key images, that is we can have f_q to compute representations of x_q and f_k to extract representations of keys $\{x_k, x_{k^-}^{(1)}, x_{k^-}^{(2)}, \cdots, x_{k^-}^{(K)}\}$.

Topological Regularization for Contrastive Learning. As discussed previously, we want to weakly supervise the learning process such that each class can only have one connect component in the embedding space. An weaker alternative to regularize this explicitly is to penalize the entire batch to have more than n connect components so that we penalize over-classification, if we have prior knowledge that the dataset has n classes. Aside from this, we also don't want the encoder function to map all images to fewer than n connected components, which would definitely lead to under-classification. Hence, we design the following topological regularization for contrastive learning, where the first part resolves over-classification and the second part penalizes under-classification,

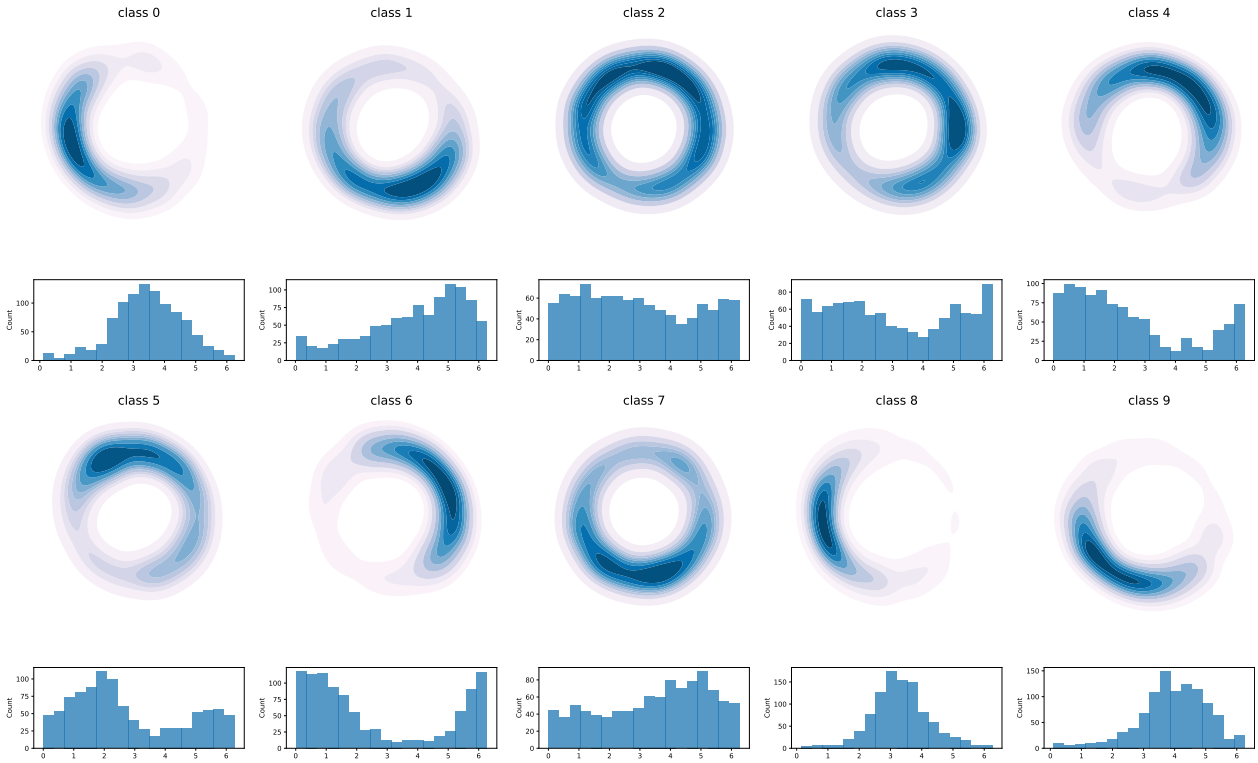
$$\mathcal{L}_{\text{contrastive}}^{\text{topology}}(\{b_i,d_i\}) = \mathcal{E}(2,0,n;\text{PD}_0) + \gamma \cdot [-\mathcal{E}(1,0,0;\text{PD}_0)] \tag{1.11}$$

Notice that the persistence homology is computed through Rips filtration introduced in Sec. 1.1. As an analogy to contrastive learning, the first part of the topological regularization can be viewed as an additional attractive force that similar samples are pushed together while the second part is an additional repulsive force while points are pushed away. We set $\gamma = 0.5$.

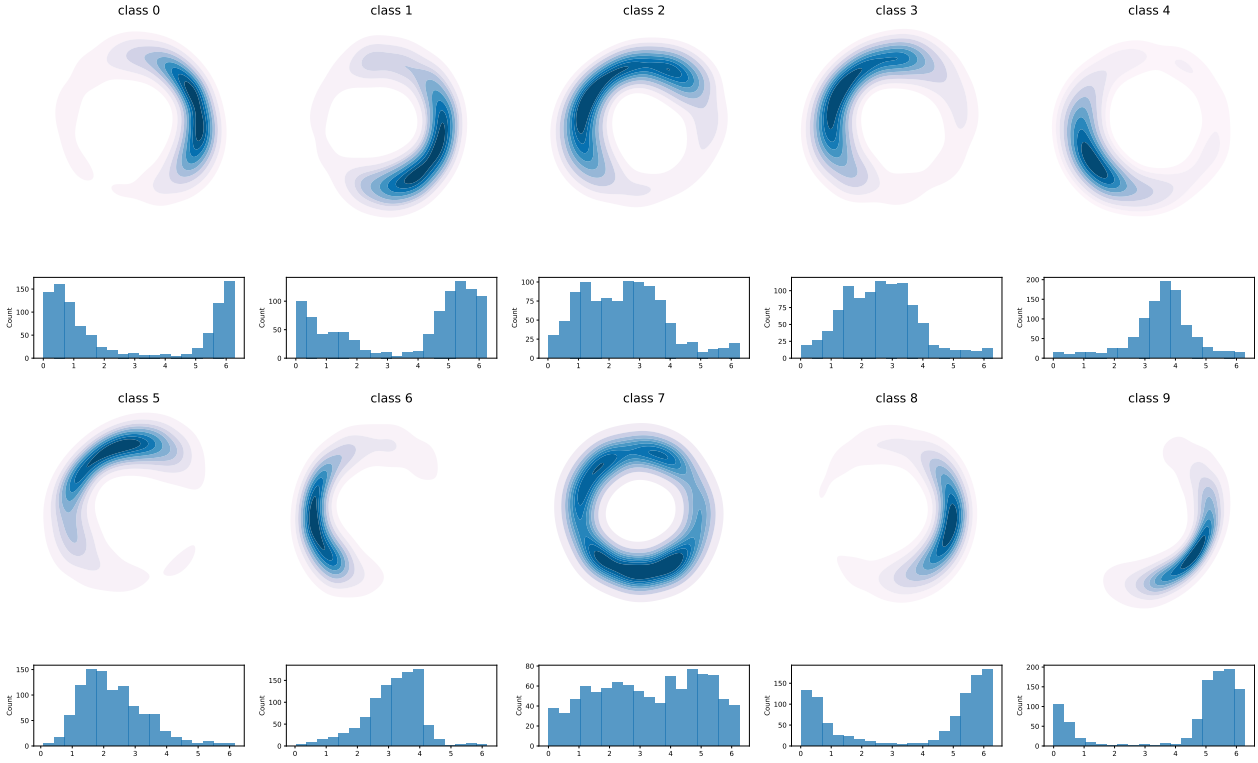
Experiments We experiment the proposed regularization on the `CIFAR-10` [41] dataset based on the MoCo [28]. The combined loss function with a weighted regularization is

$$\mathcal{L}(f;\tau,K) = \mathcal{L}_{\text{contrastive}}(f;\tau,K) + \lambda \mathcal{L}_{\text{contrastive}}^{\text{topology}}(\{b_i,d_i\}) \tag{1.12}$$

In our experiments, we train the network with the following hyper-parameters: batch size of 512, memory bank $K = 4096$, embedding hypersphere of $\mathcal{S}^{128-1} \in \mathbb{R}^{128}$, softmax temperature $\tau = 0.1$ and number of epochs being 200. We use cosine learning rate decay with an SGD



(a) Without Topological Regularization



(b) With Topological Regularization

Figure 1.5. Effects of Topological Regularization on the data embedding space $\mathcal{S}^1$. Each subfigure shows KDE visualizations of without or with regularization respectively. In each subfigure, the first and third rows shows the dimensionality reduced embedding on $\mathcal{S}^1$ while the second and fourth rows show the histograms of angles.

optimizer to achieve optimal performance. When training without topological regularization, we set $\lambda = 0.0$ and when training with explicit topological regularization, we set $\lambda = 0.01$.

We visualize the first two principal components of the embedding of all testing data of `CIFAR-10` using PCA [24], afterwards, we normalize the coordinates such that the reduced embedding also lies on a hypersphere, which in this case $\mathcal{S}^1 \subset \mathbb{R}^2$. There might be other ways of projecting high-dimensional embedding $\mathcal{S}^{127}$ towards low-dimensional $\mathcal{S}^1$, such as the newly proposed topological preserving PCA [52], but such technique may not be suitable for large datasets, such as `CIFAR-10` test set with 10,000 samples. In visualizing the estimated distribution of the reduced embedding $\mathcal{S}^1$, we apply 2D kernel density estimation [54] on coordinates and compute histograms on angles ($\arctan(y/x)$). The resulting visualization on both with and without explicit topological regularization is shown in Figure 1.5.

In Figure 1.5a showing embedding without regularization, the visualized KDE of class 2 (`bird`) and class 7 (`horse`) shows flatter distribution and that of class 3 (`cat`) shows two connected components, or peaks. However in Figure 1.5b showing embedding with topological regularization, each class is more concentrated, except for class 7 (`horse`).

However, quantitatively, we cannot improve the linear classification accuracy as expected. We trained a linear classifier on top of the self-supervised learned embedding for 200 epochs for each scenario and the Top-1 accuracy for vanilla MoCo is 82.14% while the Top-1 accuracy for MoCo with topological regularization is only 81.97%. Though this result may be subject to randomness, but it can still indicate that our proposed regularization hardly helps with linear separability quantitatively despite the qualitative improvements shown in Figure 1.5.

Limitations and Future Work The first limitation is, as discussed above, despite the qualitative analysis shown in Figure 1.5, there's no improvement on linear separability. Second, the visualization process is quite simple, it only involves PCA and normalization, which will lead to corruption of topology during the process. Finally, the proposed regularization term Equation (1.11) is limited to penalize the topology of a single batch, and this type of topological regularization would require very large batch size. This may work when the number of classes is small, such as the `CIFAR-10` dataset only has 10 classes. But when the number of classes can be way larger than the batch size, for example the ImageNet dataset or `CIFAR-100`, the proposed regularization will make no sense, since the first penalty term in Equation (1.11) penalizes nothing since the regularizer Equation (1.9) skips every birth-death pair when $\underbrace{|\mathcal{I}_k|}_{\text{number of visible points/batch size}} < \underbrace{\kappa}_{\text{number of classes}}$. Future work may include designing a framework that can aggregate multiple batches for topological regularization, or freeze most parts of the network except the last few layers during the regularization phase to make the batch size as large as possible. Frankly, the discussed work of using topological regularization for contrastive representation learning is fairly preliminary and a significant amount of exploratory work is needed to move forward.

Chapter 2

Topology for Dense Prediction

2.1 Introduction

Dense prediction problems are a class of problems which attempt to infer some value at every pixel in an image. Examples include semantic segmentation and monocular depth estimation which we consider here, as well as instance segmentation, hierarchical boundary detection, and figure-ground inference. Convolutional neural network models have proved to be highly effective at solving such tasks, and many common architectures for dense prediction are built using an encoder and decoder stitched together in some variation of a U-Net [58]. Despite their success, neural networks have drawbacks such as expensive training (both in terms of time and power consumption) and typically requiring large amounts of data [60]. Regularization using some form of prior knowledge about a problem can be beneficial in reducing the amount of data or the number of iterations needed to train a network, and can also improve the generalization of the network [27].

Dense prediction tasks on natural images typically ask a network to infer properties of regions of an image corresponding to individual objects. This partitions an image into relatively few components where pixels in a shared component are treated in a similar manner, at least in the output of the network. Viewing the output of the task as a function on pixels, this means we expect a function (not necessarily smooth) with few local extrema corresponding to the regions of the image that are most or least prominent in the prediction task. Total variation regularization [6] is commonly used in imaging tasks where the output is expected to be piece-wise constant. However, another natural point of view agnostic to smoothness of functions is to penalize the number of extrema directly using levelset topology.

We introduce a regularization method based on the super-level set topology of a function which encourages few local maxima in the output of a dense prediction problem. This method is based on persistent homology [22] and joins a growing body of topological regularization methods finding use in machine learning [46, 5, 12, 39, 33]. We find that this super-level set topology is not only important in the output of the network, but also in the internal activations of the decoder portion of the network. We demonstrate that topological regularization of the internal activations of the network during training leads to faster convergence and improved inference results on both a semantic segmentation task and a monocular depth estimation task.

2.2 Related Work

Semantic Segmentation is a common task in image processing and it requires a dense prediction on each pixel of its corresponding classification. Convolutional neural networks (CNNs) [44] have shown great capability in solving semantic segmentation problems, and many architectures and methods have been introduced. Region based methods such as R-CNN [25], and its derivative, Mask R-CNN [29] use a CNN as a feature extractor for region proposals and refine from bounding boxes to semantic segmentation masks. FCN [61] demonstrates pixel-to-pixel prediction capabilities, U-Net [58] shows great performance on medical images and DeepLab [14] proposes atrous spatial pyramid pooling (ASPP) to handle objects of multiple scales. In recent works people are also interested in finding ways of learning semantic representations in an unsupervised manner. Zhang *et al.*[72] show a way of bootstrapping semantic representation learning via primitive hierarchical grouping such as edges. Aside from development in model architectures and learning schemes, another important line of work seeks to regularize networks to produce high-quality and robust semantic results. Jia *et al.*[37] shows that integrating the total variation regularization [6] to deep convolutional neural networks can both improve the quality of segmentation and its robustness to noises. The key difference of this work to Jia *et al.*'s is that they only regularize the last softmax layer while this work uses topology to regularize internal activations, not just the last layer.

Monocular Depth Estimation is another topic of interest in the area of dense predictions. There is a wide range of contributions to learning depth from stereo images. The goal of monocular depth estimation, however, is to predict depth from a single RGB image. Eigen *et al.*[23] initiated the idea of using deep neural networks to estimate depth without using superpixelation. More recent work follow the step of improving the capability of convolutional neural networks. Alhashim *et al.*[2] propose the DenseDepth model that leverage transfer learning, to use ImageNet [59] pretrained DenseNet [36] as encoders to extract features and a decoder composed of basic convolution layers to reconstruct depth. Lee *et al.*[45] proposed the BTS model that use local planar guidence layers to further improve the encoder-decoder scheme.

Aside from the supervised manner of training on one specific dataset, Lasinger *et al.*[43] demonstrate a zero-shot technique to mix multiple data sets, even with incompatible annotations, which can improve the generalization skill of the networks. Outside of the family of convolutional neural networks, Ranftl *et al.*[57] extends the vision transformers [20] to monocular depth estimation, and achieves better results than CNNs. This work will focus on CNNs, but discussions on vision transformers will be a promising direction for future work. Depth maps are intrinsically equipped with one nice property: they only have a small number of local extrama corresponding to object instances in the images. This work will leverage this property and show that a topological regularization on internal activations can also improve monocular depth estimation tasks.

Topology and Neural Networks. The key contribution of this work is to introduce a method of topological regularization for dense prediction problems which is built on persistent homology. Persistent homology [22] is a an algebraic signature of filtrations of topological

originating in topological data analysis which measures the robustness of topological features such as connected components and holes in a topological space. There has been a recent broad effort to develop persistent homology-based losses and regularization terms in a variety of contexts, including the development of application-specific losses [4, 13, 32, 62] as well as general implementations and theory to make this tool increasingly available [46, 5, 12, 39, 33]. Several applications to computer vision have previously appeared, including a previous application to semantic segmentation by Hu *et al.*[34]. In contrast to Hu *et al.*our method does not use any form of ground truth in the regularization, or an expensive bottleneck distance computation. Our regularization scheme is similar to those suggested for the use in training generative adversarial networks on images of objects by Brüel-Gabrielsson *et al.*[5], and we leverage the implementation provided in their work.

A particularly novel aspect of our work is the application of topological regularization to the internal activations of a network, and, to the best of our knowledge, this is the first work to demonstrate the utility of doing so. Typically, topological losses or regularization terms are applied to the output of the network, but as we see, dense prediction tasks manifest similar topology in earlier activations as well. Our approach is inspired by recent work by Naitzat *et al.*[51] that has shown that trained neural networks tend to simplify topology in internal activations, although their topological construction uses Rips complexes built on entire data sets as they pass through a network, and our work focuses on level set filtrations of single inputs.

2.3 Topology of Internal Activations

In this section, we provide experimental results which demonstrate that topology can appear in the internal activations of trained neural networks. We train a convolutional neural network on binary semantic segmentation task which assigns values to each pixel indicating whether this pixel is part of a human torso or not.

Model architecture. We use the U-Net [58] architecture, which is a popular baseline for segmentation tasks. Our U-Net adopts 6 phases of encoders and 6 phases of decoders, where each block consists of two layers of convolutions, batch normalizations, and ReLU activations. The structure of the network is also shown in Figure 2.1 and the flow of the network is `Input` $\to$ `enc1` $\to \cdots \to$ `enc6` $\to$ `bottleneck` $\to$ `dec6` $\to \cdots \to$ `dec1` $\to$ `Output` with skip connections.

Dataset. We use the COCO dataset [47], specifically COCO-2014, which contains 83K training images and 41K validation images. From its provided semantic segmentation annotations, we make a subset where all images contain human annotations and further process them to only annotate pixels as human/non-human. This commonly used data set contains human objects and [73, 35] have recently studied bias in captions. However, our semantic segmentation task does not use these potentially problematic labels.

Training objectives and hyper-parameters. The training objective is the MSE loss

$$\mathcal{L}_{\text{MSE}}(y, \hat{y}) = \sum_{i=1}^{h} \sum_{j=1}^{w} (y_{i,j} - \hat{y}_{i,j})^2 \tag{2.1}$$

where h and w are the height and width of the image. We train the network for 100 epochs with initial learning rate of 0.01. We use SGD as the optimizer with learning rate decaying by half every 4 epochs.

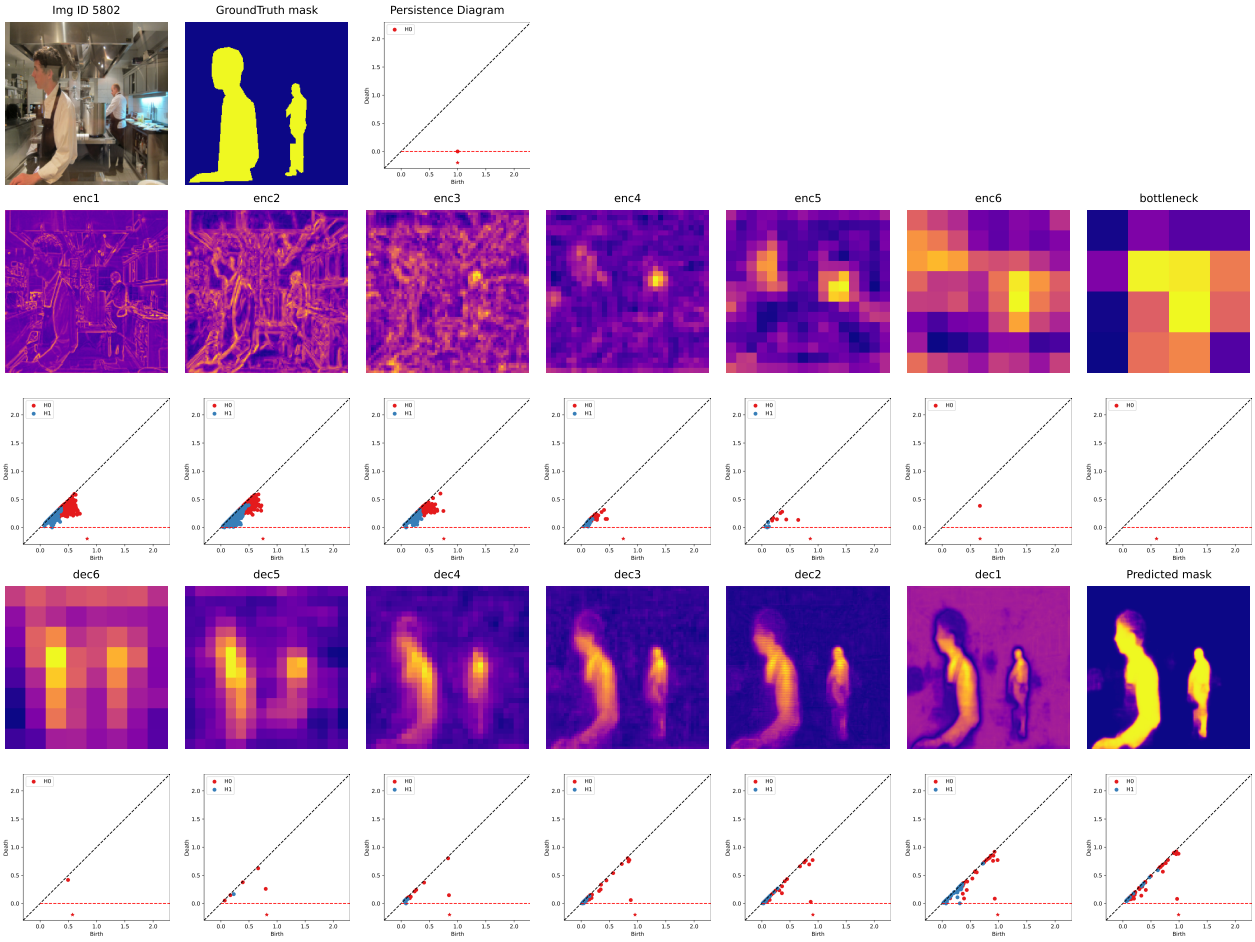


Figure 2.1. Visualizations and Persistence Diagrams of Internal Activations.

The first row shows the input image, the ground truth segmentation mask, and the mask's persistence diagram. The 2nd and 4th row show the visualizations of the magnitude of internal activations. On the 3rd and 5th row, each figure shows the persistence diagrams of the corresponding internal activations above. Points further from the diagonal are more robust features. Both visualizations and diagrams show that the level-set topology can emerge as early as in the 4th encoder layer, and it remains nearly consistent in the decoder layers. No topological regularization is used on this example.

Experimental results. As discussed in Section 1.1, since each intermediate layer $h \in \mathbb{R}^{h\times w\times c}$ has more than one channels, e.g. $c > 1$, there's a natural and differentiable projection π that send the c-dimensional vector at each pixel to a real value, in order to apply super-level set filtration and compute persistence diagrams. The projection $\pi : R^{h\times w\times c} \to R^{h\times w\times 1}$

is defined as, for each pixel (i, j),

$$\pi(\mathbf{h}_{i,j}) = ||\mathbf{h}_{i,j}||_2 \tag{2.2}$$

We evaluate on a completely trained network by visualizing the projection, by Equation (2.2), of each internal activation and computing persistence diagrams. As shown in the example of Figure 2.1, the desired ground-truth mask is consisted of two connected components since there are two human torsos in the input image.

As we follow the steps of this trained convolutional neural network by looking at its internal activations, we can see that the first 3 encoder layers, `enc1`, `enc2`, and `enc3`, present low-level edge structures, and their corresponding diagrams do not reveal any simple topology. Starting from the 4th encoder layer both the visualizations and diagrams demonstrate that topology starts to simplify and two connected components emerge seen as two PH_0 pairs well separated from the diagonal.

Once we proceed to the decoder layers, both visualizations and persistence diagrams illustrate that the topology remains fairly consistent through all decoder layers and the output layer. In Figure 2.1 these two components eventually form the segmentation masks for the two people in the image. In other images the number of robust components in the internal activations typically agrees with the number of connected components in the final segmentation mask. More examples can be viewed in the Appendix A. This provides a natural suggestion of explicitly regularizing the internal activations throughout the training stage.

Topological Regularization on binary Semantic Segmentation. We experimented using the same architecture as described above and trained the networks for 50 epochs, but varying the choice of regularization. We regularize `dec4` with $k = 8$ to penalize more than 8 connected components on this intermediate layer. As shown in Table 2.1, there is a improvement on `mIoU`, e.g., mean Intersection-over-Union accuracy, when we regularize the second decoder layer with the proposed topological regularizer. As shown in Figure 2.2, there are also improvements on convergence speed with the assistance of the proposed regularization.

	mIoU (%)
No Regularization	51.64
Topological Regularization	**52.54**

Table 2.1. Performance Improvement by Topological Regularizer

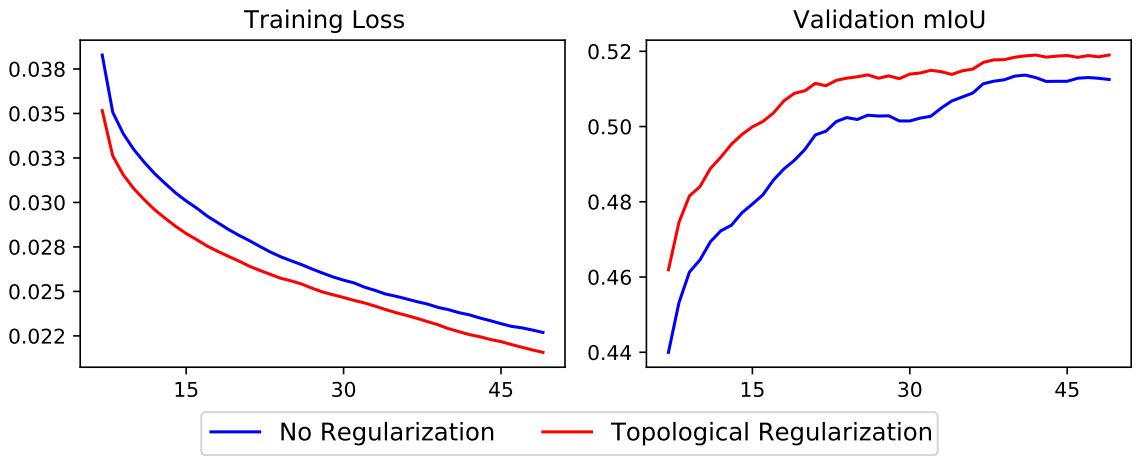


Figure 2.2. Convergence Improvement by Topological Regularizer

The next section will show how this explicitly regularization can improve the convergence and test benchmarks on several architectures on depth prediction tasks.

2.4 Experiments

In this section, we explore the performance improvement on monocular depth estimation with different level of regularizations. We start with the base U-Net architecture, and compare the performance without regularization, with only total variation regularization, and with both total variation and topological regularization. We'll show that both regularization on internal activations assist the convergence and performance of the U-Net model. Subsequently, we experiment with a recent state-of-the-art architecture DenseDepth to demonstrate the versatility of our proposed regularization.

2.4.1 Training Objectives

Our proposed training objective is a weighted sum of three loss functions and two regularization terms:

$$\begin{aligned}\mathcal{L}(y,\hat{y}) = \lambda_{\text{d}} \cdot \mathcal{L}_{\text{depth}}(y,\hat{y}) + \lambda_{\text{g}} \cdot \mathcal{L}_{\text{gradient}}(y,\hat{y}) + \lambda_{\text{s}} \cdot \mathcal{L}_{\text{SSIM}}(y,\hat{y}) \\ + \lambda_{\text{tv}} \cdot \mathcal{L}_{\text{TotalVariation}}(\hat{h}^{(a)}) + \lambda_{\text{top}} \cdot \mathcal{L}_{\text{Topology}}(\hat{h}^{(b)})\end{aligned} \tag{2.3}$$

where y represents the ground-truth label of depth, $\hat{y}$ represents the predicted depth, $\hat{h}^{(a)}$ and $\hat{h}^{(b)}$ represents some intermediate layers of the network. Each loss term is defined as follows,

Depth Loss. We use the RMSE loss in log scale which empirically converges faster than L1 or L2 loss.

$$\mathcal{L}_{\text{depth}}(y,\hat{y}) = \sqrt{\sum_{i=1}^{h}\sum_{j=1}^{w}(\log y_{i,j} - \log \hat{y}_{i,j})^2} \tag{2.4}$$

Gradient Loss. The horizontal and vertical image gradients, $\nabla_{\|}$ and $\nabla_{\perp}$, are computed by a Sobel filter [63, 38]. This further helps align the edges of the ground-truth and the

prediction.

$$\mathcal{L}_{\text{gradient}}(y, \hat{y}) = \sum_{i=1}^{h} \sum_{j=1}^{w} \begin{pmatrix} |\nabla_{\perp} y_{i,j} - \nabla_{\perp} \hat{y}_{i,j}| \\ + \left|\nabla_{\parallel} y_{i,j} - \nabla_{\parallel} \hat{y}_{i,j}\right| \end{pmatrix} \tag{2.5}$$

Structural Similarity Loss. This loss uses the Structure Similarity Index Measure (SSIM) [69] which is shown to be a good loss term for depth estimation tasks [26].

$$\mathcal{L}_{\text{SSIM}}(y, \hat{y}) = \frac{1 - \text{SSIM}(y, \hat{y})}{2} \tag{2.6}$$

Total Variation Regularization. It's often used in imaging tasks where the expected output is piece-wise constant. We apply this regularization term to the last layer $\hat{h}^{(a)}$ before the output layer.

$$\begin{aligned} \mathcal{L}_{\text{TotalVariation}}(\hat{h}^{(a)}) = &\sum_{k=1}^{c} \sum_{i=1}^{h-1} \sum_{j=1}^{w} \left(\hat{h}^{(a)}_{i+1,j,k} - \hat{h}^{(a)}_{i,j,k}\right)^2 \\ &+ \sum_{k=1}^{c} \sum_{i=1}^{h} \sum_{j=1}^{w-1} \left(\hat{h}^{(a)}_{i,j+1,k} - \hat{h}^{(a)}_{i,j,k}\right)^2 \end{aligned} \tag{2.7}$$

Topological Regularization. We apply this regularizer to the second decoder layer $\hat{h}^{(b)}$. We first use the projection described by Equation (2.2) to project this internal activation to $\tilde{h} \in \mathbb{R}^{h \times w}$ and compute its birth-death pairs (b_i, d_i) using super-level set filtration and persistent homology described in Section 1.1. Afterwards, we formulate the loss, given these birth-death pairs, through Equation (1.9), e.g., $\mathcal{L}_{\text{Topology}}(\{b_i, d_i\}) = \sum_{i>k} (d_i - b_i)^2$. For our application, we choose $k = 8$ to penalize internal activations to have more than 8 connected components or local extrema.

To achive the best performance, we set the weights of these objectives as $\lambda_d = 0.1, \lambda_g = 1.0$ and $\lambda_s = 1.0$. As un-regularized DenseDepth to have clearer topology on internal activations than U-Net (see Figure 2.3), we choose different weights of λ_{tv} and λ_{top} for each. For U-Net, we set $\lambda_{\text{tv}} = 1.0$ and $\lambda_{\text{top}} = 0.001$; and for DenseDepth, we set we set $\lambda_{\text{tv}} = 0.1$ and $\lambda_{\text{top}} = 0.0001$.

2.4.2 Models and Training Details

U-Net. We use variants of a ResNet-50 backboned U-Net developed by [49] and [70]. This replaces the encoder part of the original U-Net with a ImageNet pretrained ResNet-50 [30] feature extractor. We trained the model in three settings, without regularization, with total variation regularization, and with total variation plus topological regularization. For all three training procedures, we set the initial learning rate for parameters of the decoder to 0.03 and we keep the learning rate for parameters of the pretrained ResNet encoder to be 1/10 of that of the decoder. Models are trained for 100 epochs with batch size 16, a cosine learning rate schedule, a momentum of 0.9 and a weight decay of 1e-4. For regularization settings, the total variation regularization is enforced onto the last decoder layer and the topological regularization is enforced onto the second decoder layer.

DenseDepth. We use DenseDepth with DenseNet-161 encoder. We trained the model in two settings, without regularization, and with total variation regularization plus topological regularization. To be comparable with scores reported in [67], we use Adam optimizer [40] with initial learning rate 0.0001, $\beta_1 = 0.9$ and $\beta_2 = 0.999$. Models are trained for 20 epochs with batch size 12 and a cosine learning rate schedule. For regularization settings, we keep it the same as in the U-Net experiments, e.g., the total variation regularization is enforced onto the last decoder layer and the topological regularization is enforced onto the second decoder layer.

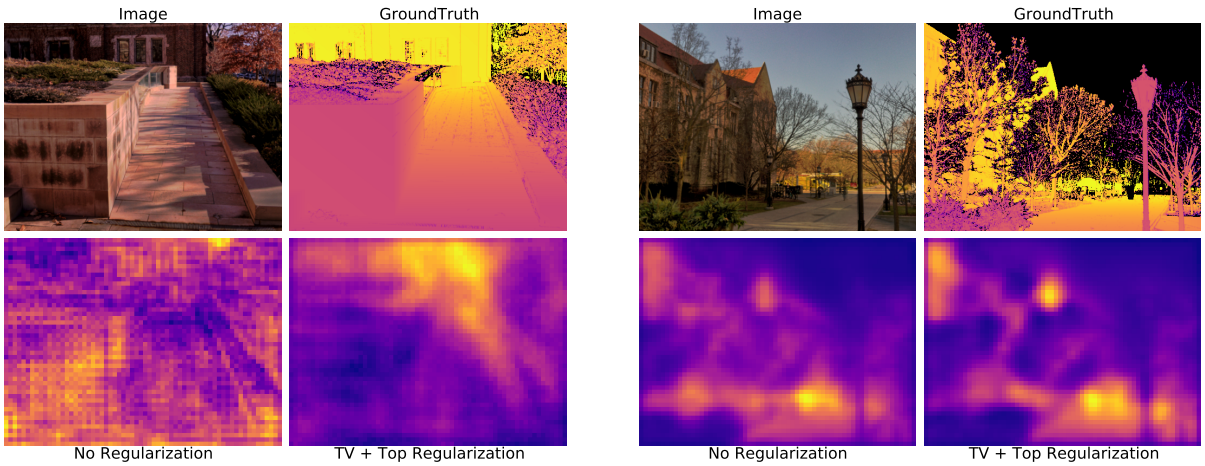
Experiments are trained on two nVidia 2080 Ti's with 11GB memory each.

2.4.3 Data Set

DIODE is a data set that provides images, depth maps and surface normals for both indoor and outdoor scenes [67]. Data provided are captured at a resolution of 1024×768. It contains 8,574 indoor scenes and 16,884 outdoor scenes for training. The maximum depth captured by the sensor is 350 meters and the minimum depth is 0.6 meters. Both our models take half of the orginal resolution as input, i.e., a resolution of 512×384. The U-Net model produces predictions at a resolution of 512×384, and the DenseDepth model produces predictions at a resolution of 256×192 followed by a 2x upsampling through bilinear interpolations.

2.4.4 Augmentation policy

Data augmentation is universally used to reduce over-fitting and can result in better generalization skills. Since the monocular depth estimation tasks aim to predict depth from an entire image, geometric transformations may not be appropriate choices since they may introduce additional distortions that are not natural in depth estimation. We adopt similar data augmentation strategies as [2]. For geometric augmentations, We only performed random horizontal flips with a probability of 0.5. For photo-metric augmentations, we performed random color jittering with a probability of 0.8, random channel swapping with a probability of 0.5, randomly converting images to gray scale with a probability of 0.2, and a random Gaussian blurring with a probability of 0.5. It's unknown if more hand-crated augmentations or automated augmentation policies [19, 18] can also help with generalization skills of trained networks on monocular depth estimation and is an interesting topic for future research.



(a) Regularization effects on U-Net. (b) Regularization effects on DenseDepth

Figure 2.3. Effects of Topological Regularization on Interval Activations.
For each architecture, we visualize the second decoder layer. Our proposed regularization helps the network concentrate better on regions of interest.

2.4.5 Qualitative comparison

We qualitatively investigate our proposed method's regularization effects through visualizations of the internal activations, and specifically, we visualize the second decoder layer for both trained U-Net and DenseDepth, as shown in Figure 2.3. In general, our proposed regularization helps the network concentrate on regions of interest at the early decoding stage.

For the U-Net without regularization, the internal activation of the trained network shown in Figure 2.3a concentrates evenly on nearly everywhere in the image. This might help the network gather local information but it's not necessary for dense estimation problems. This is due to the fact that dense prediction problems tend to have region blocks whose internal gradients are minimal. Thus, it's better for the network to concentrate on regions than on pixels during the early decoding stage, and the fine-grain information can be later accumulated through skip connections from early encoding stages. In contrast, our regularized model learns to focus on high-level regions. Our regularization also aids in reducing artifacts after unpooling and deconvolution seen in the activations.

For DenseDepth, the trained network without regularization already concentrates on regions, as shown in Figure 2.3b. Our proposed regularization further strengthens the level of concentration. The foreground lamps, bushes and trees tend to have lower values in our proposed method, and building regions now tend to have higher values.

In both models, we see that topological regularization aids the decoder in focusing on important regions of the image. This also provides a high level of interpretability into how successful networks are making inferences while abstracting away detail about the particular activations.

2.4.6 Quantitative comparison

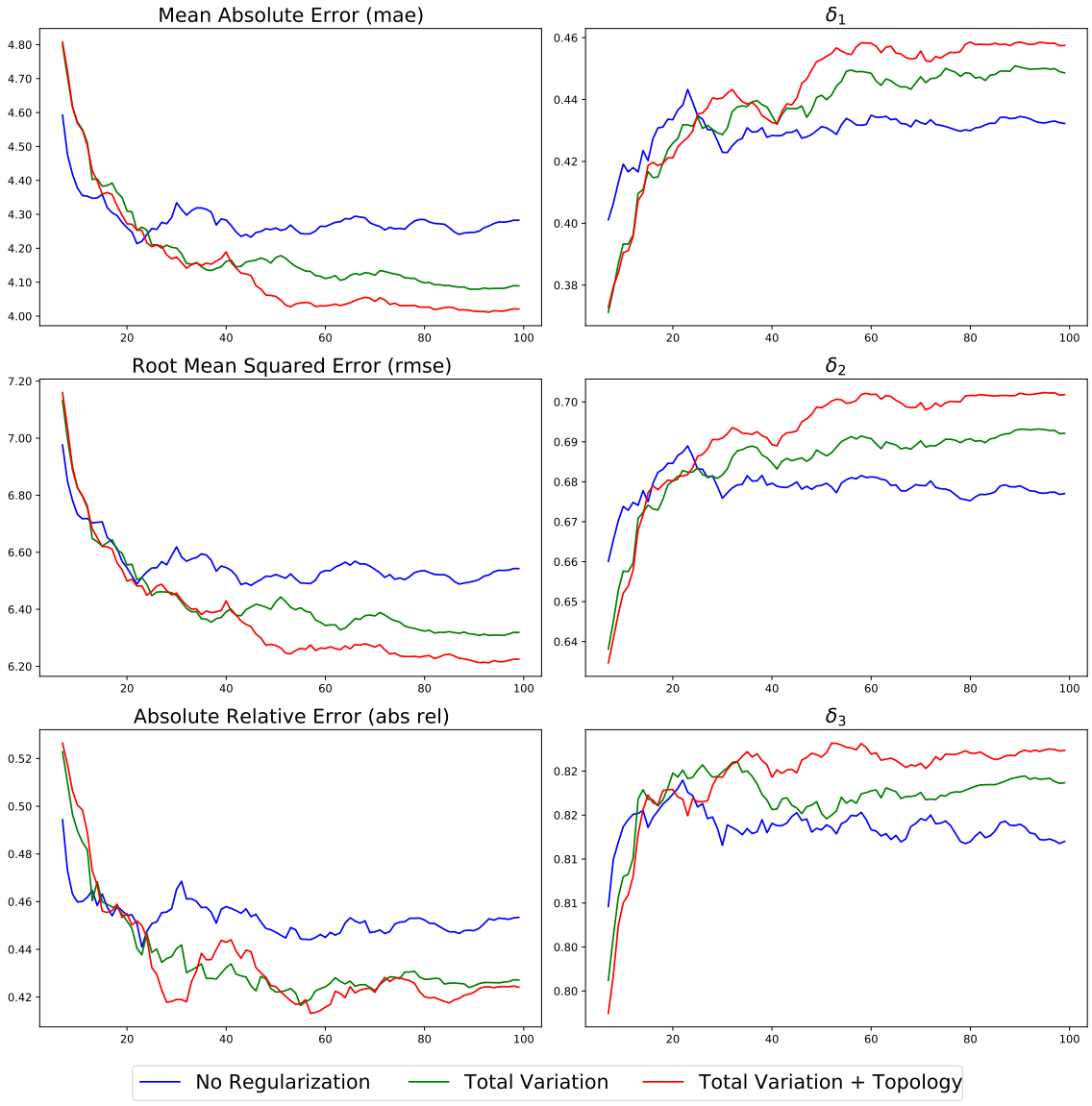


Figure 2.4. Improvements on U-Net Convergence.

Blue lines (without regularization) demonstrates some degree of overfitting as loss increases and accuracy decreases after 20 epochs. Green lines (with total variation regularization) alleviates overfitting a bit, and Red lines (with both proposed regularizations) further overcomes overfitting and converges to a better optimum.

Benchmark Metrics. We quantitatively compare the performance of our regularization on both U-Net and DenseDepth architectures. The accuracy metrics are defined as the following,

i. Mean Absolute Error (mae): $\frac{1}{hw}\sum_{i,j}|y_{i,j} - \hat{y}_{i,j}|$

ii. Root Mean Squared Error (rmse): $\sqrt{\frac{1}{hw}\sum_{i,j}(y_{i,j}-\hat{y}_{i,j})^2}$

iii. Absolute Relative Error (abs rel): $\frac{1}{hw}\sum_{i,j}\frac{|y_{i,j}-\hat{y}_{i,j}|}{\hat{y}_{i,j}}$

iv. mae $\log_{10}$: $\frac{1}{hw}\sum_{i,j}|\log_{10} y_{i,j}-\log_{10}\hat{y}_{i,j}|$

v. rmse $\log_{10}$: $\sqrt{\frac{1}{hw}\sum_{i,j}(\log_{10} y_{i,j}-\log_{10}\hat{y}_{i,j})^2}$

vi. Threshold accuracy (δ_k): Percentage of pixels such that $\max\left(\frac{y_{i,j}}{\hat{y}_{i,j}},\frac{\hat{y}_{i,j}}{y_{i,j}}\right)=\delta_k<1.25^k$. We specifically care about when $k=1,2$ and 3.

	Level of Regularization	lower is better					higher is better		
		mae	rmse	abs rel	mae $\log_{10}$	rmse $\log_{10}$	δ_1	δ_2	δ_3
U-Net	No Regularization	4.2776	6.5386	0.4524	0.1706	0.2181	0.4336	0.6778	0.8128
	Total Variation	4.0952	6.3145	0.4311	0.1649	0.2103	0.4455	0.6915	0.8184
	Topology	4.0548	6.2168	**0.4206**	0.1651	0.2121	0.4388	0.6914	**0.8295**
	Total Variation + Topology	**4.0138**	**6.2044**	0.4269	**0.1614**	**0.2069**	**0.4565**	**0.7020**	0.8232
DenseDepth	No Regularization	3.6554	5.9900	**0.3648**	0.1660	0.2452	0.5088	0.7481	0.8625
	Total Variation	3.5073	5.5763	0.3922	0.1427	0.1884	0.5151	0.7444	0.8665
	Topology	3.5857	5.7030	0.3921	0.1435	0.1887	0.5006	0.7418	0.8668
	Total Variation + Topology	**3.4065**	**5.4196**	0.3908	**0.1395**	**0.1849**	**0.5197**	**0.7582**	**0.8745**

Table 2.2. **Quantitative Comparison.** UNet and DenseDepth Performance on DIODE with different regularizations. Each model is compared internally with varying regularization choices. Numbers in Red indicate the best score whereas in Blue shows the second best.

Convergence Improvement. Our proposed method is also advantageous in speed up the converge. Figure 2.4 plots the U-Net model's convergence curves of both validation losses and validation threshold accuracies. The vanilla model without regularization demonstrates some degree of overfitting as the validation losses start to increase and accuracies decrease after 20 epochs. Adding a total variation regularization helps alleviate overfitting but also shows slight decrease in δ_3 accuracy after around 40 epochs. Enforcing an additional topological regularization further alleviates the overfitting pattern and converges to a better optimum.

Performance Improvement. We benchmark our proposed method on the entire DIODE dataset of both indoor and outdoor scenes. The benchmark results are listed for both models, U-Net and DenseDepth in Table 2.2. Generally, our regularized model can achieve better performance on nearly every metric, compared with the unregularized model.

For U-Net, we further study the effects of different levels of regularization. By simply adding a total variation regularization, we can already reduce the losses and increase the threshold accuracy. As we subsequently add the proposed topological regularization, the scores improve further. In terms of threshold accuracy, we can improve about 2.3% in δ_1, 2.5% in δ_2 and 1.0% in δ_3.

We further develop the same regularization a recent state-of-the-art network, DenseDepth, and our proposed method can also help improve benchmark scores. In terms of threshold accuracy, we can improve about 1.1% in δ_1, 1.0% in δ_2 and 1.2% in δ_3. For validation losses, the only metric our proposed method performs worse on is the absolute relative error. This may due to the fact that our regularization is based on super-level set filtration, which will naturally focus on high-value regions, and in depth estimation problems, these regions would be those at the background. In this sense, our proposed method may perform slightly worse on foreground objects than the baseline model without regularization.

2.4.7 Limitations

While we believe our proposed regularization on interval activations can be applied to boost the convergence and performance of deep convolutional neural networks on dense prediction tasks in general, our experiments only show the validity of the proposed method on limited number of architectures, data sets, and prediction tasks. Currently, our approach requires human input when deciding where to apply topological regularization. An automated procedure may be of interest for future research. We also hand picked the hyper-parameter k for topological regularization, which could also be a learned parameter as a meta-learning task .

2.5 Conclusion

We have shown how super-level set topology plays an important role in two dense prediction problems: semantic segmentation and monocular depth estimation. This offers a high level of interpretability into the internal working of neural networks trained to solve these problems, and we show this can be used to regularize training for faster convergence and increased accuracy. We anticipate these insights will be applicable to other dense prediction problems, and our topological regularization techniques may be adapted to a variety of architectures.

Avenues for future work may include extending topological regularizations to videos where consecutive frames are likely to share similar topological structures. Although this work demonstrates the existence of topological structures in learned convolutional neural networks and shows a way of regularizing internal activations, whether the phenomenon and regularization are also valid for the paradim of vision transformers (ViT) [20] is unknown and of future interest.

We believe that describing the behavior of internal activations of neural networks using topology has great potential for explaining how neural networks operate in practice. Persistent homology has the advantage of abstracting away details about the individual weights and activations in the network while retaining important geometric information – in our case the prominence of local maxima. Regularization is one method of leveraging insights from topology to improve neural networks which we have applied to dense prediction. There are also efforts to use topology to initialize weights [3] and inform decisions in network architecture [51] which may also find application to dense prediction tasks.

Chapter 3

Topology for Time-Aware Problems

In this chapter, we will briefly introduce the current work of Unsupervised Video Object Segmentation (UVOS) and will demonstrate some visualization results, assisted with analysis from a topology point of view, to guide future research.

Lu *et al.*[48] proposed the CO-attention Siamese network (COSNet) to capture correlations between two frames taken from a video sequence. The main architecture is introduced in Figure 3.1. Taking two frames $\{F_a, F_b\}$ of interest from a video sequence, we pass them to already pretrained feature extraction network, which is the DeepLabV3's implementation of ResNet [30] with atrous spatial pyramid pooling (ASPP) [14]. Given the learned feature representations $\{V_a, V_b\}$, the co-attention module is applied to encode the correlations between them. Afterwards, we can refine the embeddings to be $\{Z_a, Z_b\}$, which as experiments shows in [48] can capture more foreground information efficiently. Concatenating Z_a and Y_a, or Z_b and Y_b respectively, and feeding them to a segmentation module gives us the predicted segmentation masks.

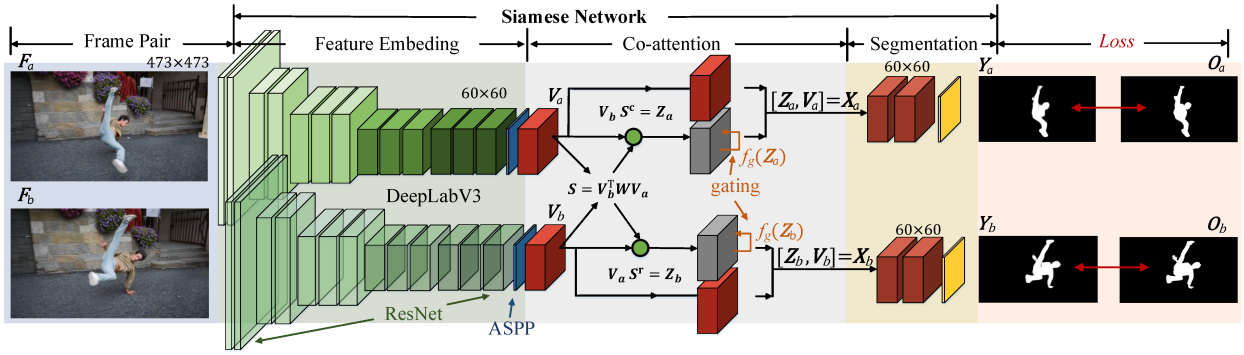
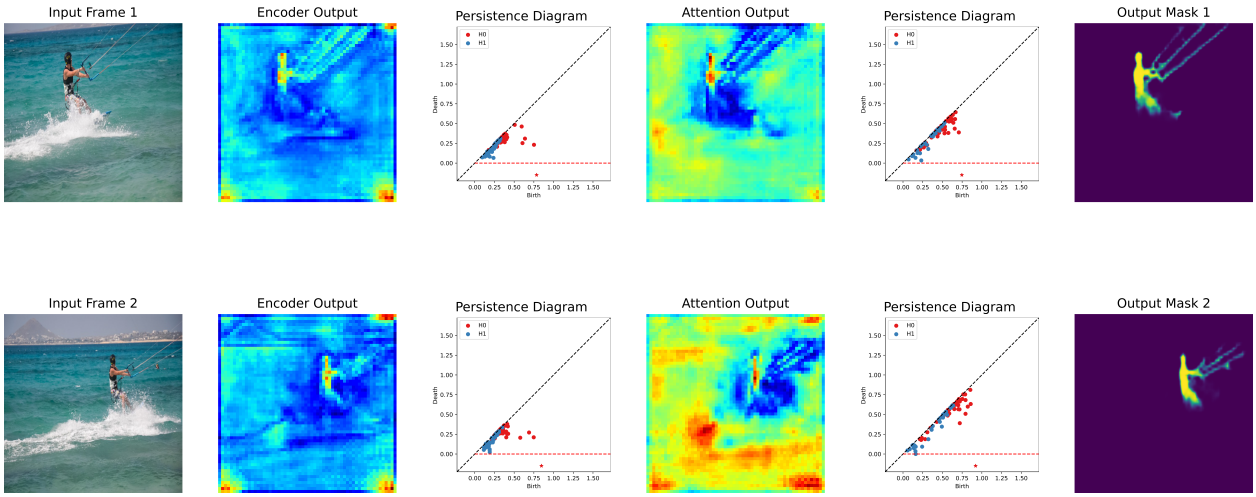


Figure 3.1. COSNet Architecture [48] for Video Instance Tracking

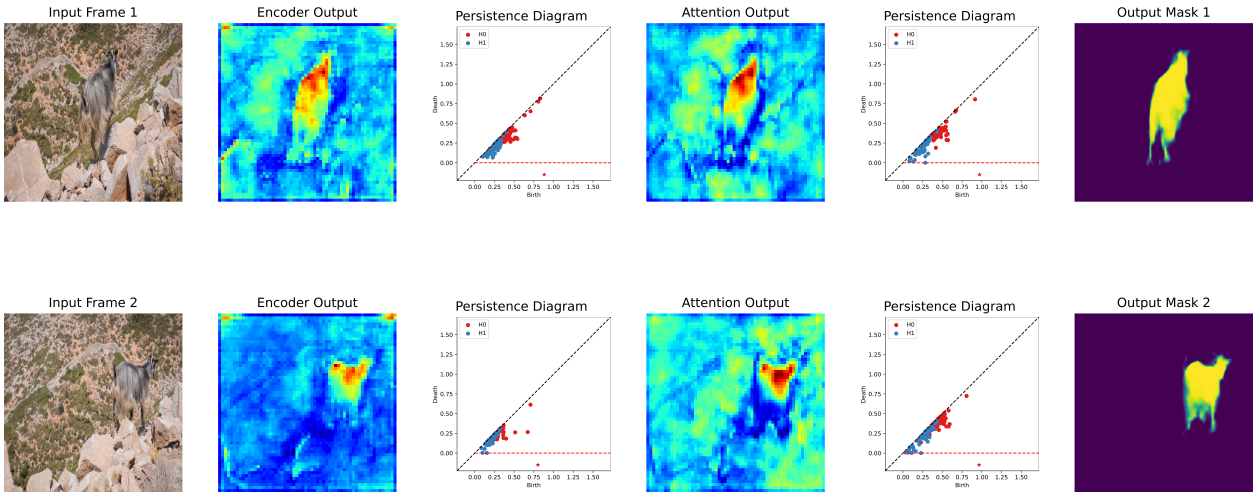
From the architecture illustration [14], as the COSNet tracks the same objects from a video sequence, their predicted segmentation masks should share the topological feature. Moreover, considering the visualization of internal activations on binary semantic segmentation experiments shown in Figure 2.1 of Chapter 2, the topological features of internal activations of both frames of interest should also remain similar. In this sense, we consider extracting persistence diagrams from Y_a, Y_b, the outputs of the feature embedding, and from

$X_a = [Y_a, Z_a], X_b = [Y_b, Z_b]$, the outputs from the Co-Attention mechanism, to see if both convolutions and attention module can keep topological features similar across frames.

Experiment Results and Topological Analysis. We conducted the experiments as discussed above on the DAVIS-16 dataset [56] and the results are shown in Figure 3.2: each row shows a frame from a video sequence while in each row, we show the visualization of Y_a (or Y_b) along with its persistence diagram through super level-set filtration and $X_a = [Y_a, Z_a]$ (or X_b) along with its persistence diagram.



(a) Visualization on the video sequence `kite-surf`



(b) Visualization on the video sequence `goat`

Figure 3.2. Visualization of Internal Activations along with their Persistence Diagrams

As shown in Figure 3.2 with the persistence diagrams, both frames' the encoder outputs and attention outputs show similar topological descriptions, as we desired. Moreover, the persistence diagrams across frames for the attention output look more similar with one another than those for the encoder outputs, and this validates the work of [48] in terms of the necessity of the Co-Attention mechanism and how they help separate foreground objects.

Future Research. Given the experimental results on pre-trained COSNet networks shown in Figure 3.2, a natural extension of topological regularization (summarized in Chapter 1 and experimented on still images in Chapter 2) to the video segmentation tasks is our first step forward. We can try putting regularization in minimizing the Wasserstein distance Definition 1.3 or bottleneck distance Definition 1.4 between encoder outputs Y_a and Y_b or between attention outputs X_a and X_b. Afterwards, we can also experiment on removing the attention module but keeping the topological regularization, to see if topological regularization can serve the same purpose as "attention", since what attention did in COSNet is to assist in isolating foreground objects from the background. Lastly, we can try vanilla LSTM networks [31] rather than so complicated CO-Attention networks to see if a topological aware simple LSTM can beat the state-of-the-art.

Acknowledgements

I would like to thank Dr. Bradley J. Nelson and Prof. Lek-Heng Lim to be my master's thesis advisors. Brad was always patient in walking me into the theory of topological data analysis, a topic that I had no experience before, and enlightening me when we discussed about putting topological regularization on computer vision tasks in Chapter 2. Lek-Heng led me into the arena of applied mathematics and optimization when I took his classes years ago, and saved me from anxiety during the pandemic by bringing me to the master's program in Statistics. Through our group meetings, Brad and Lek-Heng guided me on how to think neural networks from a mathematical perspective and I really thank them in broadening my scope and setting my future research insights.

I would like to thank Prof. Michael Maire for guiding me through the arena of computer vision during several meaningful research projects. Before meeting with Michael, I understood nothing about convolutional neural networks except for its name nor knew how to modify it to serve research purposes, but Michael was always patient in answering my elementary questions. I would like to thank Prof. Mei Wang for caring for us master's students and for giving meaningful suggestions in every aspects beyond academics. Mei is always like a thoughtful parent to the Statistics master's students family.

During my time at the University of Chicago, I was so fortunate to meet many brilliant peers, who made my days meaningful. I especially would like to thank the following during my college life: Shuhong (George) Liu for being my best friend and roommate, for sharing wonderful autumns and dreadful winters, and I still remembered the first time we met at the Career Fair; Yiyang (Tony) Ou for being a good pal in computer science and for our great memories competing for ICPC programming contests.; Dr. Disheng Xu for being a great teacher of mathematical analysis first and for being a close friend afterwards; Xiao Zhang for helping me with computer vision research and for encouraging me on graduate applications; and Yujie (Annika) Zhang, Liu Cao, Can Liu, Yulun Wang, Hao Mou, Shangxi (Richard) Zhang, Hanjue Zhu, Shuyan Xu, Yushi Hu, Miaochen Jin, Zijian Wang, Tianrui Hou, Yuan Luo and Lijia Zhou for all the days and nights we fought on solving mathematical, statistical or computer science problems together.

I would like to thank my parents, Changming Fu and Hairong Chen, for giving me the opportunity and resources to learn, for caring me physically and mentally, and for understanding and supporting every decision I made. I would also like to thank my fiancée Yanfei Zhou for always being there and giving me energy whenever needed and for creating memorable moments ever since we met. Finally, I would like to sincerely thank our cat, Xiao-Long-Bao, for meaningful discussions.

Bibliography

[1] Aaron Adcock, Erik Carlsson, and Gunnar Carlsson. The ring of algebraic functions on persistence bar codes. *Homology, Homotopy and Applications*, 18(1):381–402, 2016. 9

[2] Ibraheem Alhashim and Peter Wonka. High quality monocular depth estimation via transfer learning. *ArXiv*, abs/1812.11941, 2018. 14, 20

[3] Rickard Brüel Gabrielsson and Gunnar Carlsson. Exposition and interpretation of the topology of neural networks. In *2019 18th IEEE International Conference On Machine Learning And Applications (ICMLA)*, pages 1069–1076, 2019. 24

[4] Rickard Brüel-Gabrielsson, Vignesh Ganapathi-Subramanian, Primoz Skraba, and Lenoidas Guibas. Topology-aware surface reconstruction for point clouds. *Computer Graphics Forum*, 39, 2020. 9, 15

[5] Rickard Brüel-Gabrielsson, Bradley J. Nelson, Anjan Dwaraknath, Primoz Skraba, Leonidas J. Guibas, and Gunnar Carlsson. A topology layer for machine learning. In *The 23rd International Conference on Artificial Intelligence and Statistics (AISTATS)*, 2020. 8, 9, 13, 15

[6] Emmanuel J. Candès, Justin K. Romberg, and Terence Tao. Robust uncertainty principles: exact signal reconstruction from highly incomplete frequency information. *IEEE Transactions on Information Theory*, 52:489–509, 2006. 13, 14

[7] Gunnar Carlsson. Topology and data. *Bulletin of the American Mathematical Society*, 46(2):255–308, 2009. 3

[8] Gunnar Carlsson. Topological pattern recognition for point cloud data. *Acta Numerica*, 23:289–368, May 2014. 3

[9] Gunnar Carlsson and Vin de Silva. Zigzag persistence. *Foundations of Computational Mathematics*, 10(4):367–405, 2010. 5

[10] Gunnar E. Carlsson, Tigran Ishkhanov, Vin de Silva, and Afra Zomorodian. On the local behavior of spaces of natural images. *International Journal of Computer Vision*, 76:1–12, 2007. 6

[11] Mathieu Carrière, Marco Cuturi, and Steve Oudot. Sliced Wasserstein kernel for persistence diagrams. In Doina Precup and Yee Whye Teh, editors, *Proceedings of the 34th International Conference on Machine Learning*, volume 70 of *Proceedings of Machine Learning Research*, pages 664–673. PMLR, 06–11 Aug 2017. 6

[12] Mathieu Carrière, Frédéric Chazal, Yuichi Ike, Théo Lacombe, Martin Royer, and Yuhei Umeda. PersLay: A neural network layer for persistence diagrams and new graph topological signatures. In *AISTATS*, 2020. 13, 15

[13] Chao Chen, Xiuyan Ni, Qinxun Bai, and Yusu Wang. A topological regularizer for classifiers via persistent homology. In *The 22nd International Conference on Artificial Intelligence and Statistics (AISTATS)*, 2019. 15

[14] Liang-Chieh Chen, George Papandreou, Iasonas Kokkinos, Kevin P. Murphy, and Alan Loddon Yuille. Deeplab: Semantic image segmentation with deep convolutional nets, atrous convolution, and fully connected crfs. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 40:834–848, 2018. 14, 25

[15] Ting Chen, Simon Kornblith, Mohammad Norouzi, and Geoffrey E. Hinton. A simple framework for contrastive learning of visual representations. *ArXiv*, abs/2002.05709, 2020. 9

[16] David Cohen-Steiner, Herbert Edelsbrunner, and John Harer. Stability of persistence diagrams. *Proceedings of the twenty-first annual symposium on Computational geometry*, 2005. 7

[17] David Cohen-Steiner, Herbert Edelsbrunner, and John Harer. Stability of Persistence Diagrams. *Discrete & Computational Geometry*, 37(1):103–120, Jan. 2007. 5

[18] Ekin Dogus Cubuk, Barret Zoph, Dandelion Mané, Vijay Vasudevan, and Quoc V. Le. Autoaugment: Learning augmentation strategies from data. *2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 113–123, 2019. 20

[19] Ekin Dogus Cubuk, Barret Zoph, Jonathon Shlens, and Quoc V. Le. Randaugment: Practical automated data augmentation with a reduced search space. *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops (CVPRW)*, pages 3008–3017, 2020. 20

[20] Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, Jakob Uszkoreit, and Neil Houlsby. An image is worth 16x16 words: Transformers for image recognition at scale. *ArXiv*, abs/2010.11929, 2021. 14, 24

[21] Herbert Edelsbrunner and John Harer. Computational topology - an introduction. 2009. 7

[22] Herbert Edelsbrunner, David Letscher, and Afra Zomorodian. Topological persistence and simplification. In *Foundations of Computer Science, 2000. Proceedings. 41st Annual Symposium on*, pages 454–463. IEEE, 2000. 13, 14

[23] David Eigen, Christian Puhrsch, and Rob Fergus. Depth map prediction from a single image using a multi-scale deep network. In *NIPS*, 2014. 14

[24] Karl Pearson F.R.S. Liii. on lines and planes of closest fit to systems of points in space. *Philosophical Magazine Series 1*, 2:559–572. 12

[25] Ross B. Girshick, Jeff Donahue, Trevor Darrell, and Jitendra Malik. Rich feature hierarchies for accurate object detection and semantic segmentation. *2014 IEEE Conference on Computer Vision and Pattern Recognition*, pages 580–587, 2014. 14

[26] Clément Godard, Oisin Mac Aodha, and Gabriel J. Brostow. Unsupervised monocular depth estimation with left-right consistency. *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 6602–6611, 2017. 19

[27] Ian Goodfellow, Yoshua Bengio, and Aaron Courville. *Deep Learning*. MIT Press, 2016. http://www.deeplearningbook.org. 13

[28] Kaiming He, Haoqi Fan, Yuxin Wu, Saining Xie, and Ross B. Girshick. Momentum contrast for unsupervised visual representation learning. *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 9726–9735, 2020. 9, 10

[29] Kaiming He, Georgia Gkioxari, Piotr Dollár, and Ross B. Girshick. Mask r-cnn. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 42:386–397, 2020. 14

[30] Kaiming He, X. Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 770–778, 2016. 19, 25

[31] Sepp Hochreiter and Jürgen Schmidhuber. Long short-term memory. *Neural Computation*, 9:1735–1780, 1997. 27

[32] Christoph Hofer, Florian Graf, Bastian Rieck, Marc Niethammer, and Roland Kwitt. Graph Filtration Learning. In *Proceedings of the 37th International Conference on Machine Learning*, pages 4314–4323. PMLR, Nov. 2020. ISSN: 2640-3498. 15

[33] Christoph D. Hofer, Roland Kwitt, and Marc Niethammer. Learning Representations of Persistence Barcodes. *Journal of Machine Learning Research*, 20(126):1–45, 2019. 13, 15

[34] Xiaoling Hu, Fuxin Li, Dimitris Samaras, and Chao Chen. Topology-Preserving Deep Image Segmentation. In *Neural Information Processing Systems (NeurIPS)*, page 12, 2019. 8, 15

[35] Zeyuan Hu and Julia Strout. Exploring stereotypes and biased data with the crowd. *ArXiv*, abs/1801.03261, 2018. 15

[36] Gao Huang, Zhuang Liu, and Kilian Q. Weinberger. Densely connected convolutional networks. *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 2261–2269, 2017. 14

[37] Fan Jia, Jun Liu, and Xuecheng Tai. A regularized convolutional neural network for semantic image segmentation. *ArXiv*, abs/1907.05287, 2019. 14

[38] Nick Kanopoulos, Nagesh Vasanthavada, and Robert L Baker. Design of an image edge detection filter using the sobel operator. *IEEE Journal of solid-state circuits*, 23(2):358–367, 1988. 18

[39] Kwangho Kim, Jisu Kim, Manzil Zaheer, Joon Sik Kim, Frederic Chazal, and Larry Wasserman. PLLay: Efficient topological layer based on persistence landscapes. In *Neural Information Processing Systems (NeurIPS)*, 2020. 13, 15

[40] Diederik P. Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *CoRR*, abs/1412.6980, 2015. 20

[41] Alex Krizhevsky, Vinod Nair, and Geoffrey Hinton. Cifar-10 (canadian institute for advanced research). 10

[42] Gustav Larsson, Michael Maire, and Gregory Shakhnarovich. Colorization as a proxy task for visual understanding. *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 840–849, 2017. 9

[43] Katrin Lasinger, René Ranftl, Konrad Schindler, and Vladlen Koltun. Towards robust monocular depth estimation: Mixing datasets for zero-shot cross-dataset transfer. *IEEE transactions on pattern analysis and machine intelligence*, PP, 2020. 14

[44] Yann LeCun, Léon Bottou, Yoshua Bengio, and Patrick Haffner. Gradient-based learning applied to document recognition. 1998. 14

[45] Jin Han Lee, Myung-Kyu Han, Dong Wook Ko, and Il Hong Suh. From big to small: Multi-scale local planar guidance for monocular depth estimation. *ArXiv*, abs/1907.10326, 2019. 14

[46] Jacob Leygonie, Steve Oudot, and Ulrike Tillmann. A framework for differential calculus on persistence barcodes. *Foundations of Computational Mathematics*, pages 1–63, 07 2021. 8, 9, 13, 15

[47] Tsung-Yi Lin, Michael Maire, Serge J. Belongie, James Hays, Pietro Perona, Deva Ramanan, Piotr Dollár, and C. Lawrence Zitnick. Microsoft COCO: Common objects in context. In *ECCV*, 2014. 6, 15

[48] Xiankai Lu, Wenguan Wang, Chao Ma, Jianbing Shen, Ling Shao, and Fatih Porikli. See more, know more: Unsupervised video object segmentation with co-attention siamese networks. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2019. 2, 25, 26

[49] Kisantal Mate. Backboned unet. https://github.com/mkisantal/backboned-unet, 2018. 19

[50] Nikola Milosavljević, Dmitriy Morozov, and Primoz Skraba. Zigzag persistent homology in matrix multiplication time. In *Proceedings of the 27th annual ACM symposium on Computational geometry - SoCG '11*, page 216. ACM Press, 2011. 9

[51] G. Naitzat, A. Zhitnikov, and L.-H. Lim. Topology of deep neural networks. *J. Mach. Learn. Res.*, 21(345):1–40. 3, 4, 15, 24

[52] Bradley J. Nelson and Yuan Luo. Topology-preserving dimensionality reduction via interleaving optimization. *ArXiv*, abs/2201.13012, 2022. 12

[53] Nina Otter, Mason A. Porter, Ulrike Tillmann, Peter Grindrod, and Heather A. Harrington. A roadmap for the computation of persistent homology. *EPJ Data Science*, 6(1), 2017. 3, 9

[54] Emanuel Parzen. On estimation of a probability density function and mode. *Annals of Mathematical Statistics*, 33:1065–1076, 1962. 12

[55] Deepak Pathak, Philipp Krähenbühl, Jeff Donahue, Trevor Darrell, and Alexei A. Efros. Context encoders: Feature learning by inpainting. *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 2536–2544, 2016. 9

[56] Federico Perazzi, Jordi Pont-Tuset, Brian McWilliams, Luc Van Gool, Markus H. Gross, and Alexander Sorkine-Hornung. A benchmark dataset and evaluation methodology for video object segmentation. *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 724–732, 2016. 26

[57] René Ranftl, Alexey Bochkovskiy, and Vladlen Koltun. Vision transformers for dense prediction. *Int. Conf. Comput. Vis. (ICCV)*, 2021. 14

[58] Olaf Ronneberger, Philipp Fischer, and Thomas Brox. U-Net: Convolutional Networks for Biomedical Image Segmentation. In Nassir Navab, Joachim Hornegger, William M. Wells, and Alejandro F. Frangi, editors, *Medical Image Computing and Computer-Assisted Intervention – MICCAI 2015*, pages 234–241. Springer International Publishing, 2015. 8, 13, 14, 15

[59] Olga Russakovsky, Jia Deng, Hao Su, Jonathan Krause, Sanjeev Satheesh, Sean Ma, Zhiheng Huang, Andrej Karpathy, Aditya Khosla, Michael S. Bernstein, Alexander C. Berg, and Li Fei-Fei. Imagenet large scale visual recognition challenge. *International Journal of Computer Vision*, 115:211–252, 2015. 14

[60] Roy Schwartz, Jesse Dodge, Noah A. Smith, and Oren Etzioni. Green AI. *Communications of the ACM*, 63(12):54–63, Nov. 2020. 13

[61] Evan Shelhamer, Jonathan Long, and Trevor Darrell. Fully convolutional networks for semantic segmentation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 39:640–651, 2017. 14

[62] Suprosanna Shit, Johannes C. Paetzold, Anjany Sekuboyina, Ivan Ezhov, Alexander Unger, Andrey Zhylka, Josien P. W. Pluim, Ulrich Bauer, and Bjoern H. Menze. clDice - a

Novel Topology-Preserving Loss Function for Tubular Structure Segmentation. In *2021 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 16555–16564, Nashville, TN, USA, June 2021. IEEE. 15

[63] Irwin Sobel and G. M. Feldman. An isotropic 3×3 image gradient operator. 1990. 18

[64] Robert Endre Tarjan. A class of algorithms which require nonlinear time to maintain disjoint sets. *Journal of Computer and System Sciences*, 18(2):110–127, Apr. 1979. 9

[65] Chad M. Topaz, Lori Ziegelmeier, and Tom Halverson. Topological data analysis of biological aggregation models. *PLoS ONE*, 10, 2015. 4

[66] Aäron van den Oord, Yazhe Li, and Oriol Vinyals. Representation learning with contrastive predictive coding. *ArXiv*, abs/1807.03748, 2018. 10

[67] Igor Vasiljevic, Nick Kolkin, Shanyi Zhang, Ruotian Luo, Haochen Wang, Falcon Z. Dai, Andrea F. Daniele, Mohammadreza Mostajabi, Steven Basart, Matthew R. Walter, and Gregory Shakhnarovich. DIODE: A Dense Indoor and Outdoor DEpth Dataset. *CoRR*, abs/1908.00463, 2019. 20

[68] Tongzhou Wang and Phillip Isola. Understanding contrastive representation learning through alignment and uniformity on the hypersphere. In Hal Daumé III and Aarti Singh, editors, *Proceedings of the 37th International Conference on Machine Learning*, volume 119 of *Proceedings of Machine Learning Research*, pages 9929–9939. PMLR, 13–18 Jul 2020. 9

[69] Zhou Wang, Alan Conrad Bovik, Hamid R. Sheikh, and Eero P. Simoncelli. Image quality assessment: from error visibility to structural similarity. *IEEE Transactions on Image Processing*, 13:600–612, 2004. 19

[70] Pavel Yakubovskiy. Segmentation models. https://github.com/qubvel/segmentation_models, 2019. 19

[71] Richard Zhang, Phillip Isola, and Alexei A. Efros. Colorful image colorization. In *ECCV*, 2016. 9

[72] Xiao Zhang and Michael Maire. Self-supervised visual representation learning from hierarchical grouping. In *Neural Information Processing Systems (NeurIPS)*, 2020. 14

[73] Dora Zhao, Angelina Wang, and Olga Russakovsky. Understanding and evaluating racial biases in image captioning. In *International Conference on Computer Vision (ICCV)*, 2021. 15

[74] Afra Zomorodian and Gunnar Carlsson. Computing persistent homology. *Discrete & Computational Geometry*, 33(2):249–274, 2005. 5

Appendix A

Examples of Topology in Internal Activations

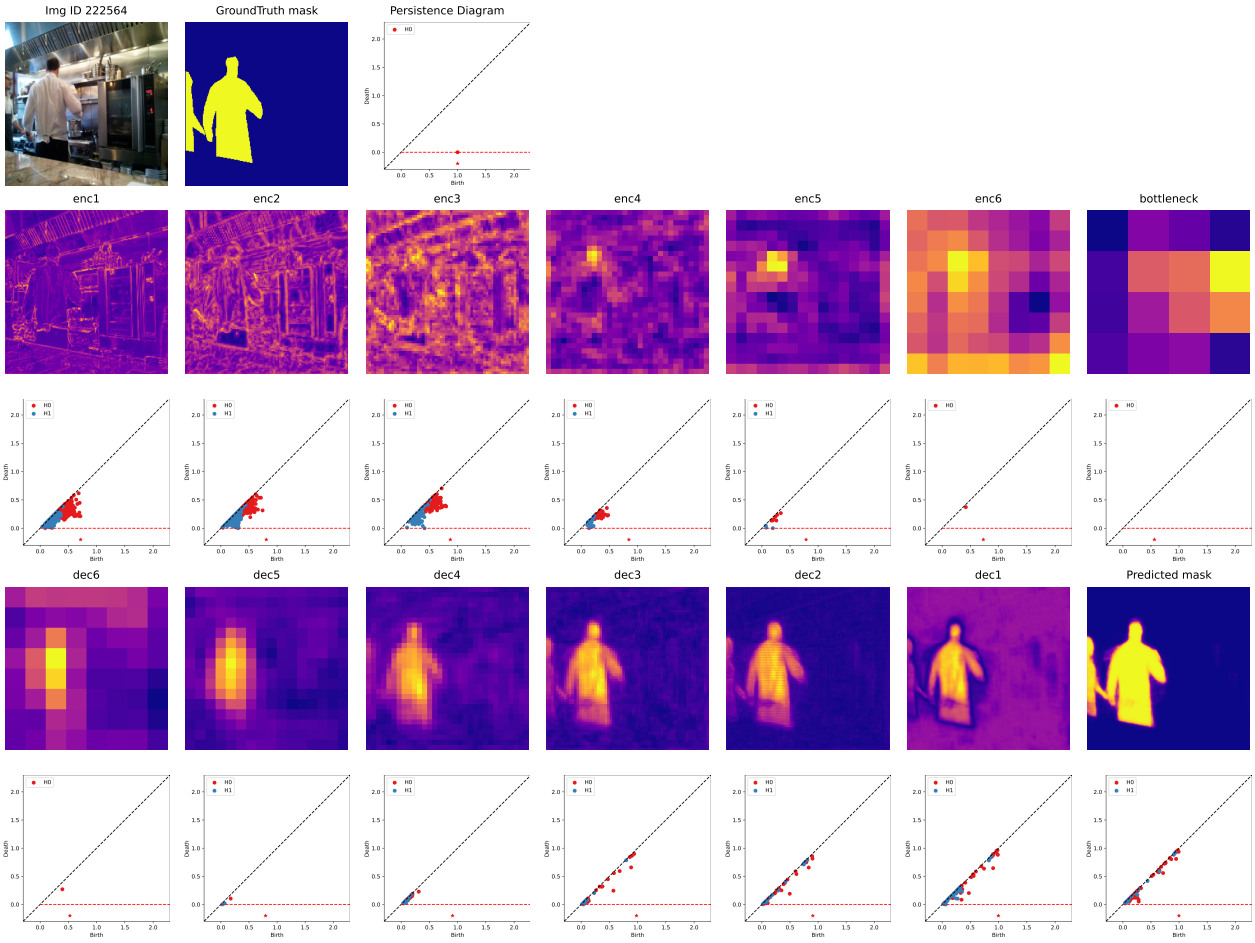


Figure A.1. Example containing two human objects. Persistence diagrams remain similar after layer dec4.

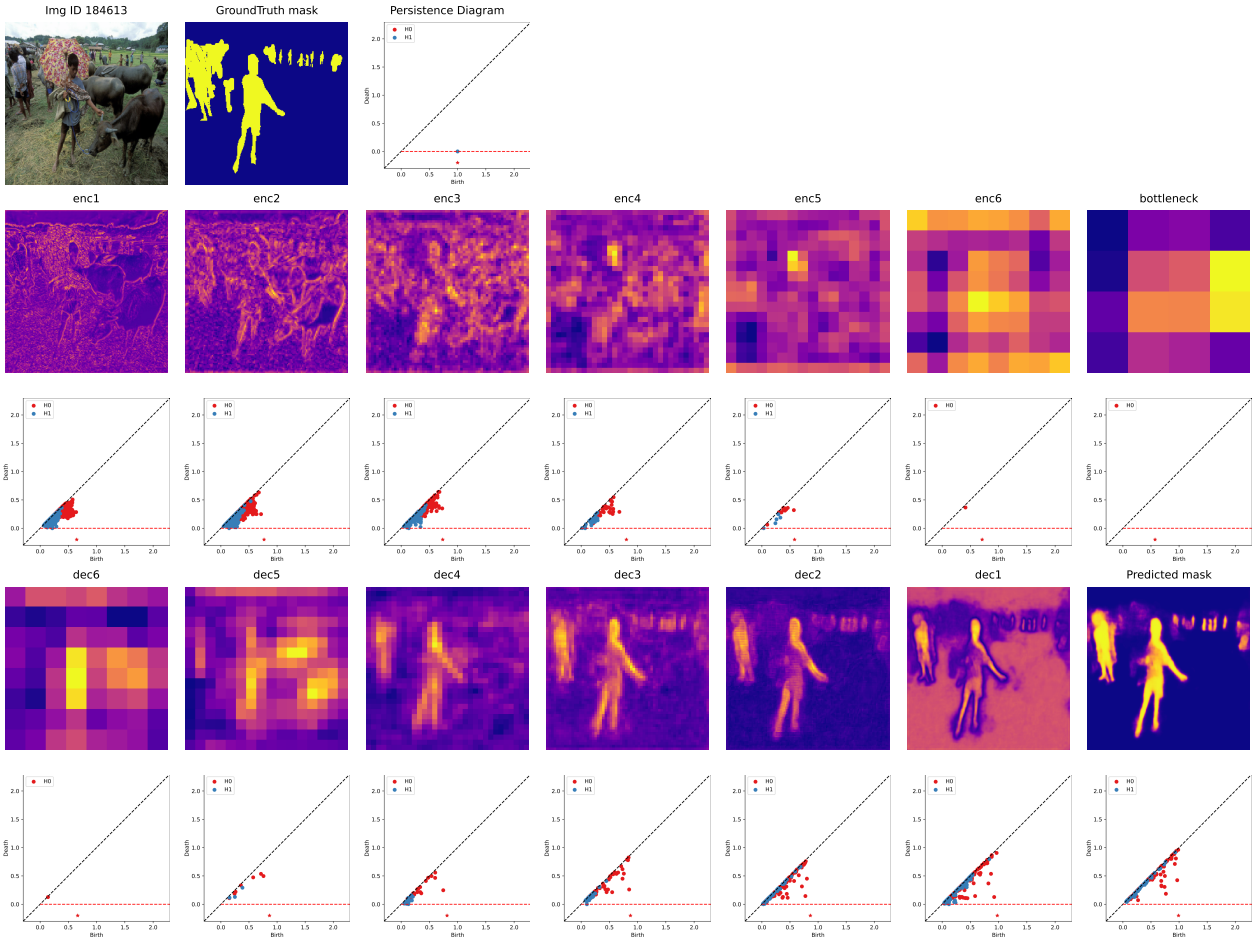


Figure A.2. Example containing multiple human objects. Persistence diagrams also remain similar after layer dec4.

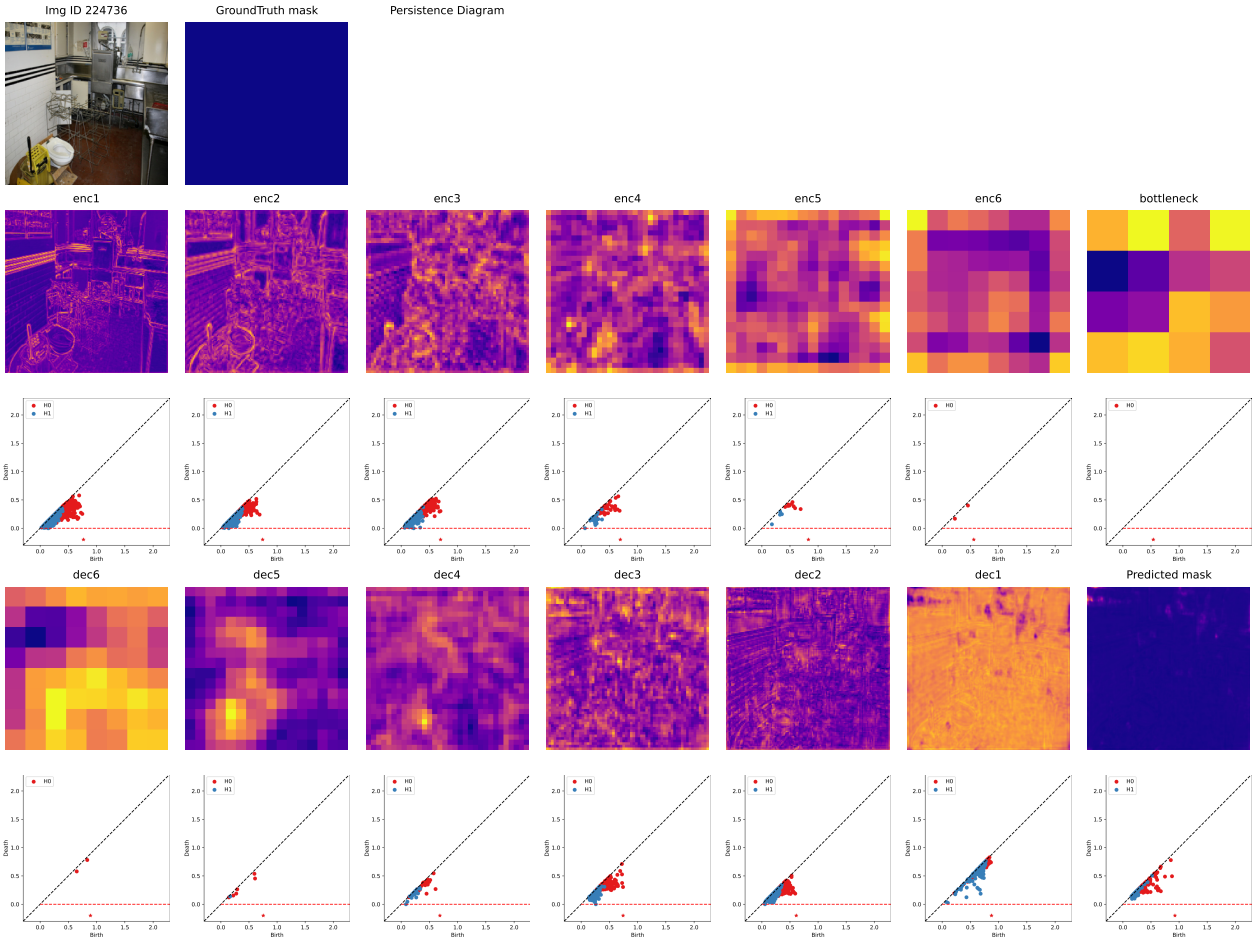


Figure A.3. Example containing no human object. Persistence diagrams change during decoder layers.

Appendix B

Regularization Effects of Internal Activations

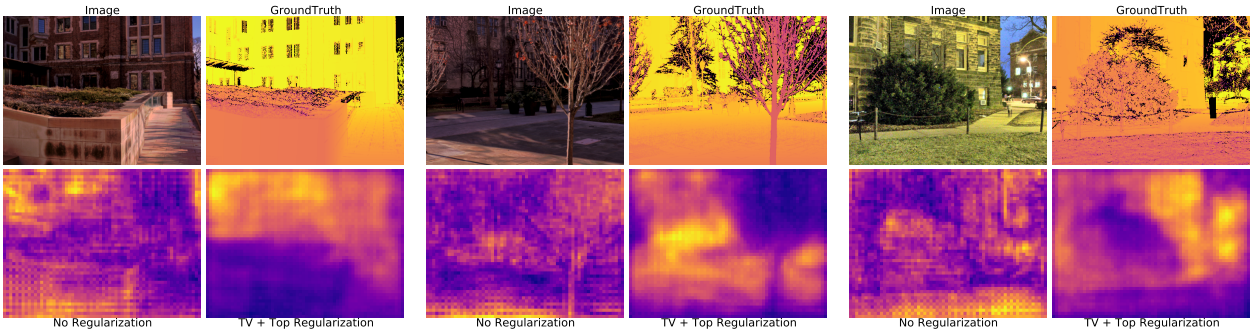


Figure B.1. Regularization Effects on U-Net.

The second row shows internal activations. Our regularized version helps reduce unpooling artifacts common in U-Net models. Most importantly, it keeps the interval activations concentrating on regions.

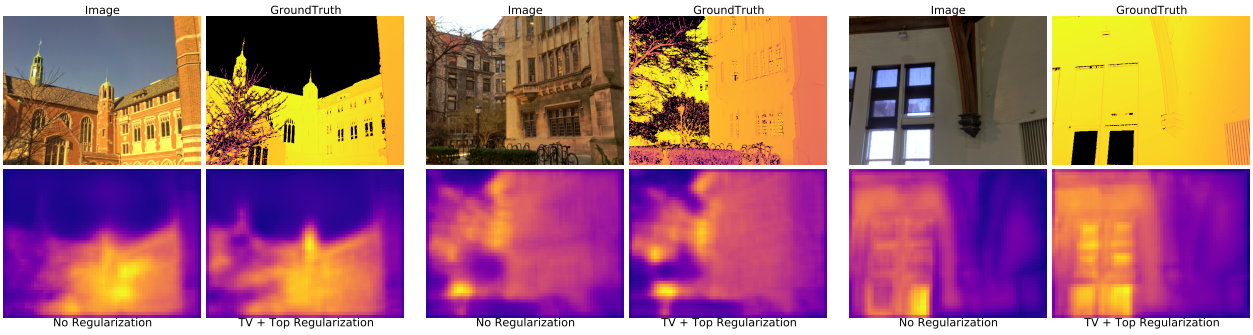


Figure B.2. Regularization Effects on DenseDepth.

The second row shows internal activations. Our regularized version also keeps concentration on regions and makes foreground and background objects more separable.