LEVERAGING TOPOLOGY TO ADVANCE MACHINE LEARNING MODELS AND METHODS

by

Samuel D. Leventhal

A dissertation submitted to the faculty of
The University of Utah
in partial fulfillment of the requirements for the degree of

Doctor of Philosophy

in

Computer Science

Kahlert School of Computing

The University of Utah

December 2024

The University of Utah Graduate School

STATEMENT OF DISSERTATION APPROVAL

The dissertation of **Samuel D. Leventhal**

has been approved by the following supervisory committee members:

Valerio Pascucci ,	Chair(s)	**10/11/2024** Date Approved
Christopher R. Johnson ,	Member	**10/03/2024** Date Approved
Dave Pugmire ,	Member	___ Date Approved
Bei Wang Phillips ,	Member	**10/04/2024** Date Approved
Aditya Bhaskara ,	Member	**10/07/2024** Date Approved

by **Mary W. Hall** , Chair/Dean of

the Department/College/School of **Kahlert School of Computing**

and by **Darryl P. Butt** , Dean of The Graduate School.

ABSTRACT

As machine learning evolves, the variety of data types suitable for learning is expanding, as are the domains where these techniques can be applied. Increasingly, data can be characterized by discrete elements and their spatial or functional relationships, making topological methods particularly effective.

This dissertation introduces new approaches to integrating topological priors in machine learning, remaining entirely within the topological domain. It generalizes topological priors by constructing a model-agnostic, learnable graph data structure that captures the relationships between cells within a complex, extending beyond simple adjacency to include their connectivity in image data. Focusing on image segmentation, we frame the problem as classifying topological cells within a Morse-Smale complex. By learning topological structures directly, without relying on preexisting topological knowledge, the model performs segmentation more accurately and efficiently than traditional pixel-based methods. An interactive labeling tool is also introduced, enabling practitioners to iteratively improve model performance by correcting predictions and retraining the model. This workflow supports applications across diverse fields, such as medical imaging, neuroscience, and materials science.

The dissertation also introduces the hierarchical priors graph (HPG), a sequence of graphs constructed from increasingly simplified Morse-Smale complexes. The reduced hierarchical priors graph (RHPG) further simplifies this by focusing on arcs between cells at different resolution levels. Using the RHPG, we propose a training method for graph neural networks (GNNs) called Hierarchical Successive Training (HST), which teaches GNNs to learn from multiple scales of connectivity by training sequentially on different prior graphs. Additionally, the Hierarchical Joint Training (HJT) message-passing scheme allows GNNs to pass messages within each prior graph and across graphs of different scales, improving learning efficiency.

Graph neural networks (GNNs) have shown significant potential in graph-structured data, particularly for tasks like node classification. However, GNNs struggle with challenges like heterophily, where nodes connected in a graph differ in class or structure. This dissertation addresses these

limitations with a novel hierarchical approach that leverages topological insights, particularly persistence filtration. By treating graphs as simplicial complexes and learning from class relationships between simplices, this approach captures multilevel subgraph hierarchies informed by topological summaries. It provides an ordered sequence of graphs that reveal both local and global structural information, offering a method resilient to oversmoothing and computational complexity. The proposed hierarchical GNN framework achieves competitive performance in both low and high heterophily scenarios, outperforming traditional GNNs in terms of accuracy and efficiency.

Overall, this dissertation introduces novel, accessible, and useful frameworks and methodologies for topologically informed machine learning and demonstrates practical benefits across various domains.

CONTENTS

LIST OF FIGURES

LIST OF TABLES

CHAPTER 1

INTRODUCTION

Many fields of study involve the analysis of complex images, which must be segmented to extract semantically meaningful objects for further investigation. Manual segmentation of images by domain experts is a time-consuming, labor-intensive, dexterity-requiring process. As a result, developing means of deferring the burden of segmentation through automated or semiautomated algorithmic simplification is of great importance. Motivated by abundant examples of the robustness and capability that machine learning (ML) models offer, computer-aided segmentation approaches have steadily been converging toward such models. A major obstacle in applying state-of-the-art ML approaches to scientific data is that often the data generated are first of its kind: no preexisting trained ML model is applicable, the objects represent a newly observed phenomenon, or the influence of image generation parameters such as sample staining or acquisition technology makes the image data different from prior applications. As a result, to adapt to variations in data acquisition or simply apply learning models to data across disciplines, a new model must be trained and often with specific nuances or considerable assumptions about the data in mind [1], [2]. Moreover, the need to obtain or generate good ground truth segmentations remains a significant roadblock, further compounding the difficulty of training robust learning models for segmentation.

An alternative solution for the segmentation task to ML has been the computation of mathematically defined objects, for instance, using scalar-field topology. In this setting, objects of interest are expressed algorithmically through topological abstractions such as elements of merge/contour trees or in Morse/Morse-Smale complexes (MSC) [3], [4]. Deterministic algorithms in this context are then applied to images to compute data structures that encode the geometric embedding of and connectivity between objects, called *topological elements*, within these topological abstractions. Scientific investigations often study collections of instances of phenomena in images, called *semantic objects*, burning cells from large-scale turbulent combustion [5], ocean eddies [6], neuron segmentation [7], ligaments in structural foams [8], and atomic structures [9]. We refer to the topological elements corresponding to semantic objects as the *semantic targets* for the segmentation

task. Computed topological abstractions then offer a geometric encoding of objects that enables higher level measurement and reasoning that answer scientific questions, such as bubble growth in turbulent mixing [4], the relationship between curvature and failure of foam struts [8], or estimating flow through porous materials [10].

Topological approaches successfully extract semantic objects in many applications, but a significant shortcoming has been bridging the gap between the theoretical description of objects and how they appear in real-world data. For instance, the neurons that constitute a brain wiring diagram can be modeled by the topology and embedding of the bright ridge-like features of the image. However, well-documented imaging artifacts that arise from uneven expression of fluorescence proteins and noise [11], attenuation [12], refraction and absorption [13], and many other sources, make straightforward computational identification of neurons difficult. Nevertheless, using topological elements as a scaffold for labeling accelerates user-guided segmentation compared to manual segmentation [7]. In this context, adding rapid inference could further reduce the burden of labeling – a gain that could be realized across domains and labeling tools.

Combining topological data analysis (TDA) and machine learning has already begun to demonstrate benefits in tasks related to classification and segmentation. Banerjee et al. showed that adding rasterized images of the MSC to a modified U-Net improved pixel classification tasks to segment neurons [14], [15]. However, pixel-level training labels must still be provided, and the final object segmentation must still be computed as a postprocess step. In contrast, we show that geometric objects themselves, as derived from topology, offer a high-quality representation for machine learning in that we can directly classify topological priors with results competitive to state-of-the-art approaches and without the need for post hoc object segmentation. We then continue to advance the use of geometric objects themselves for learning by exploiting the fact semantic objects appear as collections of data structures encoding the topological abstraction. For example, vertices adjoined by higher dimensional polylines result in a graph representation of a topological complex. Using this graph representation, we formulate the task of image segmentation as a node classification problem, which we solve using graph neural networks (GNNs). Instead of deriving only one high-granularity complex, we model multiple scales of topological information in an image using a nested hierarchy of graph representations, each derived from different levels of topological persistence.

We present an approach for representing objects composed in images as a topologically infor-

mative graph data structure for downstream machine learning tasks. We illustrate this approach using an interactive tool we have developed that allows a user to rapidly label topological priors for training various machine learning models, which extend the user's intended labeling onto the remainder of topologically encoded objects. Training and predicting in the topological domain performs as well as state-of-the-art image segmentation techniques operating in the pixel domain while requiring significantly less training time. Moreover, by remaining in the topological domain where the user has provided their labeling, we obtain the needed segmentation result directly rather than having to be constructed post hoc as is, for instance, with skeletonization of the pixel segmentation.

In summary, the contributions presented in this dissertation are as follows:

1. We present a methodology and workflow that combines topological data analysis with machine learning for learning topological priors.

   We modify the topological priors graph, a learnable, model-agnostic data structure representing cells within a complex and their relationships beyond adjacency, such as their affiliation with the connectivity of imaged objects. This graph consists of dimension-independent representations of cells in the Morse-Smale complex and their incidence relationships, tailored for specific classification tasks for any learning model. We demonstrate the practical benefits of using simplicial complexes for learning with a focus on image segmentation, where labeling, training, and inference occur in the topological domain over topological priors derived from cells in the Morse-Smale complex.

   - Reframes segmentation tasks as classification tasks.
   - Maintains the learning and classification task both in the topological domain, rather than only leveraging topological information to aid learning models restricted to the data domain.
   - Identifies the subset of topological cells corresponding to the segmentation of the object of interest.
   - Develop a learning process that generates complete topological priors by labeling all cells of the Morse-Smale complex from an initial partial labeling.

2. We introduce a novel and useful modified priors graph. We modify the priors graph by allowing relationships (arcs) between cells that differ in dimension by more than 1.

3. We demonstrate our methodologies with applications in image segmentation.

   Our methodology presented here demonstrates the benefits of using simplicial complexes for image segmentation, positioning labeling, training, and inference within the topological domain by leveraging topological priors from Morse-Smale complex cells. What is more, we also introduce a novel active learning workflow.

   - Demonstrates the benefits of using simplicial complexes for image segmentation.
   - Positions labeling, training, and inference within the topological domain using topological priors from Morse-Smale complex cells.
   - Novel active learning workflow.

4. We present a novel, efficient, and useful interactive labeling tool and active learning framework based on topological elements.

   We also introduce an interactive labeling tool for creating topological summaries and labeling topological cells to populate the training set. This workflow enables practitioners to correct predicted outputs and retrain the model, improving its performance. We compare topological prior object-level predictions to pixel-level predictions across various domains, including medical, neuroscience, and materials science, demonstrating improved accuracy, faster labeling, training, and segmentation compared to traditional pixel-based methods. Our work highlights the integration of topological elements into learning and showcases their practical benefits.

   - Interactive Labeling Tool Features
     - Interactive labeling that restricts the labeling process to the labeling of topological elements, therefore increasing precision and speed and avoiding the need for pixel-level precision.
   - Performance Assessment

  - Introduce a new method for comparing topological prior object-level predictions to pixel-level predictions.
  - Evaluates performance across medical, neuroscience, and materials science domains.
  - We demonstrate how topological priors offer improved segmentation accuracy and faster ground truth labeling, training, and segmentation in both advanced and basic machine learning models, compared to standard or state-of-the-art pixel-based models.

- Active Learning Workflow
  - Allows practitioners to contribute to the process of creating informative topological summaries.
  - Enables practitioners to correct and improve model predictions for retraining.

Recent advancements in machine learning, particularly with graph neural networks (GNNs), have shown great promise but face several key challenges. GNNs, which process data in graph structures using message passing and aggregation, encounter scalability issues as graph size increases, leading to high computational and memory demands, which is problematic in large networks, such as social or biological networks. Another issue is oversmoothing, where repeated message passing makes node representations too similar, losing discriminative power. Additionally, heterophily in graphs challenges GNNs that typically assume connected nodes are similar, yet real-world graphs often contain very different connected nodes. GNNs also struggle with imperfect graph data, as they usually assume accurate and complete graph structures. Overcoming these obstacles—scalability, oversmoothing, heterophily, and imperfect data—requires innovative model design, training algorithms, and theoretical insights.

In the realm of topological computing, persistent homology in topological data analysis (TDA) is stable under small data perturbations but sensitive to noise and outliers, leading to spurious topological features and false connectivities. Selecting the right scale for studying persistence poses another challenge, requiring domain-specific knowledge with no systematic approach for parameter selection. A key limitation in TDA, including persistent homology and Morse-Smale complexes, is translating topological information into actionable insights. Although these methods uncover

significant data structures, applying these insights to real-world scenarios remains challenging. Incorporating geometric and topological understanding into learning models has the potential to address challenges in computer vision tasks, such as occlusions and variations in object appearance, providing an approach particularly beneficial in complex scenes and precise medical imaging segmentation, enhancing machine learning model robustness, accuracy, and generalizability.

The contributions presented in this dissertation argue that using topology in GNNs to understand geometric, connectivity, and multilevel subgraph hierarchies, enhances their performance in tasks like object classification and image segmentation. These advancements address limitations in GNNs' expressiveness, particularly in identifying complex graph structures and higher order interactions. Incorporating local geometric information into GNNs can help capture spatial relationships more accurately. Using multilevel subgraph hierarchies allows GNNs to process information at various scales, potentially solving the oversmoothing problem and managing heterophily more effectively.

The work presented here also suggests that topological representations enable optimal performance in models such as random forests, which are less sensitive to feature space noise - an attribute common in imaging data where robust topological representation computation faces challenges due to perturbations and artifacts. The excelling performance of random forests contrasts the performance that was expected of GNNs. Namely, GNNs were anticipated to surpass other models due to their feature aggregation capabilities of node neighborhoods, making them aware of the connectivity and neighborhood structure of graphs. This dissertation presents solutions through approaches that learn and leverage connectivity to identify geometric- and class-based connected components. Combining edge prediction with message passing in learning models could inform information aggregation based on correlated geometries, enhancing model expressiveness.

The methodologies presented in this dissertation on GNNs explore various innovative approaches, each aimed at enhancing their performance and applicability. One such approach involves using simpler learning models to preinitialize GNN node embeddings, a strategy that promises to improve the accuracy and computational efficiency of GNNs. Additionally, this research direction includes delving into edge classification and filtration techniques that focus on heterophily, potentially leading to GNNs that are both more expressive and accurate. This focus is particularly pertinent in developing topological summaries within GNNs, targeting their existing challenges in adequately expressing specific local structures and effectively managing sparse topological data.

The novelty and usefulness of contributions presented in this dissertation can be further highlighted as follows:

5. We present a novel representation for hierarchical learning based on topological persistence sequences, denoted the hierarchical priors graph.

   We introduce the hierarchical priors graph (HPG), which combines a sequence of prior graphs obtained with successive simplifications of a Morse-Smale complex, with arcs connecting nodes corresponding to the same cells at different resolution levels.

   - Hierarchical Priors Graph (HPG)
     - Union of prior graphs $(PG_1, PG_2, \cdots, PG_n)$ from simplifications of a Morse-Smale complex.
     - Includes arcs connecting nodes representing cells at different levels of resolution.
   - Reduced Hierarchical Prior Graph (RHPG)
     - Reduction of the HPG with only arcs connecting nodes representing the same cells surviving one step of simplification.
     - Facilitates novel initialization-based training for graph neural networks (GNNs), termed Hierarchical Successive Training (HST).
   - Hierarchical Successive Training (HST)
     - Standard GNNs learn from multiple scales of connectivity by sequential training on P prior graphs.
     - Preserves neighborhood information between prior graphs using learned node representations across successive simplifications.
   - Novel Message Passing Scheme
     - Introduced based on HPG, with message passing within each PG_i and between multiple levels of prior graphs.
     - Denoted Hierarchical Joint Training (HJT), this dissertation introduces a message passing scheme that aggregates neighborhood information within and between prior graphs.

6. We extend hierarchical learning to learn over hierarchical representations of graphs obtained through topological filtration with improved results over state-of-the-art approaches and resilience to oversmoothing.

   Given a simplicial complex, a subset of the vertices have a label of any number of classes, and all have a feature vector. A simplex is then assigned with a derived binary 0/1 label as heterophyllous or homophilous if its boundary vertices have different or matching labels. Each simplex also obtains a feature vector derived as a combination of the feature vectors of its vertices. To build a filtration, the methodology presented in this dissertation uses the output of a learning model as a filter function trained on a subset of the complex with labeled simplexes. The predicted output value of the model assigned to each simplex is then used to sort simplices during topological filtration.

   - Addressing Heterophily
     - Introduces a filter function to predict the likelihood of edges being heterophilous.
     - Utilizes topological filtration based on predicted edge assignments to create an ordered sequence of subgraphs.
   - Understanding Graph Connectivity
     - Filtration helps track the evolution of graph connectivity and class connectivity over time.
     - Captures the appearance and merging of connected components and cycles.
   - Novel Methodologies
     - Introduced in this dissertation are two different methods for improving backbone GNN models, especially on graphs with heterophily, by allowing them to learn from a multi-scale sequence of graphs based on class connectivity and topological filtration.
     - Multi-Scale Successive Training (MsST): Trains GNNs for N/P epochs on each graph in the filtration sequence.
     - Uses embeddings from previous graphs to initialize node embeddings in subsequent graphs.

  - Multi-Scale Joint Training (MsJT): Performs message passing within each graph and between graph levels with common nodes presenting a novel message passing scheme.
  - Introduces a learnable node filter function using a graph isomorphism network (GIN-ε) for each graph level prior to the multi-scale aggregation scheme introduced in this dissertation [16].
  - Using the node filter function scheme presented in this dissertation, we present two novel methodologies for learning an informative node filtration for MsJT. The first is a novel multi-scale attention-based aggregation scheme between multiple levels of graphs in the graph filtration sequence, and the other is a learnable filtration of nodes in the nested sequence of graphs used in MsJT.
  - This dissertation presents a methodology for obtaining a sequence of nested graphs and a learned attention or learned filtration of nodes in that graph sequence to serve as a relaxation on graph neighborhoods and means to learn an informative graph representation, improving performance for backbone GNNs.
  - This dissertation experimentally demonstrates that the methods introduced lead to improved node classification accuracy relative to previous models .

- Combating Oversmoothing
  - Both methodologies aim to reduce oversmoothing by enhancing the influence of early birth nodes in subgraphs with higher co-class connectivity.
  - MsJT further increases the impact of informative relationships while filtering out uninformative ones through a learnable attention scheme.
  - This dissertation presents a homophily-based topological filtration method for graphs, where heterophilous edges may be used for learning but given different emphasis compared to homophilous edges. Our filtration preserves topological structure compared to approaches that simply drop heterophilous edges.

CHAPTER 2

BACKGROUND

We begin by describing both the mathematical underpinnings and common computational tools for our topological data analysis. Then, we review work on image segmentation and discuss how topological data analysis has started to be used in this context.

2.1 Computational Topology

Topological abstractions have been applied across multiple domains, such as segmentation of neurons [7], structural components of interest in metallic foams [8], eddies in ocean currents [6], bubble formation in mixing fluids [3], and ignition kernels in combustion [4]. In each case, semantic objects appear as elements (or collections thereof) of the data structures encoding the topological abstraction. Here, we present the relevant background to the Morse-Smale complex (MSC), the topological abstraction whose elements we use to generate "priors" objects for ML-assisted image segmentation.

2.1.1 The Morse-Smale Complex

A Morse function $f : \mathscr{M} \to \mathbb{R}$ is a smooth function on a manifold with nondegenerate, distinct critical points. According to the *Morse Lemma*, in a local neighborhood around a critical point b, f takes on a quadratic nature and can be written as $f(x) = f(b) \pm x_1^2 \pm ... \pm x_d^2$, with d being the dimension of $\mathscr{M}$. The number of subtracted x_i in this representation of $f(x)$ around b gives the number of "decreasing directions" from the critical point and is known as its *index*. For instance, in two dimensions, *minima*, *saddles*, and *maxima* are indices 0, 1, and 2, respectively. The gradient, ∇f defines a vector field whose zeroes are critical points. *Integral lines* are paths tangent to ∇f with lower and upper limits at critical points of f. Each noncritical point in the domain $\mathscr{M}$ belongs to a single integral line that has upper and lower limits at critical points, called the *destination* and *origin*, respectively. The partition of the domain formed by continuous clusters of integral lines sharing a common origin and destination defines the MSC. *Cells* of this complex have a dimension equal to the

difference between the index of the destination and origin critical points of their constituent integral lines. Figure 2.1 (a, b) shows a scalar function and its corresponding MSC, and the relationships between cells. The *1-skeleton* of the complex is formed by critical points, *nodes*, and the integral lines that connect critical points, *arcs*, that differ in index by 1.

2.1.2 Discrete Morse Theory

Concepts from continuous functions can be applied to a discrete pixel space following an approach based on discrete Morse theory [17]. Instead of a continuous manifold $\mathscr{M}$, the discrete 2-D domain consists of a mesh K whose cells are formed by vertices at pixels of the image along with edges and quadrilaterals of a regular grid. A discrete gradient field on K with critical cells, discrete gradient arrows, and discrete V-paths replaces critical points, ∇f, and integral lines, respectively. A discrete MSC that is structurally indistinguishable from a continuous MSC is obtained by tracing discrete V-paths, starting and ending at critical cells. In this study, we use the open-source MSCEER library [18], implementing steepest-descent [19], [20] and accurate-geometry [21] discrete gradient construction algorithms, and discrete MSC computation.

Persistence plays a hierarchical role in the degree to which the MSC encompasses a semantic object, where low persistence can be attributed to a low granularity in the MSC and vice vera. All ascending and descending arcs, however, are not necessarily relevant. The semantic target often consists of the higher intensity valued pixels in imaged data. As a result, descending paths adjoining noncritical stationary points, or 1-saddles, to minima can be considered not to cover the semantic target. For this reason, we remove 1-saddle points between critical maxima and minima along with the adjoining path. Preserving paths between stationary 2-saddles ascending to maximum critical points allows us to capture all edge and adjoining ridge-like structures that are pertinent to the semantic target. MSCEER also supports computing the MSC at a user-specified persistence simplification threshold [22].

2.1.3 Topological Simplification

Topological abstractions come equipped with well-understood techniques to order and simplify their elements to obtain successively coarser representations. For example, *topological persistence* pairs a critical point that creates a topological feature with the critical point that destroys that feature during a *filtration* [23]. The time span in the filtration in which the feature lives, i.e., the difference

in function value between the birth and death critical points, is called persistence. Intuitively, small, local perturbations (low persistence critical points) usually correspond to noise or artifacts in image acquisition or from discretization.

Many approaches exist for achieving simpler topological abstractions: the image can be locally perturbed (smoothed) such that the computed MSC will be coarser [24], [25]; the critical cell pairs in the discrete gradient field can be canceled through path reversal [17]; or the MSC can be directly simplified by successively canceling nodes connected by a single arc in the 1-skeleton [22], [23], [26]. Practically, this last approach builds a multiscale data structure that allows interactive adjustment of the simplification threshold, which enables a user to fine-tune the simplification level based on the data and task at hand. Figure 2.1 (e, f) illustrates the use of simplification to remove excess nodes and arcs from the 1-skeleton of the MSC.

2.1.4 Ridge/Valley Graph

In many applications, the 1-skeleton of the simplified MSC places nodes/arcs in a manner that covers the semantic objects of interest. However, its use as a "scaffolding" for further analysis or semantic object extraction might require modification of the structure. Integral lines for continuous functions do not merge, but the limited resolution available to discrete methods may merge V-paths, effectively creating overlapping arc segments such as in Figure 2.1 (g). Often, nonoverlapping edges are desired, for example, to enable a mapping from image pixels to unique components in a more manageable graph structure. Mcdonald et al. [7] introduced a refinement of the MSC 1-skeleton, called the *ridge graph*, that collected the mesh cells constituting the 1-skeleton, and created a new graph whose vertices were those cells without exactly two adjacent neighbors, and whose edges were the sequence of cells with exactly two adjacent neighboring cells. This ridge graph could be further refined by creating vertices for each critical cell of the 1-skeleton, splitting the arc into two edges. Figure 2.1 (h) shows this transformation of the 1-skeleton into the ridge graph (without the optional critical cell refinement). Note that ridge or valley graphs are constructed in the same way, only taking as input either the saddle-maximum or minimum-saddle arcs of the MSC 1-skeleton.

2.2 Image Segmentation

Despite the value of image segmentation to science, medical, and engineering disciplines, it remains a laborious and time-intensive task [27]–[29]. The difficulties in image segmentation result

from the scope of the domain, the amount of data needing to be segmented, and the reliance on field expert experience to properly identify semantic objects. The need for solutions to overcome these obstacles continues to grow as new imaging techniques develop and the volume of image data needing to be segmented increases.

To address obstacles in image segmentation, contemporary approaches have begun to incorporate machine learning into the segmentation task. Such methods have shown promising progress, such as with the use of U-Nets [14]. Traditional pixel-based approaches require a representative set of manually labeled ground truth data provided by field experts or an experienced eye. Manually segmented data is then used for typically time-intensive training followed by inference on unseen image data. The inferred pixel predictions then correlate to the class, or probability of the respective pixel belonging to the semantic object. Lastly, a geometric summary of the pixel predictions is performed to glean the final segmentation.

Recent works have shown that informative gains can be obtained in moving beyond per-pixel feature statistics by generalizing to superpixels [30], [31]. By generalizing pixels into groups with shared characteristics such as intensity and proximity, superpixels introduce the opportunity to assign class labels and derive feature statistics more intelligently. Konyushkova et al. employ superpixels to extract novel feature statistics as well as demonstrate the benefit of informing the learning process with geometric priors for image segmentation by introducing a geometric uncertainty measure that intelligently guides a user during active learning [32]. Chen et al. have also recently shown that shape-driven approaches improve the reliability and accuracy of segmentation results. They accomplish this by employing a deep shape Boltzmann machine as a generative model to extract the architecture of shapes during training, which, when used as a shape prior term within an objective function, later minimized by learning models, affords improved accuracy in tracking shape deformations during variational segmentation [33], [34].

2.3 Topological Segmentation

2.3.1 Topology for Digital Images

Segmentation tasks across image domains, as a first step, often convert the native image representation (e.g., RGB) to a single scalar value. For example, object detection in digital imagery can be successfully done by first converting multichannel image data to grayscale, applying a Sobel filter, and computing watershed regions [35]–[37]. Similarly, to apply the topological framework,

persistent homology has been used to understand root architecture from images better, identify cells in microscopy images, and much more [38]–[40].

2.3.2 Skeletonization

For the purpose of analysis and visualization of volumes, a simplified representative structure of shape is important. The exemplary curve-skeleton of the larger object is often interpreted to be a line simplification of the full volume, tracing the objects's center[41]. Various approaches to achieve this simplified skeleton include topological thinning[42], distance-field-based methods [43], [44], and potential field-based methods [45], [46]. Previous distance-field approaches to obtain a skeletonization have been based on penalizing a Dijkstra Shortest Path toward the boundary of the object from a core central line[43], [47], the outward flux of gradient vectors from the center of the material[44], or computing the center line through level sets [48]. The interpretability of what defines a representative skeleton has led to numerous approaches of skeletonization as well as attempts at a formal definition [45], [49], [50]. Often skeletonization algorithms approach the problem by starting outside and working their way in, i.e., computing the center line of the full object through erosion [51], thinning [52]–[56], and dilation from the object boundary[57]. A challenge when using these approaches to segment digital images is that the threshold determining the object interface completely determines the connectivity of the skeleton. There is no ability to estimate what components have been omitted. Another limitation of these approaches is that they do not relate the components of a skeleton to the void space outside the manifold of interest, for instance in connecting which portions of a skeleton surround a void or hole.

Of growing interest is the use of geodesics for constructing higher dimensional space summaries as lower dimensional path connections between points of interest or minimum spanning trees that convey significant structures within the higher dimensional shape. Previous approaches for determining these curvilinear structures have relied on computing distances from origin points using the fast marching method while minimizing an energy function or relevant measure [58] or iteratively moving from origin points to sink points through occlusion points of interest[59]. One such implementation used in neurite tracing determines geodesics as those propagated from the origin point that minimize a tubularity measure defined to be a spatial path as well as a curvilinear 'thickness' given by intersecting closed balls [60]. Performing image segmentation for holes, ends, and centers through mathematical morphology of geodesics has been of growing interest due to its

success and intuitive physical meaning, allowing easy extension of geodesic morphological transformation techniques from 2-dimensional image data into 3-dimensional volumes or 4-dimensional time dependent spaces [61]. Although skeletonization can be performed in geodesic space [62], our primary interest in it is a means of measuring properties from *inside* a topological space, specifically, the manifold pertaining to the metallic material.

Recently, approaches have begun to compute the curved skeleton through topological summaries such as Reeb graphs and merge trees based on persistence [63], [64]. The most successful methods employ the Morse-Smale complex directly on the function of interest or the signed distance field from a material interface. This approach has been used to construct the filamentary structure of the universe [65], represent bonds between atoms [66], trace lithium diffusion pathways [67], and extract the core structure of a porous material [68]. An advantage of topology-based techniques is that they account for noise's impact through persistence simplification [69]. We use the 1-skeleton of the Morse-Smale complex, as it enables reasoning about the stability of the extraction and provides a means of relating junctions and ligaments to the grains and open faces they surround.

2.3.3 Morse-Smale Complex Computation and Simplification

The MSC has been well established to extract the 1-skeleton of a distance field to the interface surface of a porous solid [68]. A key aspect that makes it attractive is the ability to simplify the representation concerning noise and to explore the reconstruction as thresholds are varied interactively [22]. We use a parallel discrete gradient algorithm [21] and MSC computation in the open-source library MSCEER [18]. The MSC of the signed distance field, without any simplification, contains a superset of topological features that may be undesired or due to noise, as shown in the example of foam materials (see Figure 2.2a).

Finding the right amount of persistence simplification to produce a skeleton consistent with the input desired structure introduces a new challenge. In other words, the 1-skeleton is most expressive when it includes all and only the arcs connecting maxima that correspond to the geometries of the object of interest, for example the junctions in the foam. To find the right persistence to simplify the complex, removing noise while keeping the main ridge-like structures becomes a critical step. The 1-skeleton is formed by the critical points and the 2-saddle-maximum arcs (ridge-like structure) that remain after simplification as shown in Figure 2.3. In practice, exact matching is unnecessary; instead, the 1-skeleton can be used as a graph embedded in the domain that illuminates the

connections between junctions and their pathways.

2.4 Topological Features

2.4.1 Feature Tracking

A powerful ability afforded by skeletonizations is tracking components of the 1-skeleton over a time series. Overlap between features in subsequent time steps is central to many approaches and is used to categorize events such as continuation, creation, dissipation, bifurcation, and amalgamation [70], [71]. Such approaches rely on spatial overlap, and they have been extended to topological tracking of contours [72], for varying thresholds with Reeb graphs [73], space-time isosurfaces [74], or building a space-time function to track features defined on level sets [75].

By providing a robust and concise feature description, skeletonization offers a new way of feature tracking and matching. A hierarchical view of the computed spanning tree object makes comparison possible using isomorphic or nearly similar skeleton trees [76], [77]. This approach has also been employed in animation through computation of the centerlines, manipulation, and reconstruction of the original object [78]. Comparing objects based on their shape skeletons has also been used as a similarity measure for feature tracking over time [79] and feature matching following deformation [80], [81]

2.4.1.1 Feature Extraction

The quantities of interest often depend on the application case or interests of domain experts. The geometric realization of a ligament originates from the 1-skeleton of the MSC, which is computed using a discrete gradient approach. As a result, initial line segments on a ligament are aligned with axis orientation. Constrained iterative smoothing can be performed to remove these effects and obtain a smooth line for each ligament. Viable features the 1-skeleton affords are:

- Length of a ligament. From the 1-cell polylines, the accumulated Euclidean distance between adjacent points on the ligament is representative of the element's length.

- Curvature of a ligament. For computing curvature $\mathscr{C}$, consider a normal vector $\hat{n}_{p_i}$ to each point p_i along a ligament ℓ of length b. Adjacent points, p_i to p_{i+1}, are considered to curve proportionally to the cosine similarity of these normal vectors. The curvature of the ligament is then the sum of all cosine similarities between adjacent normal vectors along the ligament.

$$\mathscr{C}(\ell) = \sum_{i=0}^{b} cosim(\hat{n}_{p_i}, \hat{n}_{p_{i+1}})$$

- Orientation in 3-Dimensions. Similarly, the orientation of a ligament is defined to be the cosine similarity of the initial values of the 1-skeleton of a ligament to the $\hat{x}$, $\hat{y}$ and $\hat{z}$ axis vectors.

- Cross-section area and perimeter of a ligament. To collect cross-section areas and perimeters along several slices normal to the ligament along its path, as done with curvature, the first step is to obtain the normal vector from a voxel or pixel on the 1-skeleton to its neighbor. Using the normal vector and the perpendicular vector tangential to the ligaments 1-skeleton, it is then possible to compute a cross-sectional slice. Since multiple ligaments can be close, potentially almost touching, it is necessary to identify which ligament was initially sliced. For this reason, the ligament of interest is marked, and once the slice is obtained, we compute all connected components. The connected component with the marked voxel is then the cross-sectional slice.

2.5 Machine Learning

2.5.1 Shallow Learning

2.5.1.1 Random Forest

Random Forest is a versatile and powerful ensemble learning algorithm used for both classification and regression tasks. It combines the predictions from multiple decision trees to provide more accurate and robust results.

Random Forest starts by creating multiple decision trees, typically hundreds or even thousands of them. Each tree is trained on a different subset of the original dataset. Training subsets used are obtained through bootstrapped sampling, where data points are randomly selected with replacement from the original dataset to create a new training set for each tree. Some data points may appear in multiple subsets, whereas others may not be included at all. In addition to sampling the data, Random Forest also randomly selects a subset of features (columns) at each node of each tree, which ensures that the trees are diverse and not overly reliant on a single feature. The number of features to consider at each split is a hyperparameter, typically denoted as ”m” or ”max features.”

Each decision tree in the Random Forest is grown using a process similar to the standard decision

tree algorithm, like CART (Classification and Regression Trees). At each node, the algorithm selects the best feature among the randomly chosen subset and splits the data based on a criterion, such as Gini impurity for classification or mean squared error for regression. Once all the trees are constructed, they are used to make predictions. For classification tasks, each tree "votes" for the class it predicts, and the class with the most votes becomes the final prediction. In regression tasks, the predictions from all trees are averaged to obtain the final result.

Random Forest has several mechanisms to reduce overfitting. The combination of bootstrapped sampling and random feature selection introduces diversity among the trees, making them less prone to overfitting. Additionally, the majority voting or averaging process helps reduce the bias that can be present in individual trees.

Random Forest has several mechanisms to reduce overfitting. The combination of bootstrapped sampling and random feature selection introduces diversity among the trees, making them less prone to overfitting. Additionally, the majority voting or averaging process helps reduce the bias that can be present in individual trees.

2.5.2 Deep Learning

2.5.2.1 Multilayer Perceptrons

Multilayer Perceptrons (MLPs) are canonical feed-forward neural networks consisting of three fully connected layers of neurons, each with a nonlinear activation function, trained using backpropagation [82]. For our purposes, cross entropy loss was used with activations being logistic functions defined as $\sigma : (\mathbf{z}, \mathbf{W}) \mapsto \frac{1}{1+e^{-\mathbf{z}^T \cdot \mathbf{W}}} = p \in (0,1)$ for a given input $\mathbf{z}$ and weight $\mathbf{W}$ where $p \in \mathbb{R}$ of the final output can be taken as a probabilistic value of belonging to a given class in a binary setting. The loss function taking into account the logistic function for labels y becomes

$$L_{\text{CE}}(\mathbf{z}) = -y \cdot \sigma(\mathbf{z}^T \cdot \mathbf{W}) + log(1 + e^{\sigma(\mathbf{z}^T \cdot \mathbf{W})}).$$

2.5.2.2 Graph Neural Networks

Graph Neural Networks (GNNs), which generalize deep neural network operations such as convolutions to graph-structured data, have risen in popularity for graph machine learning. In contrast to hand-engineered feature construction or unsupervised vertex embedding methods, graph neural networks may be trained with task-specific objectives that produce multidimensional embedding representations of vertices and a mapping of labels to embeddings. During classification, inference

is then performed on unseen regions of the priors graph where node labels are inferred from their embeddings.

Starting with initial features for each vertex v, $\mathbf{x}_v$, graph neural networks learn features for each vertex v by repeatedly aggregating its own features and those of other nodes in a receptive field $N(v)$:

$$\mathbf{r}_v^{(k)} = f\left(\mathbf{r}_v^{(k-1)}, \{\mathbf{r}_u^{(k-1)} : u \in N(v)\}\right), \mathbf{r}_v^{(0)} = \mathbf{x}_v \tag{2.1}$$

Here f is a nonlinear function that is applied repeatedly in K total rounds of feature aggregation, with learnable parameters controlling the aggregation at each round. Most commonly, the receptive field $N(v)$ for each vertex v consists of its immediate neighbors, although it is sometimes beneficial to aggregate features from more distant nodes. This general formulation includes methods such as graph convolutional networks [83], GraphSAGE [84], and many others.

2.5.2.3 U-Nets

U-Nets are a well-established and notably robust neural network architecture for image segmentation [14]. The canonical U-Net architecture is comprised of two main components following an autoencoder-decoder paradigm. The first is a contractive path that reduces the input image's dimensionality at each layer. The encoded representation of the image is then passed to a decoder using upconvolutions and reduced channels to raise the dimensionality of the encoded embedding. The final layer uses 1×1 convolutional filters to map each component feature vector to the desired number of classes [14]. A unique aspect of the U-Nets architecture that improves its robustness is the use of skip-connections, in which the embeddings produced during encoding for any given layer k are later used again during the upconvolutional decoding stage. Specifically, for a depth K network, the embedding produced by layer l is concatenated with the decoding layer $(K-l)'s$ input embedding, resulting in the symmetrical architecture of the U-Net from which it gets its name. The result of this concatenation affords U-Nets their improved ability to learn segmentation information during supervised training [14].

2.6 Computational Topology for Machine Learning

Several recent works have used topological methods, specifically in the task of segmentation. Banerjee et al. [15] applied the MSC as an image-level prior to be used in a modified U-Net [14] architecture. They found that the rasterized representation of the simplified MSC concatenated

into an encoder-decoder network improved pixel-level classification in microscopy images. For the segmentation task of reconstructing roads from satellite images, recent work has eliminated the need for labeled data by employing a topological approach with improved results compared to other state-of-the-art approaches that were previously reliant on manually labeled training sets. They accomplish this by generating training samples for a Convolutional Neural Network (CNN) using a discrete-Morse graph reconstruction algorithm to identify road network connectivity [85].

TDA has also been incorporated into a neural network architecture to train deep learning segmentation techniques to conform to higher topological accuracy. Hu et al. [86] improved a model's topological accuracy by updating the neural network during training using an adapted loss meant to guide the model's predictions to more closely adhere to the 1-skeleton and 2-D cells of the MSC. Through their adapted loss, they can more heavily penalize the misclassification of pixels belonging to components, such as polylines, of the MSC during training. Similar modifications incorporating TDA into neural network architecture have shown promise for autoencoders and GNNs [87], [88]. Our work, in contrast to these methods, performs learning in the topological domain over priors, separate from the image space.

2.7 Prior Work in Topological Machine Learning

To give a perspective of the landscape populated by prior research and development of topological machine learning approaches, we provide a comprehensive but concise survey of past works that employ topological methods to enhance or augment classical and deep learning models. The overview presented is meant to not only be a robust summary but also a categorization of learning methods taking advantage of the multiscale, global, intrinsic properties, and more generally encoded geometric shapes captured by topological features with a sectional view of publications from major machine learning conferences and journals such as NeurIPS, AISTATS, ICLR, ICML, and JMLR.

Learning methods that integrate topological approaches can be framed in an informative taxonomy and broadly grouped as *extrinsic* and *intrinsic* approaches [89]. Methods categorized as extrinsic incorporate topological information through topological features, such as vectorization, or construct neural network layers equipped to handle topologically informed features. Topological features are derived from data for downstream machine learning models in this setting. In this setting, extrinsic methods do not perform topological analysis of machine learning models nor work to incorporate topological information into model architecture, in contrast to intrinsic approaches

that directly analyze or imprint topological features on learning models.

Additional subclassifications, denoted as either *observational* or *interventional* methods, can be further specified based on how a method incorporates topological information into the learning framework. Methods considered observational are those whose focus is on the topology of data or model as opposed to explicitly influencing model training or architecture. Interventional methods differ in that model architecture or training is topologically informed by direct topological information derived from data or the learning model. In most cases, we observe that intrinsic features tend to be interventional, and extrinsic features are generally observational.

When homological information is of interest, particularly when persistence diagrams are used, two main approaches by which extrinsic methods represent learnable topological features are representation mapping and kernel mapping of persistence diagrams. The motivation for approaches to first map topological summaries originates from the obstacle of using persistence diagrams directly as input for learning models due to their being multisets with limited metrics available. Representation mapping of topological information overcomes these difficulties via maps into auxiliary vector or function spaces. Similarly, extrinsic methods may use the feature map of kernel methods to map persistence diagrams into a domain, such as a Hilbert space, whose structure affords more manageable and less computationally restrictive metrics than those supported by persistence diagrams alone (such as the bottleneck distance). Both vector/function-based methods and kernel-based methods are often observational methods due to following the methodology of constructing manageable representations of topological information for classifiers to use. Often, the object of observation in these approaches is scalar-valued summary statistics such as the total persistence of a persistence diagram, persistence p-norm, or the Shannon entropy of the individual persistence values in a persistence diagram.

To illustrate approaches that explore the representational mapping of persistence diagrams, we discuss four techniques used in the literature that make use of vector/function-based methods: Betti curves [90], [91], high-dimensional feature vectors [92], persistence landscapes [93], and persistence images [94]. Betti curves, often used to analyze data or summarize descriptions of features, have grown in interest because they allow the calculation of a mean curve and provide an easily computed distance and kernel methods [95]. However, Betti curves have limited expressive power since, as a summary statistic of persistence diagrams, they do not afford an injective mapping from a persistence diagram to a curve, do not permit the tracking of single features, and only provide

counts of topological features.

Methods aiming to construct vector representations of persistence diagrams have done so through the pairwise distribution of elements within a persistence diagram. Vectorization of topological information, such as the construction of high-dimensional feature vectors, does, however, run the risk of poor scalability. Persistence landscapes employ an invertible map of persistence diagrams into a Banach- or Hilbert-function space in which stability properties concerning the bottleneck distance of persistence diagrams are maintained, allowing persistence landscapes to enjoy injective and stability properties. Persistence landscapes have also drawn attention because various summary statistics, such as norms and measures for distance and kernels, become available. The use of persistence landscapes to inform learning models has been achieved through vectorization by binning their domain. An important means by which persistence landscapes can be incorporated arises due to their lossless representation, allowing them to be integrated into machine learning algorithms in a differentiable manner. An example is the persistence landscape-based topological neural network layer introduced by Kim et al. [96].

Another vector-oriented extrinsic approach is persistence images, which offer a hierarchical vectorization of persistence diagrams [97]. Obtaining persistence images requires three noncanonical functional decisions to be made: a weighting function is chosen to accentuate features in the persistence diagram with a large value, a probability density function over "(birth, persistence)" points in $\mathbb{R}^2$ resulting from the linear transformation of a persistence diagram's "(birth, death)" points, and lastly, the resolution (or level of discretization) of the $\mathbb{R}^2$ codomain space constituting the pixels of the persistence image. Limitations of persistence images arise due to their quadratic storage and computational complexity, as well as the choice of appropriate parameters. Persistence images are, however, very flexible and can allow learning models to be more topologically aware. One of the limited number of approaches that is interventional with extrinsic topological features is done using persistence images derived from graph-based persistence diagrams for graph classification by learning task-based weights for each element in the standard grid [97]. The elements in the resolution hierarchy of persistence images from discretization of the density function on persistence diagrams can also be used as feature vectors.

Methods utilizing extrinsic topological features based on kernels are often motivated by either a desire to directly compare persistence diagrams or an interest in kernels derived from topological information. Extrinsic models that learn from observing the direct comparison of persistence dia-

grams often employ kernel feature maps for an alternate representation of persistence diagrams with a more manageable Hilbert space structure. Importantly, kernels, along with a feature map, defined over sets of persistence diagrams implicitly provide a vector representation of persistence diagrams through the feature map, such as persistence weighted Gaussians or lower dimensional projections via the slices Wasserstein distance [98]–[100]. Alternatively, some methods construct kernels from topological information, such as the Persistent Weisfeiler–Lehman (P-WL) kernel for graphs used for measuring graph similarity and for graph classification.

On the boundary between extrinsic- and intrinsic-based approaches for the correspondence of learning models and topological computing are recent works whose aim is to imbue topological descriptors into neural network layers. Notably, Hofer et al. construct a projection of parameterized persistence diagrams [101]. The implications of differentiable, and therefore learnable, topological descriptors are particularly interesting in that such descriptors can be integrated into neural network infrastructure. Works such as Hofer et al. continue to incentivize the construction of learnable topological representations. Similarly, Carrier et al. introduce PersLay, a neural network layer based on mapping, weighing, and vectorizing persistence diagrams [102].

The second category of topological machine learning models denoted as intrinsic, incorporates topological information into the learning model architecture or performs direct topological analysis of the machine learning model. In recent works, intrinsic approaches that integrate or apply topological analysis into or on learning models do so by using topological approaches in one of two ways: the regularization of learning models or the development of means to interpret, explain, or inform the construction and/or training of neural network architectures. Of the regularization-based approaches, Moor et al. (2020) construct a topological autoencoder that preserves the topological similarity between low-dimensional representations of the input data and its latent space by comparing their respective persistence diagrams. The similarity measure used for regularization proposed by Moor et al. performs topological feature tracking of simplices within the Vietoris-Rips complex obtained from the input and latent space to regularize over the variation in distance between simplicial elements from each space. As the regularization technique is differentiable, Moor et al. can intrinsically integrate topological features of the data space into the autocoder's training process [103]. Chen et al. develop a measure of the topological complexity (in terms of connected components) of the classification boundary of a given classifier that is regularized to incentivize fewer low-persistence features to simplify the decision boundary. Training the classifier is then

topologically informed by the topological complexity of the classifiers' predictions [104]. Zhao et al. (2020) weight message passing in a graph neural network (GNN) with the scalar value output of a multilayer perceptron given persistence images obtained from shortest path filtration of graph neighborhoods, thus integrating topological representations of graph neighborhoods into embeddings learned by the GNN [105].

The second class of intrinsic topological features in machine learning employs topological interpretations in model analysis. Two main groupings of this type are observed in recent works. First are those motivated by the 'manifold hypothesis' and the notion that high-dimensional data distributions lie, and can be sampled from, some low-dimensional embedding space. Topological analysis of the manifolds data belongs to is particularly applicable to generative adversarial networks (GANs), which learn through a zero-sum game between a generator model attempting to learn the statistical distribution of data to generate samples that a classification model misclassified [106]. This context has two underlying manifolds: the data and the generator. Contemporary works have used topological tools to measure their similarity [107]. The second group of recent works that use intrinsic topological features for studying learning models topologically interpret model behavior, selection, and design. Model selection can be aided by topological analysis when considering the architecture of a neural network, such as is done by Rieck et al., who introduce a topological complexity measure of neural network graph architecture that summarizes topological features that arise when calculating a filtration of the neural network graph [108]. Recent work by Ramamurthy et al., which shows the use of a simplicial complex, a labeled Vietoris-Rips complex, can aid in explaining the classification boundary of a classifier [109]. Gabrielsson and Carlsson utilize topological data analysis to analyze topological information encoded in the weights of convolutional neural networks (CNNs) [110].

2.8 Challenges Facing Topological Machine Learning

Promise has been seen in recent research aiming to bridge topological data analysis (TDA) and machine learning (ML). As stands, however, there is no 'silver bullet' resulting from their union for scientific application or in comparison to canonical model performance. Despite the diverse and robust computational tools offered within the domains of machine learning and topological computing, recent research into the topic of their combination has primarily focused on the interplay of persistence diagrams from persistence homology and machine learning, and more specifically,

to machine learning methods that learn topological information through feature engineering for extrinsic learning of topological representations. These approaches, which provide models with expressive topological representations of data for learning, differ from those aiming to more directly integrate topology and machine learning through the intrinsic topological adaptation of learning models or their architecture to inform the learning process.

Contemporary advances where topological features extracted from data to be used by downstream learning models use vectorization or kernel mappings of topological features or design specialized layers of neural networks capable of handling such features [90]–[94], [96], [97], [102], [103]. Persistence diagrams offer limited metrics with a heavy computational overhead, such as the bottleneck distance, as they do not belong to any natural Hilbert space. Kernel methods are often used to introduce new metrics; however, the issue of complexity remains. These approaches are also limited to gaining topological information as a secondary step external to the learning process. What is more, approaches oriented around topological features created for downstream learning models often overlook many available representations TDA offers and instead focus on persistence diagrams. This narrow cross-section of TDA leaves many avenues for future research into approaches incorporating topological analysis techniques beyond persistence summaries, specifically diagrams, such as the Morse-Smale complex, Reeb graphs, or the persistence homology transform [111].

The second emerging direction of research aims to merge topology and machine learning intrinsically, namely, contemporary methods that incorporate topological analysis into aspects of the machine learning model itself. One approach to this end common among recent works has been integrating regularization terms that marshal classifiers to better align with topological constraints [103]–[105], [109], [112], [113]. Another interesting interplay of ML and TDA to directly impact model infrastructure is in regard to explainable artificial intelligence (AI) and model selection [114].

Work has shown promise to move beyond persistence diagrams when designing learnable feature representations and advantages to focus on connectivity or interpret learning models' aspects as simplicial complexes to aid model design and architecture. Ramamurthy et al. show using a labeled Vietoris-Rips complex helps explain the classification boundary of a classifier [109]. Other uses of topological data analysis have aimed to study the topological information, such as global structure and connectivity, encoded in the weights of CNNs [110]. Another such application is that by Chen et al., who simplify the decision boundaries of a classifier during training by minimizing the number

of connected components at low values of persistence [104]. Lastly, Betti curves as topological feature statistics for machine learning are often used for topological summary descriptors of data because they allow the calculation of a mean curve and provide easily computed distance metrics and kernel methods [95].

Available areas for research for topological machine learning remain expansive and fertile. One area that would help promote interest in the community, as well as make employing topological machine learning more approachable for use by practitioners, is expanding the software available, such as GUDHI [115] and giotto-tda [116]. Other remaining open challenges are reducing the computational complexity of obtaining intrinsic or extrinsic representations from topological summaries, developing learnable topological feature representations, and, importantly, the often heavy role of noncanonical choices to be made by researchers or practitioners concerning parameterization, thresholding, or relevant functions for the generation representative topological summaries from simplification through filtration, learnable feature representations (such as persistence values in the simplification sequence of the Morse-Smale complex or those used in generating persistence landscapes) for downstream learning models [94]. Importantly, current research is often limited to uniform topological summaries, such as triangulations that do not necessarily discretize the manifold domain. Beneficial research remains available in investigating how to inform learning models with representations that can be defined for unstructured cell complexes or mesh structures such as the Morse-Smale complex. Expanding research in this way would allow learning models to understand better and incorporate scale and connectivity that topology offers.

2.9 Figures

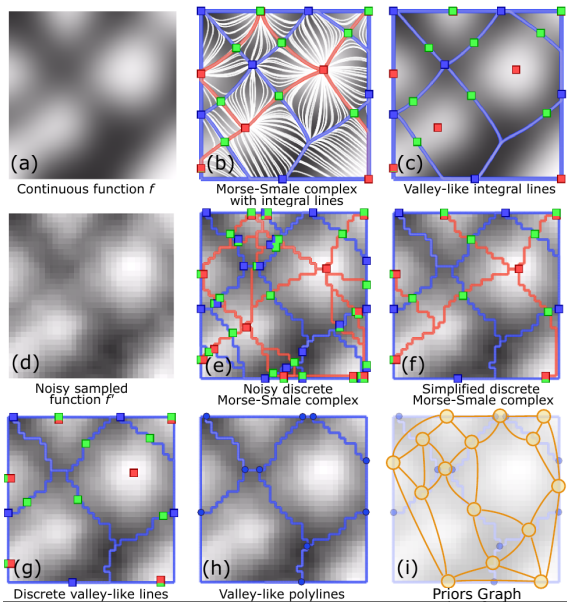
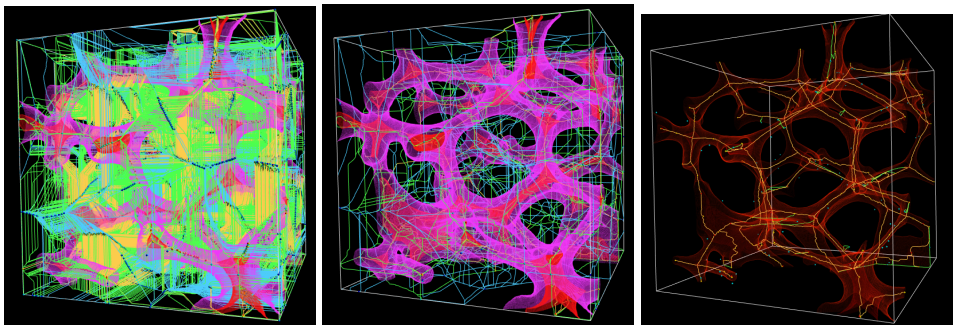
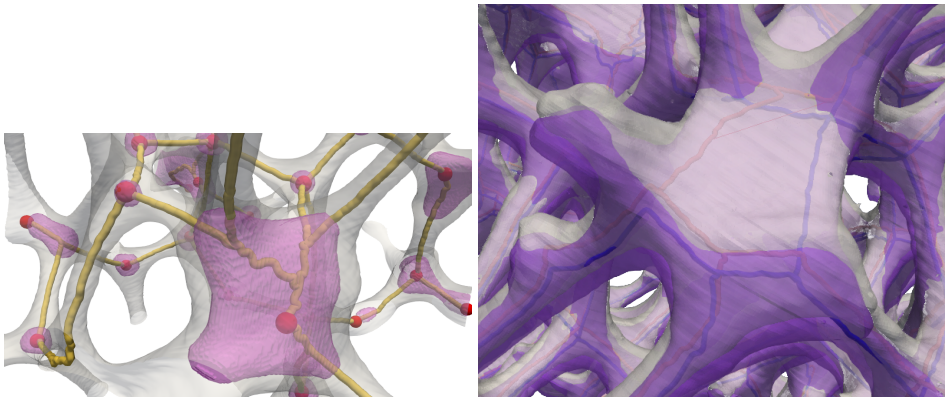


Figure 2.1: Construction of the Morse-Smale complex, the discrete Morse-Smale complex, their simplification into the valley/ridge graph, and the construction of the topological priors graph. Morse-Smale complexes are defined for functions with continuous gradients **(a-c)**. A smooth function **(a)** can be partitioned based on the behavior of integral lines **(b)**, with selected integral lines shown in white. This partition forms a cell complex, where integral lines within each cell share a common origin and destination. The 0-dimensional cells are maxima (red), saddles (green), and minima (dark blue)); the 1-dimensional cells are formed by ascending (orange) and descending lines (light blue) from saddles (green); and 2-dimensional cells are bounded by 0- and 1-cells **(b)**. Elements of this complex often form semantic features of interest in a scientific domain, such as valley-like lines **(c)**. Real-world functions often come from noisy sources and are available as samples on a grid **(d)**. Discrete Morse-theory-based methods allow practical computation of Morse-Smale complexes **(e)**, which encode both noise and discretization artifacts that may be simplified to recover the coarse-scale behavior of the function **(f)**. The valley-like structures may be extracted from this complex **(g)**, and converted to a set of priors between non-degree-2 vertices denoted the valley graph **(h)**. The priors graph (yellow), **(i)**, represents each prior as a vertex with edges between incident priors.



(a) Full MSC graph (b) Simpified 1-Skeleton (c) Inside material

Figure 2.2: Using Morse Smale Complex (MSC) representation to compute a $1-skeleton$ from the signed distance field.



(a) Maxima in junctions (b) Unstable arcs

Figure 2.3: A visualization of the 1-skeleton of the MSC computed over an open cell foam material. The 1-skeleton generated by the MSC places a maximum (red spheres) in each junction (pink blobs) (a), with arcs (yellow tubes) connecting them. This graph structure is used to derive the connectivity of junctions. The locations of the 1-skeleton within disk-like regions are not stable. In (b) the skeleton in consecutive time steps (red for gray time step, blue for purple) shifts from one side of the disk to the other, highlighting the need for stable junction extraction.

CHAPTER 3

TOPOLOGICAL MACHINE LEARNING

3.1 Introduction

In this chapter, we present our prior projects that have advanced topological machine learning. To begin, we would like to contextualize our recent works, "Classification of Topological Priors with Machine Learning" and "Learning Hierarchical Topological Structure with Graph Neural Networks" [117], [118], which interface topological data analysis and machine learning, within the landscape of contemporary works in the field as described in Section 2.7. Namely, we discuss our proposed methodologies through a taxonomic grouping of topological machine learning approaches taken in recent years.

We categorize contemporary works in topological machine learning as extrinsic- or intrinsic-topological methods. Extrinsic approaches use topological features extracted from data for downstream machine learning models. Extrinsic methods often employ vector- or function-based methods to vectorize topological features, design specialized neural network layers that can manage topological representations, or use kernel mapping of topological summaries.

Intrinsic topological methods incorporate topological features and analysis into aspects of a machine learning model's design. Two main intrinsic topological approaches in recent works are regularization-based, penalizing models to adhere to topological criteria, or developmental, employing topological analysis to interpret or inform the construction of neural network architectures. Methodologies can then be further categorized based on how intrinsic or extrinsic approaches are incorporated into the learning workflow in an observational or interventional way. Observational methods use expressive summaries resulting from topological analysis but do not use topological analysis to influence the training or architecture of a model. Interventional methods, in contrast, integrate topological aspects derived from the data or learning model itself to topologically inform the training process or architectural construction of a model.

Considering the approaches used in "Classification of Topological Priors with Machine Learning," our topological-based learning methods employ extrinsic topological features in an observa-

tional setting. In this work, we propose a learning workflow that utilizes topological analysis to construct an expressive data structure well suited for a multitude of downstream learning models tasked with the goal of image segmentation. Namely, using notions from discrete Morse theory, we can construct a topological summary of an image, the Morse-Smale complex (MSC). The discrete MSC extends notions of smooth Morse theory that can be applied in the discrete pixel space, which partitions the image space into monotonic regions that share a critical maximum and minimum. The 1-skeleton of the complex is formed by these critical points and their adjoining integral lines, which, in the discrete setting, are nodes adjoined by V-paths represented by arcs. Such a topological abstraction can be hierarchically refined for successively coarser representations. The lifespan and 'scope' of connectivity of topological features can be controlled through topological persistence, which uses a functional filtration based on the scale of difference in the value of critical points to determine a pairing of critical points that create and destroy topological features. Persistence then allows us to offer a multiscale data structure that can be tuned by the user for a given task at hand.

We now highlight how our approach allows machine learning models to learn from extrinsic topological features observationally. From a refinement of the MSC, we obtain a 1-skeleton topological representation of an image in which nodes and arcs cover the semantic object of interest we wish to segment. From this 1-skeleton, we construct a topological priors graph, the line graph of the 1-skeleton - mapping arcs of the 1-skeleton to nodes and nodes to arcs while maintaining adjacencies. Arcs of the MSC covering pixels in the image domain are then mapped to nodes of the priors graph. Connected components in the topological summary of the image expressed as topological features in the MSC can then be represented as high-dimensional vectors. Vectorized Morse cells can then contain normal image features per pixel, such as maximum and mean of pixel neighborhoods or Sobel filtration, as well as be imbued with aggregate statistics among pixels sharing a common arc, such as maximum, mean, and variance, afforded by the geometric embedding of the MSC. The vectorized representation from the geometry of topological priors then serves as topologically informed input for downstream learning models to observe, such as a Random Forest (RF) classifier or a multilayer perceptron (MLP). For the case in which graph neural networks (GNNs) are the downstream learning model, this vectorization of the geometric embedding offers another mapping of the topological summary into an informative graph structure. The priors graph we construct represents each encoded topological prior as a vertex with a high-dimensional feature vector and builds edges based on the adjacency of the topological priors in the ridge/valley graph,

allowing us to frame the learning problem as a vertex classification task in a graph leveraged by existing machine learning architectures.

In our work "Learning Hierarchical Topological Structure with Graph Neural Networks," we propose two approaches to intrinsically incorporate topological information into graph neural networks in an interventional manner. Extending notions from our previously discussed manuscript "Classification of Topological Priors with Machine Learning," we investigate the manners in which GNNs may benefit from the hierarchical role persistence plays when deriving a geometric embedding from the MSC to encompass the semantic object we wish to segment from within an image. To model multiscale topological information, we compute the MSC at multiple levels of persistence. Using sublevel sets within the ordered sequence of MSC observed during persistence filtration we construct several priors graph representations of the underlying data to use as input to a graph neural network. The priors graph then allows us to frame the task as a node classification problem using GNNs. Using various sublevel sets in the MSC sequence allows GNNs, a message-passing-based learning model, to learn from the multiscale information captured by the MSC persistence hierarchy and its subset complexes, each representing a range of local and global connectivity. Within the multiscale data structure of the persistence hierarchy priors graphs, resulting from a high persistence value, can be attributed to low granularity in the MSC and vice versa, allowing us to obtain successively coarser representations. For example, the supergraph of the priors graph hierarchy with the highest granularity is derived from the MSC, resulting from simplification with the lowest persistence value.

To incorporate the multiscale topological information captured by the priors graphs we propose two approaches, one of which modifies the training procedure of an existing GNN, denoted Hierarchical Successive Training (HST), and another that uses the resolution hierarchy of the MSC to adapt message passing of the GNN so that aggregation of node embeddings is extended across all levels of the graph hierarchy, denoted Hierarchical Joint Training (HJT). In HST, rather than training the GNN for N epochs on the highest resolution graph representation from our hierarchical geometric embeddings, we select p levels of graphs in the persistence hierarchy and train the GNN for $\frac{N}{p}$ epochs on each subgraph in the filtration sequence in increasing order by graph resolution. During this process, the GNN uses the learned node embeddings and the model weights learned during the aggregation of node neighborhoods from lower levels of the graph hierarchy to initialize the subsequent subgraph level before training. Pre-initialization from pre-trained subgraphs lower

in the filtration hierarchy allows node and node neighborhood embeddings and the learnable weights used to generate those embeddings to be shared between graphs in the hierarchy from the lowest to highest resolution graph. The final and highest resolution node embedding becomes the combined culmination of all embeddings from previous aggregations at lower persistence subgraphs. We note that this approach of HST extends beyond using the vectorization and graph data structure derived from the MSC as extrinsic topological features by interventionally adapting the GNN's training to intrinsically initialize the weights and learned embeddings of a GNN such that message passing, during the training process, spans across an increasing resolution of graph structures derived from hierarchical topological summaries computed over an image. The MSC does not necessarily need to be interpreted as a graph structure to allow learning. As a result, such a decomposition could afford a hierarchical vector representation derived from the MSC. Although we do not investigate this approach in our work, it would be interesting to see the prospective benefits of using an extrinsic hierarchical vector representation derived from the topological summary for interactively training other models beyond GNNs.

The second of our two proposed approaches, HJT, rather than interventionally training with intrinsically initialized GNN weights, maintains the extrinsic vector/graph data representation afforded from the geometric embedding of the image but intrinsically modifies the message passing process of the GNN to aggregate neighborhood information across all levels of the persistence graph hierarchy. During each iteration of training, the GNN interventionally learns node embeddings from each level of the multiscale graph hierarchy. The weight matrices used for message passing are shared between graphs in the persistence graph hierarchy, thereby cross-pollinating between the multiscale data structure to inform aggregation of embeddings, and the learned node embeddings at each persistence subgraph in the hierarchy are all combined at each iteration during training.

3.2 Topological Priors for Machine Learning

In many scientific endeavors, increasingly abstract representations of data allow for new interpretive methodologies and conceptualization of phenomena. For example, moving from raw imaged pixels to segmented and reconstructed objects allows researchers new insights and means to direct their studies toward relevant areas. Thus, the development of new and improved methods for segmentation remains an active area of research. With advances in machine learning and neural networks, scientists have been focused on employing deep neural networks such as U-Net

to obtain pixel-level segmentations, namely, defining associations between pixels and corresponding/referent objects and gathering those objects afterward. Topological analysis, such as the use of the Morse-Smale complex to encode regions of uniform gradient flow behavior, offers an alternative approach: first, create geometric priors, and then apply machine learning to classify. This approach is empirically motivated since phenomena of interest often appear as subsets of topological priors in many applications. Using topological elements not only reduces the learning space but also introduces the ability to use learnable geometries and connectivity to aid the classification of the segmentation target. We propose here an approach to creating learnable topological elements, explore the application of ML techniques using topological elements for classification tasks in several areas, and demonstrate our approach as a viable alternative to pixel-level classification, with similar accuracy, improved execution time, and requiring marginal training data.

3.3 Topological Primitives for Segmentation

In this section, we introduce a representation to employ topological primitives for learning, called *topological priors*. Empirical evidence has shown that the MSC, and hence the ridge/valley graph, covers semantic objects. The vertices (junctions) and edges (arc segments) of this representation provide an opportunity to recast segmentation from determining which pixels belong to a semantic object, to which elements of the graph do. Motivating our approach is that larger objects, compared to pixels, may have richer feature sets, enabling ML approaches to better discriminate between objects or backgrounds. Provided as a high-quality geometric embedding, we also observe that the ridge graph's sparse summary of a semantic object's structure reduces the image space to only the set of pixels that are associated with geometries of the object as represented by topological priors - the sum of which are sparse with respect to the entire image.

We begin by describing our notion of topological priors and their origin, followed by how topological priors may be encoded into a learnable data structure, the *priors graph*. We then provide an overview of our proposed workflow - starting with a description of the feature image kernels used, followed by an explanation of the interactive process presented here for the construction of a robust MSC segmentation to be converted to a priors graph augmented with rich feature statistics, geometric attributes, and connectivity information. Finally, we demonstrate how the interactive tool introduced in this work affords the priors graph to be conducive for fast manual ground truth labeling.

3.3.1 Topological Priors

To move past the pixel-level learning process, we introduce the notion of topological priors. Following computation and simplification of the MSC, we obtain a more refined geometric summary with granularity that is better suited to the semantic objects, the ridge/valley graph. Polylines realize the edges of the ridge graph, and its vertices are junctions, and both are embedded in the underlying manifold of the image domain. We call these elements *topological priors*, as they originate from a topological decomposition, and they come equipped as a geometric embedding with connectivity. As a result, topological priors provide a group of relatable encoded geometric objects within an image. Topological priors then present opportunities for new metrics, similarity measures, relational concepts, and options for feature statistics within a novel feature space. The specific encoding will likely depend on the application, and we describe an instance of the encoding in Section 3.4.

In this paper, we focus on 1-cells of the Morse-Smale Complex that correspond to polylines and their junctions as topological priors. This focus indeed best captures complex line structure, which, as we have shown, is of interest in datasets from several application domains. However, our representation could be extended to 2- and 3-cells of the MSC as well in order to model more complex shapes. We have added more detail on how this could be done in Section 3.3.3.

3.3.2 Topological Priors Feature Vectors

Once a simplification threshold has been identified, a ridge/valley graph and then a priors graph are constructed. Topological priors allow new additions to feature sets: statistics aggregated over their geometric embeddings. For each image feature outlined in Section 3.4.1.1, we compute the *median*, *minimum*, *maximum*, *standard deviation*, and *variance* among pixel intensities under the pixels covered by the geometry of the topological prior. Therefore, if the original pixels then have d-dimensional features, the priors are represented by a $5 \times d$-dimensional feature vector.

3.3.3 Priors Graph

In our segmentation tasks, we investigate use cases where the priors themselves are classified as foreground/background. Our approach is to frame the learning problem as a vertex classification task in a graph, to leverage existing machine learning architectures. The *priors graph* represents each encoded topological prior as a vertex with a high-dimensional feature vector and builds edges based on the adjacency of the topological priors in the ridge/valley graph. An illustration of the

priors graph computed from the ridge graph can be seen in Figure 2.1 (i).

Extending the priors graph construction to other topological primitives originating from the MSC mesh, such as faces or voids, can also be done using the encoding approach shown here. For example, a priors graph whose topological priors originate from 2-cell faces (such as the interior of the region highlighted in Figure 3.1 (b)), adjacent area features of an object sharing a boundary, could be encoded as nodes in the priors graph by taking the dual of the face graph, namely, encoding topological faces as nodes, and assigning edges between topological prior nodes that originate from 2-D cells sharing a boundary.

3.4 A Workflow for Learning and Comparison

We describe an interactive learning workflow that uses topological priors for fast and accurate segmentation. Following the precomputation of a stack of scale-space image features to enrich the image representation, a user precomputes the MSC and uses an interactive visualization to select the largest simplification threshold where the 1-skeleton best covers the semantic objects of interest. A ridge/valley graph is then computed, a priors graph is built, and feature vectors are constructed for each prior. Given the priors graph, a user then labels a training region/ground truth. The labeled priors graph is then given to learning models for training and inference. A user can choose to repeat this process by correcting misclassifications to use as an enlarged training set, for another round of training/predictions. The workflow ends with a fully labeled image. We illustrate this workflow in Figure 3.2.

3.4.1 Image Augmentation and Preprocessing

Humans, when segmenting semantic objects, will leverage their a priori knowledge of an object's structure and the full power of the visual system, which is hard-wired to detect scales, edges, and patterns. To give the ML methods the best chance of leveraging the same information, we augment the image with derived features of the image intensity. Furthermore, topological priors of one of these derived fields often cover the semantic objects, not the image itself.

3.4.1.1 Feature Images: Pixel-Level Feature Statistics

An image $\mathbf{X}$ is a 2-D tensor whose i, j-th entry $\mathbf{X}_{ij}$ represents the value, (e.g. grayscale intensity) of pixel (i, j) in that image. Images are first normalized prior to feature extraction, in which

additional features for each pixel (i, j) are computed. We apply a series of transformations to the image, each modeled as a function $f(\mathbf{X})$ that returns an image of the same size, and consider the values $f(\mathbf{X})_{ij}$ for various functions f as additional features for pixel (i, j). We create features of dimension $d = 47$ by using d different choices of image transformation functions f. Our goal here is not to perform exhaustive feature engineering but to provide samples from widely used techniques. Our transformations include:

- *Identity*: The original image
- *Gaussian blur*: The image convolved with a Gaussian function with standard deviation $\sigma \in \{2,4,8,16\}$ as 4 features. Gaussian blurring is a representative smoothing kernel for capturing pixel neighborhood information and minimizing fine-scale structures while not introducing unrelated structures in higher scales [119], [120].
- *Maximum, minimum, median, and variance* of pixel values in a neighborhood: The maximum, minimum, median, and variance of pixel values within a neighborhood of radius $r \in \{2,4,8,16\}$ centered around each pixel contributing 16 features within our feature set.
- *Difference of Gaussians*: The pixel-wise difference of two Gaussian blurred images as above, with respective neighborhood sizes $(r_1, r_2) \in \{(4,2),(8,4),(16,8),(32,16)\}$ adding an additional 4 features.
- *Sobel filter*: The Sobel filter [121], [122] over the image with a sliding kernel of dimensions 1×3 computing partial derivatives based on pixel intensity that contributes 7 features along with an additional Sobel filtration of the unblurred original image totaling 8.
- *Gaussian Edge Detection:* Intensities of local gradients are computed over scales and kernel sizes for Gaussian blurring ranging from 1 to 64 with a step size by powers of two, contributing an additional 7 features.
- *Hessian Eigenvalue Filter*: For each kernel neighborhood range $\sigma_r \in \{1,2,4,8,16,32,64\}$ eigenvalues of the Hessian matrix $(H_{\mathbf{X}_r})_{(i,j)} = \frac{\delta^2 \mathbf{X}_r}{\delta x_i \delta x_j}$ are computed [122] over the Gaussian blurred images $\mathbf{X}_r$ and included in the feature set contributing 7 feature attributes [123].

3.5 Interactive Computation of MSC

We describe a workflow that computes a simplified MSC, the basis for later priors graph computation. Computation of the MSC involves identifying a suitable scalar function whose topological

features cover the desired semantic objects, computing a hierarchical MSC, and identifying the appropriate threshold for simplifying this structure.

3.5.1 Expressive Topological Abstractions from User Selected Scalar Field

To ensure that topological elements entirely cover the semantic object, it is necessary first to determine an adequate *scalar field image* which, once derived, allows the computation of the topological summary to account for all pertinent imaged attributes. The aim of an initial functional filtration of the image, or *scalar function*, is to create details of the basic structure of the semantic object known to the practitioner, which may require accounting for unwanted imaging artifacts. The workflow for choosing a derivation for a scalar field image for use in topological methods has not been robustly addressed in prior work; there is no "ground truth" method to identify the scalar field that produces the priors with the desired connectivity and geometric embedding. Therefore, the steps to derive a scalar field image are chosen such that, empirically, the priors produced are sufficient for the examples in this study. For example, high-amplitude speckle noise in images usually requires some degree of smoothing. If boundaries are desired, an edge detector with a user-selected kernel size may produce the desired priors. In some cases, the priors may even be constructed over the predicted class probability field produced by an initial ML model.

As the methodologies, quality, and errors introduced when obtaining data vary widely across domains, there is no singular encompassing functional filtration to highlight attributes of interest best as is intended when constructing the scalar field image. However, there are often commonalities among data samples within individual fields where segmentation is of interest. Thus, a practitioner can likely easily recycle the approach used to construct the informative scalar field found to account for domain-specific noisy artifacts introduced during data acquisition properly. Our approach is to allow a user to select a scalar field image from the precomputed feature images. Once selected, as shown in Figure 3.2, the system computes the discrete gradient and the MSC 1-skeleton, and builds a hierarchy. The results are displayed in an interactive viewer.

3.5.2 User Selection Cycle of Topological Simplification Threshold

Using the precomputed MSC hierarchy, the first interactive cycle shown in Figure 3.2 allows the user to select an MSC with sufficient granularity to cover the semantic object by adjusting the

persistence threshold. The topological elements, at the finest scale, often encode noise as well as the objects of interest. Persistence simplification the MSC allows for coarser representations. However, in natural images, the level of simplification needed is not known in advance. We allow the user to select a simplification threshold interactively, leveraging the precomputed MSC hierarchy. The user picks a threshold that computes the sparsest topological structure that covers all semantic objects. If semantic objects are missing, the user may select a different scalar function for computing the MSC (Section 3.5.1).

3.5.2.1 Interactive Labeling Cycle

We present an interactive workflow for fast and accurate labeling, training, and inference with the use of topological priors. For fast labeling of semantic objects, we introduce a tool for the selection of topological priors. We provide background on the learning models chosen for training and inference.

3.5.3 Fast User Annotation of Topological Priors in 2D

We have developed an interactive tool to facilitate and accelerate labeling of priors graphs. The tool is a standalone application (built with C++/FLTK) for visualization and interaction with the underlying image, scalar function, topological priors, labelings, and predictions [124]. Several interactions are supported for easy selection and "painting" foreground/background labels, as highlighted for various datasets in Figure 3.1. The tool uses spatial acceleration structures to clamp the user's interaction to the nearest relevant topological element and uses shortest path algorithms to facilitate drawing long paths, branching trees, and closed loops. Flood fill and region selection tools allow rapid labeling of homogeneous regions. Using this tool, it took between 2 and 10 minutes to generate each ground truth labeling in this study.

Through the labeling tool, users can interactively set persistence values for computing decidedly appropriate priors graphs that robustly cover the semantic object, eliminating the need for extensive manual segmentation and affording an easy means to label geometries for training models tasked with learning semantic segmentations.

This approach of labeling topological priors has already been shown to accelerate the segmentation process in the neurosciences [7]. We extend the benefit of fast labeling of topological priors to train downstream models to learn and identify semantic structures quickly and accurately.

Topological priors labeled interactively in this way present the opportunity for users, if desired, to correct the inferred segmentations of their chosen learning models for re-training, allowing for more robust learning models. This tool also allows users to expand the labeled region by pulling prediction values in a region post inference and correcting them with the selection tools.

3.5.4 Topological Priors for ML Assisted Labeling

In this section, we explain the topological machine learning models we propose. We present both shallow and deep learning models, each adapted to use topological priors.

3.5.4.1 Shallow Learning

We apply a random forest to directly train over the priors graph in a supervised setting with vertex feature vectors and manually assigned class labels as described in Sections 3.4.1.1, 3.3.2, and 3.5.3. Predictions, similarly, are made over the remaining unseen priors of the priors graph as either belonging to the foreground or background class. With the priors graph fully predicted, the collection of priors represented by the foreground topological priors provides the segmented objects. If a pixel representation is desired, the pixels beneath foreground elements (priors) are painted onto a flat background-valued image, with a user-selected radius.

3.5.4.2 Deep Learning

We use a conventional feedforward neural network, or multi-layer perceptron (MLP), and an inductive graph neural network, GraphSAGE [84], for node classification of the priors graph as described in Sections 2.5.2.1 and 2.5.2.2, respectively.

We use the popular GraphSAGE [84] architecture, which implements f by concatenating each vertices feature representation with the mean-pooled features of its neighbors and passes the result through a feedforward layer. In the end, a softmax classifier is applied to the final representation of v (obtained from the last layer of the network, after the last round of propagation) to predict the class label y_v. The weight matrix $\mathbf{W}$ of this classifier along those at each layer may be learned end to end by minimizing a cross entropy loss. We also employ jumping knowledge [125]: during aggregation while learning vertex representations, concatenating the feature representations of earlier layers with the final layer's output. Our graphs have high heterophily, meaning that adjacent nodes often belong to opposite classes, and jumping knowledge was shown to be a useful design when applying GNNs

to such graphs [126].

3.6 Evaluation

In the evaluation of our approach, we pick data and associated tasks for which a user desires to segment repetitive objects. We study the resources needed and the performance of standard machine learning algorithms framed around topological priors, allowing us to understand better the effects of moving learning and classification from millions of pixels to orders of magnitude fewer high-dimensional vectors. The performance of each ML model is evaluated as a function of training set size. We further identify which learning models support the fast labeling, learning, and inferring interaction cycle needed by the example ML-guided labeling workflow we introduce as shown in (e) of Figure 3.2.

As the first method to directly predict on topological priors, we devise a new approach for evaluating its performance when compared to existing pixel-based methods. Since the goal of a practitioner is often to identify geometric objects of interest, e.g. line segments that make up a neuron, which carry both a geometric embedding and connectivity information, we propose an approach to bring pixel-level segmentation results to priors objects. Furthermore, our "ground truth" is produced with a fast labeling tool over priors objects, further justifying this decision. Our primary metric, foreground F1 score, can be viewed as the percentage of priors objects that are correctly labeled and acts as a rough proxy for how much manual correction would have to be done after ML prediction to achieve the desired segmentation.

Studies have found that high computational accuracy leads to a reduction in human interaction to achieve a satisfactory segmentation when comparing automated techniques across application domains [127], [128]. What is more, it has also been shown that reduced human interaction allows for a better user experience [129]. Furthermore, approaches with a high computational accuracy that also enable user steering and editing demonstrate higher overall accuracy and repeatability [128]. Although the inverse correlation between high computational accuracy and reduced human effort observed in recent works supports our proposed workflow, a full user study is needed in future work.

3.6.1 Comparing Priors Graph to Pixel-Level Classifications

To establish a performance baseline, we devise methods to translate priors-level ground truth to pixel-level ground truth, and pixel ML predictions to priors predictions. We describe standard

approaches to pixel segmentation.

A pixel ground truth is constructed as a binary image by first labeling all pixels as background, followed by painting all pixels under foreground priors as foreground, as shown in Figure 3.3(c). We then thicken the ground truth segmentation beyond the initial one-pixel width of priors: we perform a max radial filter, in which pixel neighborhoods assume the maximal value within their local region, to extend foreground labeling to neighboring pixels. The radius of this filtration was chosen depending on the visible thickness (in pixels) of semantic objects within the image under question. In the case of neurons, foam walls, and blood vessels, this radial increase was by two pixels, whereas it was a radius of four for the neuron cell membranes.

3.6.1.1 Shallow Learning

We train a random forest classifier [130] for our shallow learning model as described in Section 2.5.1.1. RF-Pixel approaches the task of feature detection as a trainable segmentation problem, training over a subset of the image using a manually labeled ground truth pixel segmentation. We predict over the remainder of the image during classification to generate the global segmentation. Once inference has been performed on the remainder of the image pixels, classes are then assigned back to priors, as explained at the beginning of this section.

3.6.1.2 Deep Learning

We train two deep learning models to classify pixels: U-Net and MLP-Pixel [14], [82] as described in Sections 2.5.2.3 and 2.5.2.1, respectively . The architecture of MLP-Pixel is the same as that described in Section 3.5.4, differing only in the learning rate and the number of epochs used for training as given in Table 3.1. For both models, pixel intensities are used to construct feature representations per pixel, which informs the inference decision. An image is first tiled into 64×64 rectangular subsets moving in half steps. Training is done over a subset collection of these tiles, and inference is done over all tiles. The image is then re-tiled with the inferred tile set with a padding of 10 pixels removed from all tile borders to eliminate edge artifacts in the composite prediction.

The reconstructed segmentation shares the same dimensionality as that of the input image and, once trained, serves as a pixel-level segmentation for a given input image.

3.6.2 Predicted Pixel Probabilities to Topological Priors

To compare priors- and pixel-based ML approaches, we translate pixel results back to priors objects, and compute metrics over the set of labeled priors. After pixel-level classification, each topological prior averages the predicted pixel values covered by its prior's geometry. This average is taken to be the class probability of the topological prior, as shown in Figure 3.3(e).

3.7 Experimental Setup

We outline in Figure 3.3 our experimental design for comparing pixel-based methods against the prior-based workflow introduced in this work.

3.7.1 Data

Experiments use images from different application domains, including biomedicine, neuroscience, and materials science. For each image, to optimize the ridge/valley graph and corresponding topological priors so as best to cover the semantic object, a functional operator, denoted the scalar function, is first applied to the image. Performing this preprocessing step allows for a topologically informative scalar field representation of the image (see Section 3.5.1), allowing for a more robust and expressive MSC summary of the target semantic object. Training regions for each image are grown centered in areas that illustrate the diversity of canonical representatives of the semantic objects and train over potentially confounding artifacts or morphologies. We provide a summary of attributes and statistics for each dataset in Table 3.2.

3.7.1.1 Retinal

Imaging and tracing of blood vessel arbors is used to classify disease states of the eye [131]. Obtaining a wire representation of the blood vessels is one of the first diagnostic steps. The scalar field image used to compute the Morse-Smale complex was a Laplacian of the image with kernel size 2.0 to better capture faint blood vessels, as in Figure 3.3.

3.7.1.2 Berghia

An electron microscope image is taken of a cross-section across neurons of the Berghia sea slug, that has a roughly 10,000-neuron central nervous system. Membrane prediction allows segmentation of individual neurons, which enables researchers to easily investigate relations between genes, brain, and behavior as well as study or modify entire neural circuits [132]. Identifying the

boundaries of neurons is an open challenge. The scalar field image used to compute the Morse-Smale complex was a membrane probability prediction (using a previously trained U-Net) applied to a normalized edge detection with a kernel of size 4.0. The resulting scalar field allowed for topological priors that aligned with and covered cell boundaries, filling gaps in existing predictions, as shown in Figure 3.4(a). The user's (and machine learning) task is to classify priors as either "cell boundary" or not.

3.7.1.3 Foam

A computed tomography (CT) image of closed-cell foam is used to characterize the deformation of cell walls that may result from its manufacturing process. The thin film polymer boundaries are faint compared to the background CT noise. The scalar field image used to compute the Morse-Smale complex was computed over the original image with a maximum convolution with a rotating line 10px in length and a total of 16 directions to enhance linear structures and smooth noise. The topological priors of resulting from this field include both cell boundaries and noise, as shown in Figure 3.4(b).

3.7.1.4 Diadem and Neuron

Viral expression fluorescent proteins and tissue clarification techniques allow for imaging of neurons and their projections. Understanding neurons and their constituent dendritic and axonal subtrees is a central task in many biomedical and neuroscience applications. In images from the Diadem challenge [133] and macaque brain, neurons appear as ridge-like structures in a noisy background field (Figure 3.4(c), and (d), respectively). The scalar field image used to compute the Morse-Smale complex was a Gaussian smoothed image with a kernel size of 2.0 to generate topological priors, which are classified as part of neurons or background.

3.7.2 Metrics

We compute the class F_1 score to measure the performance of all models. Representing the harmonic mean between precision and accuracy, the F_1 score can be expressed as $F_1 = 2\left(\frac{\text{precision} \cdot \text{recall}}{\text{precision} + \text{recall}}\right)$. To define precision and recall, we first formalize some notation by denoting TN for true negatives, or correct labeling of the background class; TP for true positives, correctly labeling the foreground class; FP for false positives, incorrectly labeling the background class as foreground; and FN for

false negatives, incorrectly labeling the foreground class as background. With this notation, the precision and recall are defined as precision $= \frac{\text{TP}}{\text{TP+FP}}$ and recall $= \frac{\text{TP}}{\text{TP+FN}}$. For each model and inference run, we perform a parameter sweep for the foreground/background probability threshold to maximize the class F_1 score - which directly correlates to the Dice score and is functionally equivalent to the mean IOU class score. Intuitively, our choice for using this metric is that it translates to how much work a user would have to perform to correct the segmentation.

3.7.3 Hyperparameters

We performed a parameter sweep of learning rates $\{1,2,3\} \times \{1e^{-2}, 1e^{-3}, 1e^{-4}\}$ for all relevant models. We found that each model converged after training for 10 epochs, except MLP-Pixel. For this reason, to prevent bias in timing measurements, we train all models for 10 epochs, excluding MLP-Pixel, which required 64 epochs to converge and higher weight decay to avoid overfitting. For the GNN, we use hidden vertex embedding dimensions of 512 and 1024 with output vertex embedding of 256, and aggregate neighbors' embeddings by maximum pooling. We also add jumping knowledge between aggregation layers of the GNN [125] to combat high class heterophily (see Section 3.5.4). For other parameters, we stuck to default values used in official implementations, namely, the ensemble random forest classifier from scikit-learn [134], GraphSAGE's supervised GNN for vertex classification [84] and canonical UNet architecture [14]. Our MLP used a standard architecture with three 32-dimensional layers and a sigmoid activation function. Using standard settings likely represents the procedure in the use cases we envision by domain specialists. We report the best parameters in Table 3.1.

3.7.4 Computing Environment

All experiments were run on a laptop with 3 GB GeForce GTX 970M with 1280 CUDA Cores GPU and 3.5 GHz i7-6700HQ (2.6 up to 3.5 GHz – 6MB Cache – 4 Cores – 8 Threads) processor running Ubuntu, Linux.

3.7.5 Training and Inference Procedure

For training, we chose subregions that accurately capture the diversity of geometric information and variability within each dataset, starting with size 64×64. We then grew the training boxes by approximately 10% of the image until terminating once the percentage of training exceeded more

than 60% of the image.

For MLP-Pixel and U-Net, where fixed-sized images are required, the input image was decomposed into 50% overlapping tiles of size 64×64. Tiles intersecting the training region(s) were used for training, the rest for inference. During training, tiles were augmented to reduce overfitting and increase the size of the training set. Augmentations used were random rotations up to 50 degrees, counterclockwise shearing up to 0.5 degrees, random zoom in the range of $[80\%, 120\%]$, width and height shift by 20% of the image width and height, and point filling after augmentation using reflection. A result image of the same dimensions as the input was obtained by compositing the non-overlapping centeral parts of predicted tiles. For RF-Pixel, tiling was not necessary and the respective training region was used in its entirety. Each pixel-level model, for feature statistics, uses the image transformation functions discussed in Section 3.4.1.1 and performs training and inference over the image-feature tensor with depth equal to the feature space. Pixel predictions are then sampled to the priors graph for the final segmentation.

For the priors-based models, only topological priors entirely within the training regions were considered for training. Following training, inference was performed over the remainder of the priors graph to assign probabilities to all priors as belonging to the foreground target object class. Although this inferred labeling of the priors graph is the primary segmentation result, predictions assigned to each prior can be painted back onto its constituent points in pixel space if desired.

3.8 Experimental Results

The main performance and timing results are shown in Figure 3.5. The class F_1 scores are arranged in two ways: as a function of training set size and as a function of training time. For each example, once sufficient training data (generated by manual labeling) has been provided, U-Net achieves the highest scores but at the cost of lengthy training time. A significant factor in this high score/slow speed is likely the automatic image augmentation we use to expand the available training data. Remarkably, across datasets, priors-based approaches (RF-Priors, MLP-Prior, and GNN) yield competitive scores, with orders of magnitude less training time - including the minor computational expense incurred from the required preprocessing to acquire the components used in our proposed approach, such as the priors graph, topological priors feature vectors, and the feature images.

In Table 3.3, we can see the computational overhead required to acquire these functional components in our proposed workflow contributes little. Given the minor computational expenses

required, we maintain that the advantages illustrated in Figure 3.5 serve to demonstrate the comparatively low time and effort expected of a user to achieve competitively accurate segmentation results compared to those observed in contemporary state-of-the-art approaches. From Figure 3.5 we see that RF-Priors consistently lies on (or nearly on) the Pareto-frontier of accuracy and computational time across all experiments; it achieves near-best performance at a fraction of the run time of the highest performing method (U-Net). Although RF-Pixel performed similarly in most cases, for Berghia Membrane, it had low accuracy, whereas RF-Priors maintained competitive accuracy. The demonstrated consistency of RF-Priors supports its use as a generalist tool, to be applied to data of unknown variety that may arise during scientific investigations.

In Table 3.4, we provide a snapshot of performance for each dataset where all performance curves first level off. We note that, due to the difference in methodology, the exact subset used for training between pixel- and priors-based approaches differs. The percentages shown for pixel-level methods are computed as the total labeled pixels associated with the object being segmented within the training region with respect to the total number of pixels labeled as belonging to the target object in its entirety. For priors-level approaches, percentages are computed as the total length of priors within the training region covering the object with respect to the cumulative length of priors covering the target object. As Table 3.4 shows, priors-based approaches yield competitive results with U-Nets, at a fraction of the training time. As data acquisition techniques and field-specific segmentation methods have been independently developed for some time, contemporary segmentation techniques are often highly curated to the domain-specific task [135]–[138]. As a result, performing a thorough qualitative comparison of the methodology we introduce to those methods refined for domain-specific use is out of scope in this work. Rather, what we aim to present is a fast and accurate approach suited to generalize across domains, thereby offering an alternative for practitioners who may otherwise be confined to more time- and effort-intensive approaches. A visualization of the classification of priors is provided in Figure 3.6, corresponding to the models trained in Table 3.4. For most datasets, the approaches correctly classify salient, exemplary semantic objects, and mainly differ where there is ambiguity in the images.

Although Neuron and Diadem are the same segmentation task, identifying neurons in fluorescence microscopy images, the F_1 scores of the models were dramatically different, likely because neurons empirically look the same in the Diadem data, whereas in Neuron they appear with vastly different intensities and densities. Anecdotally, the ground truth segmentation used for Neuron also

was the most subjective, with less certainty whether to label a prior as neuron or background. For many datasets, RF-Priors shows only mild improvements over RF-Pixel, indicating that the aggregation done along the pixels comprising a prior did not add discriminable information. However, for Retinal and Berghia, the priors-based approaches produced clear gains. We speculate that either the additional statistics proved important or the coverage of the priors produced a better training set, for instance, by excluding a priori pixels/subimages that may act as confounders.

3.8.1 End-to-End Segmentation

Although a full user study to evaluate the speed-up obtained through ML-assisted labeling is beyond the scope of this work, we assess the performance of such a workflow using different modalities. To be successful, labeling, training, and inference must be fast, and the model prediction accurate to minimize proofreading and correction. We estimate the fitness of each model (both pixel- and priors-based) for an interactive labeling workflow. The accuracy scores presented in Figure 3.5 can be taken as a proxy for the amount of time a user would need to correct inference results using the labeling tool.

Table 3.3 augments the model performance (Figure 3.4) with the costs to compute feature images, (re-)compute the hierarchical MSC, and build topological priors and their features. Within our workflow, the necessary computational components contribute little overhead as compared to the learning model used. Therefore, future interactive labeling tools incorporating ML-based suggestions could benefit from the high accuracy and fast training/prediction times of the RF-Priors. We highlight the clear gains RF-Priors offers for fast, accurate active learning due to its training and inference consistently falling within a reasonably timed "interactive" zone, which we illustrate in Figure 3.5. We heuristically define this as the region in which there is a time to accuracy trade-off where a practitioner can be reasonably expected to train learning models actively.

Meanwhile, when training data is smaller, priors-based approaches produce superior results. High accuracy given reduced training set sizes is an ideal case in a practical setting, since it means that a human annotator needs only to annotate a small part of the image and have a machine learning model extrapolate the segmentation of the rest of the image. As the right column of Figure 3.5 shows, when training time is an important factor, such as in an interactive session, the shallow learning approaches (based on random forest) provide the best results.

3.8.2 Limitations

Priors-based learning shows promise in improving automation in segmentation tasks, but the barrier to entry remains high. Primarily, a user currently must hold an expectation of what derived function from the input image will lead to topological priors that cover the semantic objects. The trial-and-error approach in this work requires a user first to imagine which topological abstraction to use, then deduce which derived scalar function yields those, and then visualize and evaluate the computed MSC while adjusting a persistence simplification slider. Even after extensive exploration, the selected function and its priors graph may omit or poorly represent certain semantic objects. New work is needed to assist users in selecting derived functions of the input images that generate semantically meaningful topological priors.

3.8.3 Takeaways for Continuing Research

We have developed an efficient pipeline for image segmentation combining topological data analysis and machine learning. An interactive tool allows users to quickly label a ground truth segmentation within an image for training, and subsequently, we demonstrate the use of machine learning techniques to complete the segmentation of the image.

We show that methods performing inference at the level of the topological priors, rather than the pixel-level, achieve competitive task performance and excel in low-labeling regimes requiring minimal human annotation to succeed. Moreover, they are generally computationally faster than their pixel counterparts.

By framing segmentation in the context of topological priors, active learning becomes a straightforward extension; namely, a user can select geometrically informative regions quickly, allow training and inference, and, given the predicted result, quickly identify and relabel regions of interest to better inform a model before retraining.

Using our workflow and the learning models observed to be the fastest, we plan to build an interactive ML-assisted labeling tool and perform a user study. An interactive tool will also require we develop a more formal methodology for selecting a scalar field and topological abstraction that provides the best set of priors for the segmentation task. Similarly, further investigating the feature space made possible by topological priors, such as statistics derived from the geometry and connectivity of topological primitives, may lead to significant performance improvements. Other research areas of interest to expand on the work shown here will be to target other topological

primitives in order to incorporate other geometries such as 2-D cell faces and 3-D cell voids.

Future work also arises due to an interesting question raised from our experimental results in the case of random forest, namely, what leads random forest to behave differently when presented with pixels versus topological priors, as is seen in the Berghia dataset where the pixel-based approach leads random forest to perform poorly. Moreover, we plan to extend graph machine learning models, namely GNNs, to inform their representation learning geometrically.

3.9 Figures and Tables

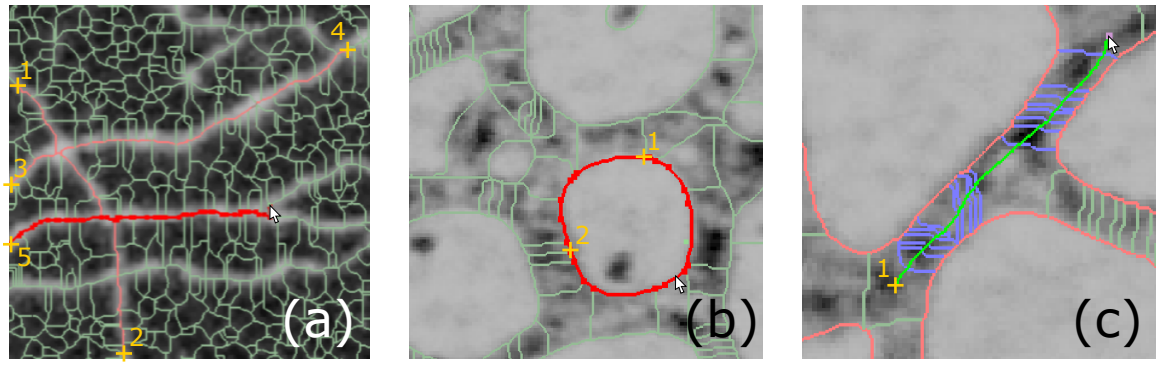


Figure 3.1: Interactive tool allowing for human segmentation by labeling topological priors, specifically arcs, as opposed to individual pixels. With the shortest-path tool, a user is six clicks into labeling the foreground neurons (a). Only three clicks are needed to draw a closed loop (b). Finding objects crossing a free-form stroke allows rapid labeling (c).

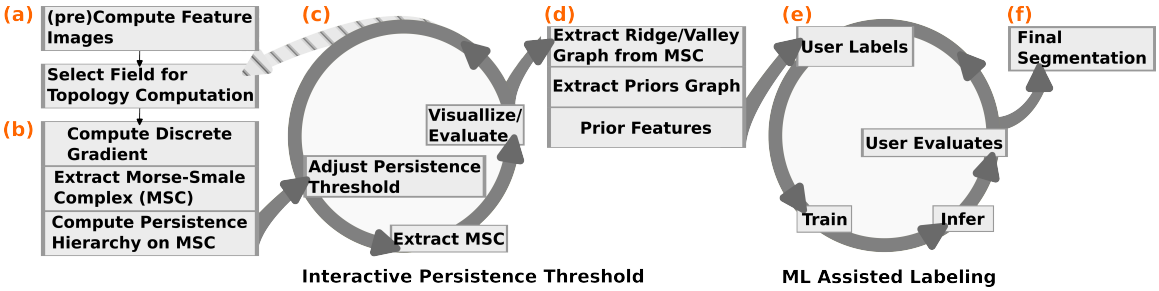


Figure 3.2: Illustration of proposed workflow using topological priors with machine learning to accelerate segmentation. Beginning at (a), feature images are precomputed. Next, the user provides a topologically informative scalar field. Using the scalar field image, in stage (b), the discrete gradient is computed in order to construct a persistence hierarchy of the MSC. Beginning in stage (c), the user interactively evaluates and selects a suitable persistence threshold affording an MSC that sufficiently covers the semantic object. During this cycle, the user can choose to provide a new scalar field image and begin the workflow again at stage (b). For stage (d), the priors graph is computed along with the aggregate statistics for the topological priors. In the next interactive cycle, stage (e), the user labels a training segmentation by selecting topological priors. The practitioner then trains the chosen learning model over the labeled segmentation, performs inference on the remainder of unseen structures, and is then able to choose to correct misclassifications made by the learning model interactively. This corrected labeling can then be used as a more robust training set for re-training a more informed classifier. Once the learning model/predicted segmentation is sufficiently accurate, the user obtains the final segmentation in (f).

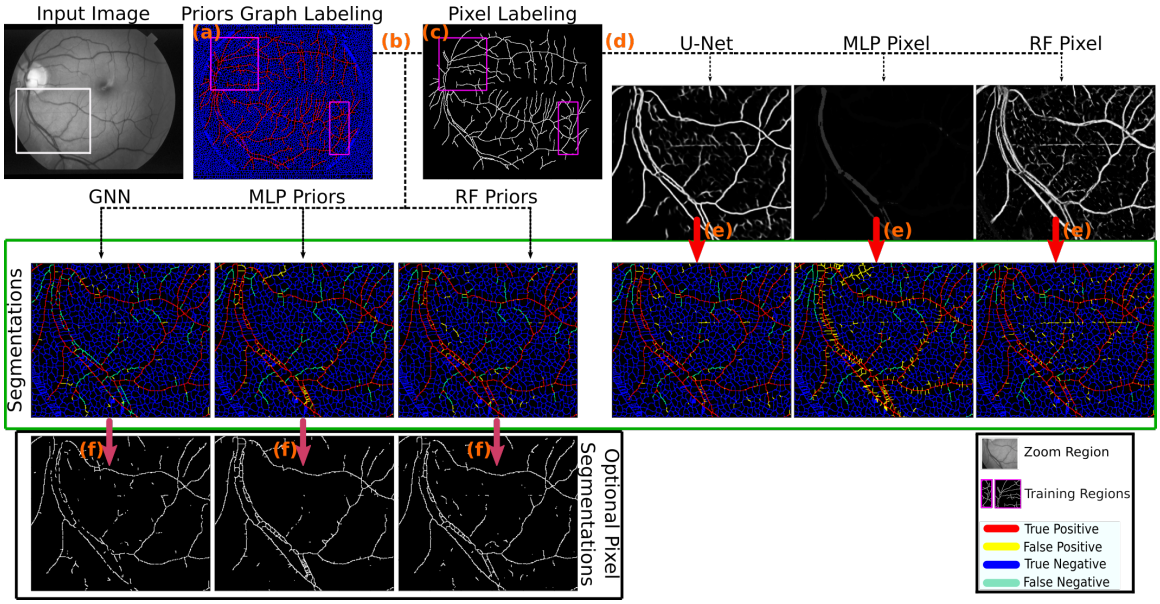


Figure 3.3: We provide a comparative example of our methodology for comparing topological priors graph predicted segmentations to pixel-level predicted segmentations for the Retinal dataset. The priors graph is computed over the image and manually labeled to create a ground truth (a). Subregions to train (pink) the ML models are identified so that the training region reasonably covers representative samples of the semantic object's diverse structures. The topological approaches train and infer directly over the priors graph (b). The pixel-level methods map the labeled priors graph to ground truth pixel labels by rasterizing the foreground priors graph labels with thick lines (c) to then train and predict over pixels (d). Pixel predictions are averaged under each prior to obtain a priors segmentation (e), enabling performance comparison. An optional pixel-level segmentation can be obtained from the priors approaches by rasterizing them (f).

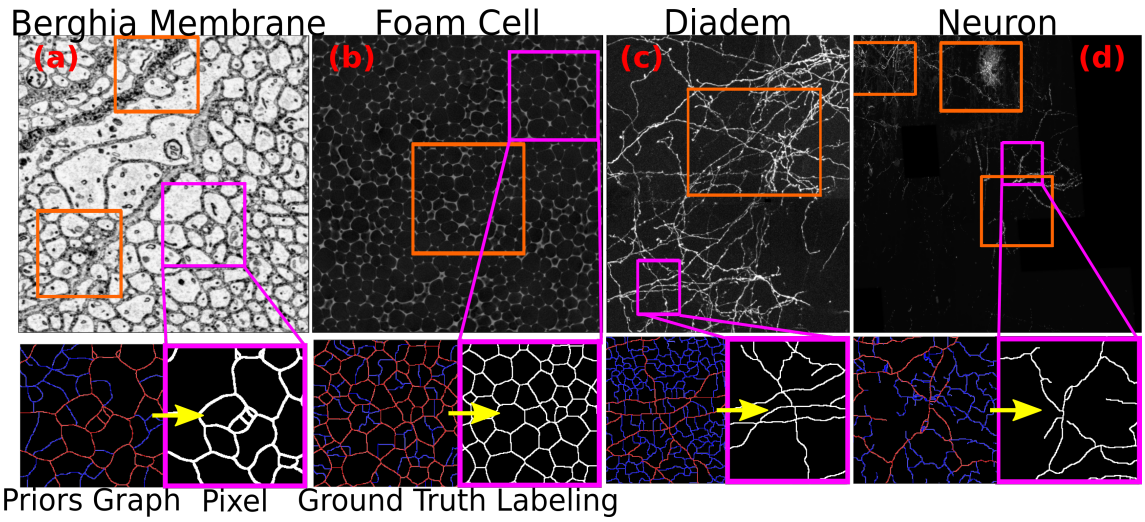


Figure 3.4: Training regions (orange) and enlarged example of priors graph and pixel-level ground truth. The provided summary for each dataset of the training regions (orange) was the training regions used to achieve the accuracy results provided in Table 3.4 as well as an enlarged view of the prior graph and pixel-level ground truths (pink). Highlighted in orange is the subset used for training all models. Each original image highlights in pink the region expanded for the visualization of results provided in Figure 3.6.

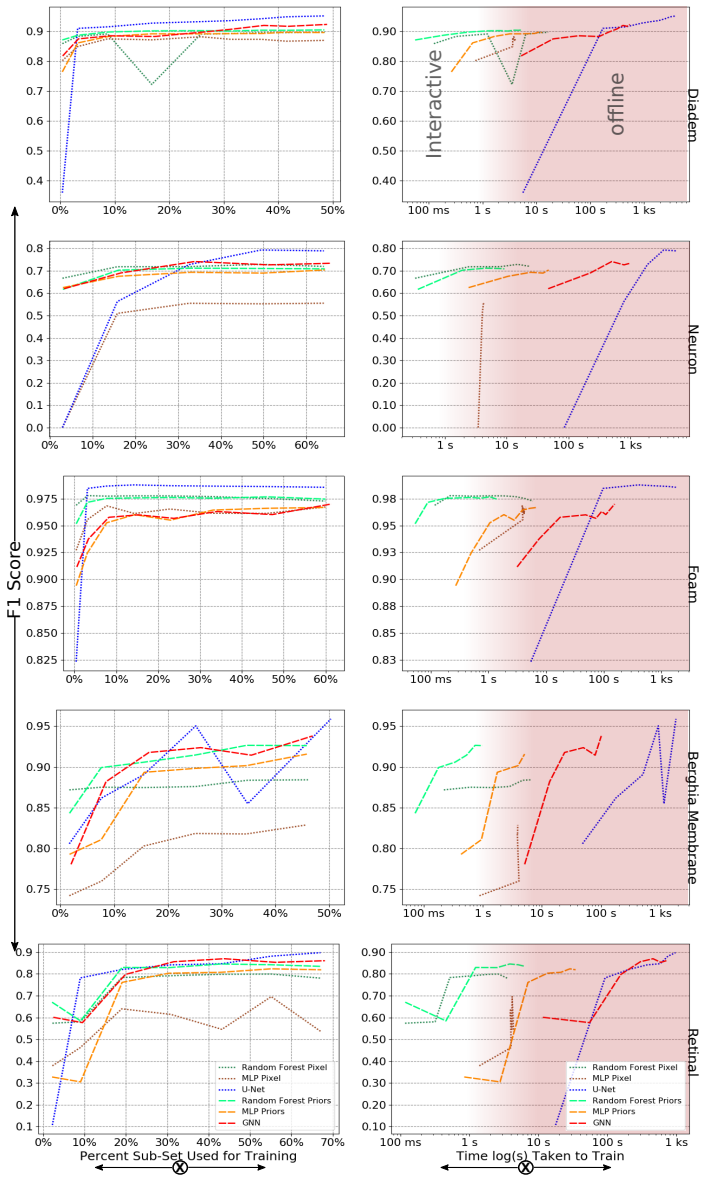


Figure 3.5: F_1 score accuracy with respect to training size and time. **(a)** F_1 scores for all methods based on training region size (left plot column) and the corresponding runtime to train (right plot column). Methods shown are RF-Pixel (dark green), MLP-Pixel (brown), U-Net (blue), RF-Priors (light green), MLP-Priors (amber), and GNN (red) over all datasets (plot rows). We provide the highlighted "interactive" and "offline" zones to delineate the time expected of a user to steer and edit a model by (re-)labeling and (re-)training. We see comparable or improved performance in priors methods performing prior-level vs. pixel-level predictions—particularly for small sizes of labeled data, the benefit being less effort required from an annotator. Meanwhile, RF-Priors is the fastest method and reaches highly competitive performance. In contrast, the pixel-level U-Net requires a much longer training time to reach higher F_1 scores.

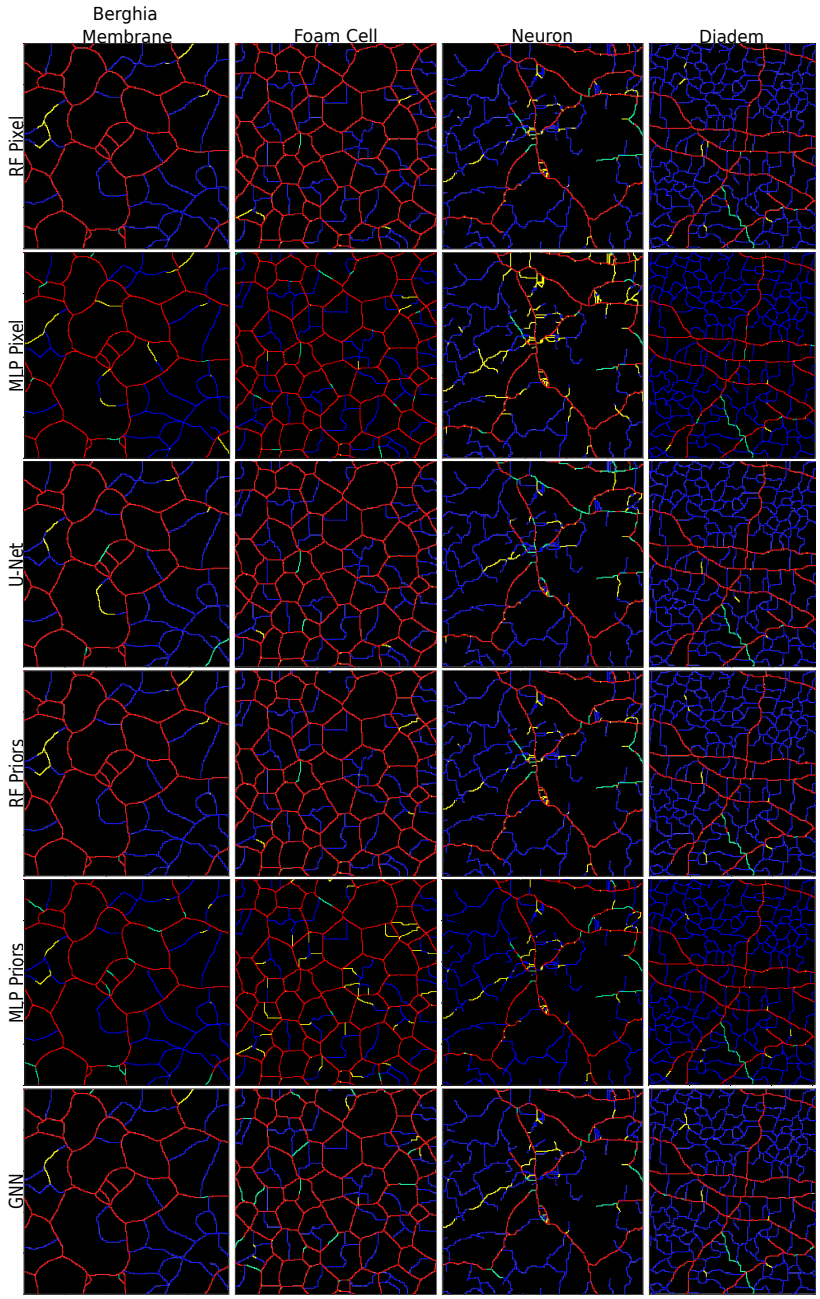


Figure 3.6: For all models and all datasets, we show the resulting predicted segmentations, which are colored according to the given model's prediction as true positive (red), false positive (yellow), true negative (blue), and false negative (cyan). Resulting inferred segmentations for all models Resulting inferred segmentations for all models **(b)** Segmentation results are shown for each model. The regions shown correspond to the pink boxes in Figure 3.4. The region(s) used for the training are those reported in Table 3.4.

Table 3.1: Hyperparameters used for each model.

Model	Learning Rate	Weight Decay	Epochs	Layers/Depth	Estimators
U-Net	$1e^{-3}$	0.0	10	23	-
GNN	$3e^{-3}$	$1e^{-7}$	10	4	-
MLP-Priors	$2e^{-3}$	0.0	10	3	-
MLP-Pixel	$1e^{-2}$	$1e^{-3}$	64	3	-
RF-Priors	-	-	-	10	50
RF-Pixel	-	-	-	10	50

Table 3.2: Summary statistics for all datasets. Numbered by column: Statistics for each (1) dataset, associated (2) pixel dimensions of the image, (3,4) total vertices/edges in the affiliated priors graph, (5) total pixels in all topological priors of the priors graph, and (6) percentage of pixels corresponding to the semantic object.

Name	Image Shape	Vertices	Edges	Total Length	% Foreground
Retinal	700×605	32,299	52,431	269,036	11.6%
Berghia	891×896	5,469	8,415	125,903	54.9%
Foam	828×846	6,268	10,058	142,895	69.9%
Neuron	$1,737 \times 1,785$	31,723	49,475	425,441	15.5%
Diadem	$1,170 \times 1,438$	28,606	45,108	475,655	21.9%

Table 3.3: Data acquisition computational times (secs.) explained by column for each (1) dataset, and its associated computation for the (2) scalar field feature image to improve the MSCs coverage of the object being segmented, the (3) multilevel MSC hierarchy based on persistence, and the (4) feature vectors associated with each topological prior.

Dataset	Feature Images	Hierarchical MSC	Prior Features
Retinal	16.39	3.65	6.11
Berghia	30.84	2.99	4.72
Foam	27.05	3.56	6.52
Neuron	100.17	9.54	7.23
Diadem	64.09	8.83	14.71

Table 3.4: Columns of F_1 scores and training times for each model for all datasets. Training region sizes were chosen at sizes where all models plateau in performance. Training regions for pixel and prior-based methods are given as the percent of pixels and the percent of priors associated with the semantic object used for training, respectively.

		RF Pixel		MLP Pixel		U-Net		% Semantic Object	RF Priors		MLP Priors		GNN	
		F1	Train Time(s)	F1	Train Time(s)	F1	Train Time(s)		F1	Train Time(s)	F1	Train Time(s)	F1	Train Time(s)
Neuron	Diadem	0.888	5.85	0.882	3.81	0.932	1337.71	26	0.902	2.07	0.892	5.19	0.900	43.77
		0.716	7.87	0.555	4.18	0.728	1887.93	33	0.712	4.60	0.693	24.40	0.741	95.62
Berghia	Foam	0.978	1.03	0.961	4.19	0.988	402.65	14	0.976	0.31	0.960	1.87	0.960	8.90
		0.873	1.43	0.803	3.89	0.891	508.86	16	0.906	0.34	0.893	1.76	0.918	7.98
	Retina	0.787	0.52	0.640	3.96	0.820	207.77	19	0.830	1.23	0.761	7.09	0.800	31.23

Highest Accuracy Model Score | Second Highest Accuracy | Fastest Model's Train Time | Percent Training Subset Used

CHAPTER 4

TOPOLOGICAL SIMPLIFICATION TO IMPROVE MACHINE LEARNING MODELS

Topological analysis reveals meaningful structure in data from a variety of domains. Tasks such as image segmentation can be effectively performed on an image's topological connectivity using graph neural networks (GNNs). We propose two methods for using GNNs to learn from the hierarchical information captured by complexes at multiple levels of topological persistence: one modifies the training procedure of an existing GNN, and one extends the message passing across all levels of the complex. Experiments on real-world data from three domains show the performance benefits to GNNs from using a hierarchical topological structure.

4.1 Related Work

Related works that consider topologically informed message passing for GNNs [88], [139] are typically presented with a single graph structure a priori as input (either one graph in which to make node-level predictions, or one graph per data point for graph-level prediction tasks). In contrast, we are constructing an ordered set of graphs from a hierarchy of successive topological simplifications of our (image) data. Each topological summary is determined by topological persistence, and the sequence of these summaries is ordered by persistence level, which allows us to obtain a hierarchy of graphs representing different scales of topological information. Our new challenge, then, is to adapt the message passing of graph neural networks to happen across several graphs, whether jointly or successively (we consider and compare both strategies).

Our work has parallels to the exciting research area of multilevel frameworks for graph neural networks [140], [141], which derive a multi-scale hierarchy of graphs from a single input graph using spectral coarsening methods (differing from our topological constructions). Note that their primary goal is to speed up the computation of GNNs by training them on the smallest coarsened graph and projecting the results back to the larger original graph, whereas we perform training at all

scales of information.

4.2 Topological Graph Hierarchy

To model multiscale topological information, we compute the MSC at P levels of persistence. Through our previously proposed approach for constructing topological priors graphs as described in Section 3.2, we are able to obtain a sequence of P increasing in resolution. In contrast to hierarchical graph-level representation learning methods that learn to pool each input graph [142], this gives us several graph representations of the input to train a graph neural network. Computationally, a smaller persistence value produces a finer granularity graph, higher in the MSC hierarchy. Thus, we obtain a hierarchy of graphs $G_1, \ldots, G_P$ where $\forall p_i$ for $i \in [1, \ldots, P], G_{p_{i-1}} \subseteq G_{p_i}$: that is, $G_{p_{i-1}}$ is an induced subgraph defined on a subset of the vertices in G_{p_i}. Node neighborhoods thus share this property: for each node $v_i \in \mathbf{V}_{p_i} \cap \mathbf{V}_{p_j}$, e.g. $\mathcal{N}_{p_i}(v_i)$ for $v_i \in \mathbf{V}_{p_i}$ and $\mathcal{N}_{p_j}(v_i)$ for $v_i \in \mathbf{V}_{p_j}$, we have that $\mathcal{N}_{p_i}(v_i) \subseteq \mathcal{N}_{p_j}(v_i)$. We give all graphs the full node set found at the lowest persistence level of the hierarchy, but nodes that would not otherwise exist in the graph at a higher persistence level are disconnected.

4.2.1 Hierarchical Topological Priors Graph and Message Passing

We introduce the hierarchical priors graph (HPG), which combines a sequence of prior graphs obtained with successive simplifications of a Morse-Smale complex PGn (fine resolution priors graph), PGn-1, ... PG1 (coarsest resolution priors graph), with arcs connecting nodes corresponding to same cells at different resolution levels. Additionally, we introduce the reduced hierarchical prior graph (RHPG), obtained by reducing the HPG to using only arcs connecting cells from level i to level i+1. Using the RHPG, we propose a novel initialization-based training paradigm for graph neural networks (GNNs) called "hierarchical successive training" (HST), where GNNs learn from multiple scales of connectivity by sequential training on the prior graphs for N/P epochs each during all N epochs.

We also introduce a novel message passing scheme for GNNs based on the HPG which performs message passing along arcs within each prior graph PGi (same level), followed by passing messages along arcs between different prior graphs (different levels). This novel message passing scheme is illustrated in Figure 4.1. This "Hierarchical Joint Training" (HJT) aggregates neighborhood information within and between neighborhoods of each prior graph and across multiple priors

graphs of different connectivity scales. Unlike other topologically-informed GNN message passing approaches, our method constructs an ordered set of graphs from successive topological simplifications of image data, allowing learning from multiple scales of topological information. We are also able to exploit the novelty of the priors graph in that message passing can be performed between any neighboring topological cells as colored green in Figure 4.1. The final step, highlighted in orange in Figure 4.1, illustrates a novel approach to aggregate and combine learned embeddings from each graph in the graph hierarchy, outlined in purple and represented as a filled green dot in Figure 4.1, from each of those graph's, shown on the left of Figure 4.1, respective multiple neighborhoods of topological cells, with priors in the neighborhood colored green and outlined in blue in Figure 4.1, multiple graphs within the graph hierarchy.

4.3 Framework for Learning

4.3.1 Feature Construction and Labeling

We construct features by applying standard transformations to an input image used in our previous study described in Section 3.2. These include the identity of the image; Gaussian blurring, differences of Gaussians with standard deviation $\sigma \in \{2,4,8,16\}$ to serve as a smoothing kernel for capturing pixel neighborhood information; the maximum, minimum, median, and variance of pixel neighborhoods for radius sizes $\{2,4,8,16\}$; Sobel filtration; Gaussian edge detection; and Hessian eigenvalue filtration. Nodes originating from topological priors also afford aggregate pixel statistics for those covered by priors. We compute the median, minimum, maximum, standard deviation, and variance among pixel intensities for these. Altogether, the features computed give us 235-dimensional features per topological prior. If the original pixels have d-dimensional features, the priors are represented by a $5d-$dimensional feature vector resulting in 235 features per topological prior.

We use an interactive tool first proposed in Section 3.2 to facilitate fast selection and human labeling of priors graphs. Through the tool, a human expert can directly label topological priors as foreground or background. The tool also allows a user to construct and visualize the MSC persistence hierarchy, interactively select a persistence threshold affording an MSC found to cover the semantic object sufficiently, and subsequently label topological priors from the priors graph derived from the user's chosen MSC with several easy interactions to "paint" label assignments to priors.

4.3.2 Identifying and Labeling Topological Graph Hierarchies

A higher persistence subgraph can encompass multiple topological primitives from a lower persistence supergraph. In a labeled priors supergraph, a topological primitive in the subgraph receives the majority class label from corresponding parts of the supergraph if the proportion of primitives in the supergraph surpasses a user-defined union threshold. In this way, each prior in the subgraph is given the predominant label of it's corresponding priors in the supergraph. This approach identifies significant priors from the supergraph as components of the subgraph. Formally, for persistence levels p_i and $p_j \geq p_i$, the class ℓ_j of a topological prior v_{p_j} in the priors subgraph of persistence p_j is a subset to priors supergraph of persistence p_i with labeling scheme ℓ_i. Given a union threshold u the label ℓ_j is given as follows: $\ell_j = \ell_i$ if $\frac{1}{|v_{p_i} : v_{p_i} \in v_{p_j}|} \sum_{v_{p_i} \subseteq v_{p_j}} 1(\ell(v_{p_i}) = \ell_i) \geq u$.

4.3.3 Learning Models Compared

We train a random forest (RF) and multilayer perceptron (MLP) on pixel-level feature statistics, denoted **RF Pixel** and **MLP Pixel**. We also investigate the performance of pixel-level classification using a U-Net [14]. We construct features for each topological prior using the aggregate statistics (median, minimum, maximum, standard deviation, and variance) for pixel intensities of the pixels covered by the geometry of the topological prior. We perform classification in the topological prior space without using the graph structure using a random forest and an MLP, denoted **RF Priors** and **MLP Priors**, and with the priors graph's graph structure and topological priors' features using GraphSAGE [84] as a baseline **GNN** model to learn node representations. We denote our hierarchical GNN methods as **GNN-HJT** and **GNN-HST** for hierarchical joint training and hierarchical successive training, respectively.

4.4 Methodology

We consider two methods for learning node representations for classification using GNNs while exploiting the full hierarchy of topological information. The first method modifies the training of GNN over graph hierarchies, and the second modifies the architecture to pass messages jointly between all hierarchical graph levels.

4.4.1 Hierarchical Successive Training (HST)

For our first method, we use a GNN, which learns node features via message passing in the spatial domain [84]. For the i^{th} persistence level p_i (moving from sparsest to densest), the aggregated node embedding for node v_i from neighbors $u_i \in \mathscr{N}_{p_i}(v_i)$ of persistence subgraph G_i is given by

$$h^{k_i}_{\mathscr{N}_{p_i}(v_i)} \;=\; \sigma\big(AGGR(h^{k_i-1}_{u_i} \;:\; \forall\, u_i \in\; \mathscr{N}_{p_i}(v_i))\big)$$

Node v_i's updated embedding is concatenated with the target node embedding from the previous iteration of aggregation $h^{k_i}_{v_i}$ and the aggregated neighbor features, parameterized with target embedding and neighbor embedding weight matrices $W^{k_i}_s$ and $W^{k_i}_n$, respectively. Similarly, the target node embeddings and neighboring node embeddings are passed through a two-layer multilayer perceptron (MLP), which after sum-pooling, are multiplied with weight matrices W_s and W_n to complete the AGGR step.

Instead of training a GNN for N epochs on the finest graph G_P, we train it for $\frac{N}{P}$ epochs each on graphs $G_1,\ldots,G_P$, using the embeddings from G_{i-1} to initialize the embeddings of corresponding nodes in $G_i\ \forall\ i \in [1\ldots,P]$. The MLPs and node embedding weight matrices for target and neighboring node embeddings are shared between graph hierarchies. Training begins with the smallest subset graph and iteratively increases in graph size. We also successively share the learned embedding weight matrices and matrices of the MLPs as well. As a result, when we begin training on graph G_p, the embedding for node $v_p \in \mathscr{V}(G_p)$ is the combined (in our implementation with concatenation) embedding from previous aggregations at lower persistence subgraphs:

$$h^{k_p}_{v_p} = COMBINE\big(h^{k_{p-1}}_{v_{p-1}}, h^{k_p-1}_{v_p}\big)$$

4.4.2 Hierarchical Joint Training (HJT)

For node v_i in node set $\mathbf{V}_{p_i}$ of subgraph $\mathbf{G}_{p_i}$, the feature representation $h^k_{v_i}$ at GNN layer $k \in \{1,\cdots,K\}$ first combines self-representations from the previous level of aggregation with aggregated neighbor information:

$$h^k_{v_i} = COMBINE\big(W^{k-1}_s h^{k-1}_v, AGGR\big(\{W^{k-1}_n h^{k-1}_u : u \in \mathscr{N}_{p_i}(v_i)\}\big)\big)$$

The resulting embedding is combined with the embedding representation from other persistence subgraphs. As an example for two persistence levels p_i and p_j where $v_i \in \mathbf{V}_{p_i} \subset \mathbf{G}_{p_i}$ and $v_j \in \mathbf{V}_{p_j} \subset \mathbf{G}_{p_j}$, we have

$$h^k_v = \sigma\big(COMBINE\big(h^k_{v_i}, h^k_{v_j}\big)\big).$$

4.5 Experimental Setup

4.5.1 Datasets

We evaluate our methods on three image datasets used in prior studies (see Section 3.7.1), which were selected to ensure sufficient variety to explore the general applicability of the methods we introduce in this work [143]. For each image, a functional operator is first applied. This preprocessing step allows for a topologically informative scalar field representation of the image, or *scalar field image*, which optimizes the expressivity of the MSC summary and allows the corresponding topological priors to cover the target semantic object best.

The datasets chosen were *Retinal* segmentation of blood vessel arbors (biomedicine) [131], cell boundary segmentation from computed tomography imaging of closed-cell *Foam* (materials science) [144], and segmentation of projected sparse *Neuron* imaging (neuroscience) [145]. For preprocessing and the construction of scalar field images, we follow as is described in Section 3.2 [143], which also includes justification in the scalar field image design choice.

As a basic demonstration of using a topological hierarchy, we use two levels of persistence (resulting in one subgraph and one supergraph). We chose persistence levels per dataset such that the subgraph was sufficiently smaller in size and sufficiently covered the semantic object. We give more detailed summary statistics and descriptions of the datasets used for evaluation in Table 4.1.

4.5.2 Homophily and Class Balance

We report the homophily and class balance ratios for the derived priors graphs in Table 4.2. The class balance ratio is defined as the total positively labeled foreground over the total negatively labeled background nodes. For nodes u and v with labels y_u and y_v and edge set $\mathscr{E}_i$ for graph G_i in the persistence hierarchy, the homophily ratio [126] is given by $h_i = \frac{1}{|\mathscr{E}_i|} \sum_{v \in \mathscr{E}_i} \frac{|\{u \in \mathscr{N}(v) : y_u = y_v\}|}{|\mathscr{N}(v)|}$.

Intuitively, we expect the class imbalance to be larger in subgraphs due to being associated with a sparser persistence subgraph since the ridge/valley graphs these subgraphs originate from span larger regions of connectivity in the image space, which are more likely, but with less accuracy, to be associated to the ridge lines of the semantic object. Similarly, we anticipate the homophily ratio to be lower for lower level persistence graphs since, being sparser and of low granularity, topological priors cover and are adjacent to more expansive but less accurate topological 1-cells.

4.5.3 Metrics

We use the class F_1 scores of all pixel-and priors-based models for comparison. each learning from pixel and topological level features, as outlined in Section 4.3.1. Training regions for each image are grown and first centered in areas that illustrate the diversity of canonical representatives of the semantic objects and train over potential confounding artifacts or morphologies. We perform a parameter sweep for each run over each model's inference foreground/background probability threshold to maximize the class F_1 score.

4.5.4 Training and Inference Procedure

For training, we chose subregions that accurately capture the diversity of geometric information and variability within each dataset, starting with size 64 $\times$ 64. We then grew the training boxes by approximately 10% of the image until terminating once the percentage of training exceeded more than 60% of the image. Figure 4.2 illustrates an instance of the training subset regions from all datasets, as was used by all models during our study. Topological priors belonging to the priors graph entirely within the regions of intersecting tiles of size 64 $\times$ 64 were used for training, and the remainder for inference probabilities were assigned to all priors as belonging to the foreground target object class.

4.5.5 Hyperparameters and Computing Environment

We performed a parameter sweep of learning rates $\{1,2,3\} \times \{1e^{-2}, 1e^{-3}, 1e^{-4}\}$, finding $3e^{-3}$ to be the best. We trained each model for 10 epochs following the original GraphSAGE paper [84] and used a weight decay of $1e^{-7}$. For each variant of GNN, we used four layers and hidden vertex embedding dimensions of 512 and 1024 with output vertex embedding of 256, and aggregate neighbors' embeddings by maximum pooling. We also added jumping knowledge between GNN layers [125], which has been found helpful when class heterophily is high, as in our datasets [126]. All experiments were run on a laptop with 3GB GeForce GTX 970M with 1280 CUDA Cores GPU and 3.5GHz i7-6700HQ processor running Ubuntu, Linux.

4.6 Experimental Results

Figure 4.3 shows that our hierarchical approaches generally improve over the conventional GNN and non-GNN baselines, particularly for the Neuron and Retinal datasets, where there is a larger

difference between the topological summary of the computed subgraphs and supergraphs. This difference can be seen by comparing the respective number of nodes and edges shown in Table 3.2.

In Table 4.3, we provide empirical measurements of the time required to compute the topological structures needed for our methods. Compared to traditional image segmentation approaches operating in pixel space, our methods result in appreciable speed-up even when accounting for the overhead of the topological data analysis. In Figure 4.4, we demonstrate the comparatively low time and effort expected of a user to achieve a given segmentation accuracy (also shown in Figure 4.3), with our methods showing competitive results to that of contemporary pixel-based approaches, and especially in comparison to U-Net, which takes considerably longer to train. Moreover, of the GNN approaches, the hierarchical approaches are the fastest, notably GNN-HST, which performs a portion of the training on smaller graphs in the hierarchy as opposed to traditional methods, which train exclusively on the larger supergraph.

Our GNN-HST variation also enjoys runtime gains compared to the conventional GNN using the full supergraph for all of training, due to the successive training scheme allowing us to perform a portion of training on the lower complexity smaller subgraph. Our joint training yields slightly higher accuracy as the model simultaneously uses multiple scales of topological information, and the runtime remains manageable. Although non-network baselines, using famously fast ML models such as random forest and few-layer MLPs, are faster, we have seen that they generally are less accurate.

4.7 Limitations

Hierarchical GNN training was less effective on the Foam dataset, which had fewer differences between subgraph and supergraph. On this dataset, methods performing learning in the pixel space achieved higher performance than learning with topological priors, which may be due to the sparsity of the priors graphs and high class imbalance, as shown in Section 4.5.2. Future work should explore how well topological priors model semantic structure, how to construct informative topological information, including scalar field images, and the impact of homophily and class balance.

4.8 Takeaways for Continuing Research

We propose graph neural network methods that learn from hierarchies of graphs representing the multiscale topological structure of image data. Our results demonstrate promise for increased

accuracy and improvement in training time compared to conventional GNNs, and we provide exploratory insights into the effect of class imbalance and heterophily in graph learning.

4.9 Social Impact

Our methods have the potential to minimize the amount of human effort required to extract semantic structure from scientific images, which may contribute to socially beneficial breakthroughs in medicine, materials science, and more. As training data comes from human annotations, our methods are subject to reproducing annotators' biases and expertise knowledge. A socially impactful future direction is to understand the extent to which topological priors reflect semantic structure, which we have found to vary across datasets, and which may also vary for subpopulations within a dataset.

4.10 Figures and Tables

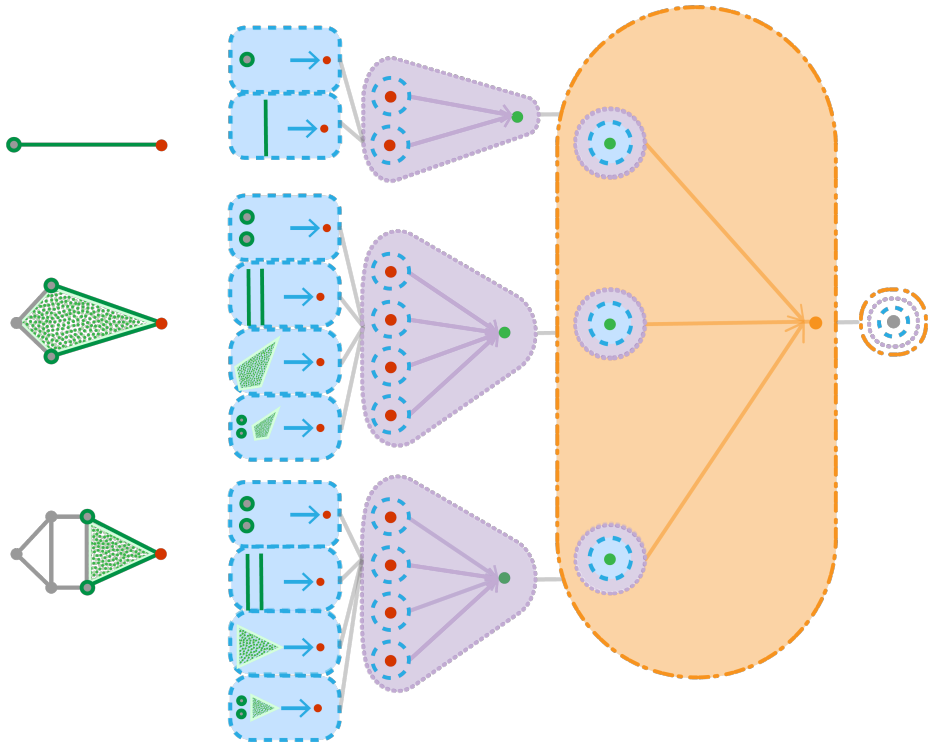
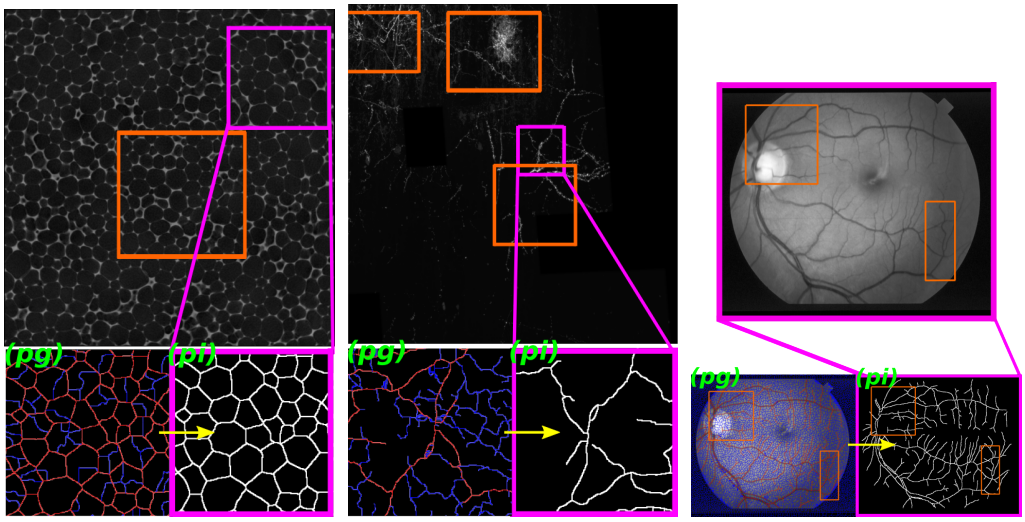
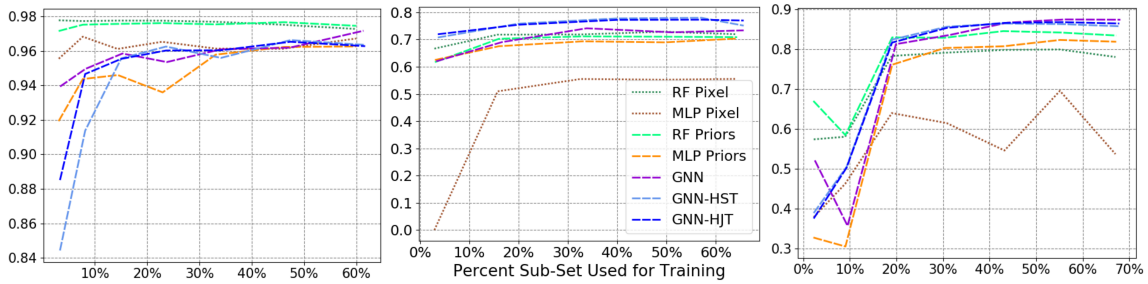


Figure 4.1: Illustration of novel message passing scheme for hierarchical joint aggregation. The process starts with a sequence of topological simplifications of a simplicial complex, with the lowest resolution at the top, consisting of two 0-cells and one 1-cell. The highest resolution and full simplicial complex is shown in the bottom left, with six 0-cells, eight 1-cells, and three 2-cells. Highlighted in green are the neighborhoods to the target simplex (red 0-cell) we are currently updating with hierarchical joint message passing. First, messages are passed from each neighborhood of the target 0-cell in the simplicial complex. The topological priors graph offers a novel notion of neighborhoods - rather than being restricted to the geometric constraints of the simplicial complex, messages can be passed between cells of arbitrary rank and dimension. Once messages from neighbors have been passed within the neighborhood, aggregation within each neighborhood is performed, highlighted in blue. Following this and for each subgraph, between neighborhood aggregation (purple) is performed, aggregating all neighborhoods' information. Introduced in this work and highlighted in orange is across neighborhood aggregation, aggregating all combined neighborhood embeddings from the previous step across all subgraphs. The final representation of the target simplex is then updated as the last step.



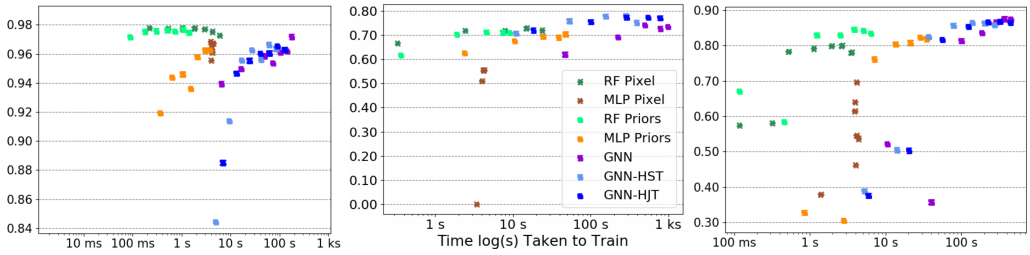
(a) Foam (b) Neuron (c) Retinal

Figure 4.2: A summary of each dataset with the subset region(s) (highlighted in orange) used to train all models for the experimental run at which the accuracy results of all models begin to plateau, as seen in Figure 4.3. Highlighted in pink is the enlarged region demonstrating the priors graph (pg) and pixel-level (pi) ground truths.



(a) Foam (b) Neuron (c) Retinal

Figure 4.3: Class F_1 versus percent priors graph used in training. We see improvements from hierarchical training on the Neuron and Retinal datasets with lower performance on Foam.



(a) Foam (b) Neuron (c) Retinal

Figure 4.4: Class F_1 versus time taken to train priors graph used in training. Times and accuracies correspond to training region size and accuracy as provided in Figure 4.3. Among the GNN approaches, the fastest are the hierarchical methods and notably GNN-HST.

Table 4.1: Dataset and priors graph statistics.

Name	Retinal	Foam	Neuron
Image Shape	700×605	828×846	$1,737 \times 1,785$
|V| Sub/Sup	676 / 24,851	1,703 / 8,069	323 / 23,038
|E| Sub/Sup	1,097 / 39,765	2,429 / 13,234	480 / 34,527
p_i **Sub/Sup**	0.8 / 0.01	80 / 20	45 / 11
Total Length	269,036	142,895	425,441
%Fgrnd.	11.6%	69.9%	15.5%

Table 4.2: Homophily and class balance ratio for sub- and supergraphs.

Data	Homophily Ratio		Class Ratio	
	Sub-	Supergraph	Sub-	Supergraph
Retinal	0.39	0.91	0.55	0.16
Foam	0.38	0.76	5.12	1.93
Neuron	0.35	0.86	1.32	0.28

Table 4.3: Computational times (secs.), explained by column, for each (1) dataset, associated scalar field image to improve MSC coverage (2), the multilevel MSC hierarchy (3), and feature vectors for all topological priors (4).

Dataset	Scalar Field Image	Hierarchical MSC	Prior Features
Retinal	16.39	3.65	6.11
Foam	27.05	3.56	6.52
Neuron	100.17	9.54	7.23

CHAPTER 5

TOPOLOGICAL FILTRATION LEARNING FOR GRAPH NEURAL NETWORKS

We now consider some of the strengths and weaknesses common and unique to the three topological machine learning methods introduced in our prior works: an observational approach that uses extrinsic topological features presented in "Classification of Topological Priors with Machine Learning", an interventional approach to train graph neural networks (GNNs) to intrinsically initialize weights and embeddings of the GNN with a hierarchical topological representation, and an intrinsic adaptation of message passing to interventionally train a GNN across hierarchies of topological features.

In our first work, we present an approach to vectorize the geometric embedding afforded by the Morse-Smale complex (MSC). The simplicity of this approach affords a fairly straightforward means of deriving an expressive vectorization from the topologically computed geometric embedding. An additional strength of this approach is that it allows a practitioner to employ a number of topological descriptors in machine learning tasks through aggregate statistics computed from the "superpixel" geometric embedding of arcs delineating regions of topological connectivity from the MSC. The benefit of the feature space made possible by topological priors is open-ended and, therefore, poses a weakness to our methodology. Namely, further study must be done over the feature space made available from topological priors, such as meaningful aggregate statistics and summary statistics derived from the geometry and connectivity of the topological summary itself. A second strength of our proposed work is that it is model agnostic, allowing any downstream learning model to use vectorized topological features derived from the topological summary of an image for learning.

A major caveat of our work, which requires future attention, is the number of noncanonical functional decisions required of a user to obtain an expressive geometric embedding of topological priors. Firstly, a user must provide a scalar function to construct a suitable scalar field image to aid the MSC computation. As there is no precept to constructing the most expressive MSC, it

is left to the user to determine which scalar function will accentuate or produce values that align with attributes of interest in the semantic object. This scalar field image is then used to aid the construction of the MSC. Similarly, no 'reasoned' persistence value will afford the optimal coverage of the MSC. As a result, it is up to the practitioners' empirical judgment to determine the optimal granularity of the MSC through manual tuning of the persistence value used in filtration. What is more, the user is required to employ the same empirical judgment during the derivation of a suitable scalar field image when obtaining an augmentation of the image that is expressive of the semantic object. However, the ability to determine which scalar field image to employ and fine-tune the topological simplification during filtration grants the user more control over obtaining a robust MSC and allows for an interactive segmentation workflow during the MSC's construction and more informative topological primitives for interactive selection. For this reason, in our work, we present an interactive learning framework where the user is an active participant during the construction and training process of the model.

Our subsequent work suffers the same weaknesses and enjoys the same strengths since it is also dependent on priors graphs obtained from the nested sequence of subsets in the persistence filtration of the MSC. However, due to the multiscale data structure employed from the persistence hierarchy, our approach offers both a hierarchical vectorization of extrinsic topological features and a learnable hierarchical graph representation. As with other vectorization-based approaches, one potential concern is scalability. However, since the computation of the discrete MSC is well studied and can be done efficiently, we find that, compared to traditional image segmentation methods operating in pixel space, our methods result in appreciable speedup even when accounting for the overhead of the topological data analysis. Related works that consider topologically informed message passing for GNNs [103], [105], [146] are typically presented with a single graph structure a priori as input (either one graph in which to make node-level predictions or one graph per data point for graph-level prediction tasks). In contrast to these approaches, we are constructing an ordered set of graphs from a hierarchy of successive topological simplifications of our (image) data. Each topological summary is determined by topological persistence, and the sequence of these summaries is ordered by persistence level. The nested sequence of subset MSC increasing in granularity allows us to obtain a hierarchy of graphs representing different scales of topological information. Our work then has parallels to the exciting research area of multilevel frameworks for graph neural networks [147], [148], which derive a multiscale hierarchy of graphs from a single input graph

using spectral coarsening methods (differing from our topological constructions). Note, however, that their primary goal is to speed up the computation of GNNs by applying them on the smallest coarsened graph and projecting the results back to the larger original graph, whereas we perform training at all scales of information. Unlike previous works, our challenge is to adapt GNNs to learn from several scales of topological information using a sequence of subset graphs, requiring message passing to happen across several graphs, whether jointly or successively (we consider and compare both strategies).

From our investigation into adaptations of GNNs for topologically based multi-level graph learning, though, we demonstrates notable speed-up as a coincidental benefit of our proposed methodologies as compared to their canonical GNN counterparts (and notably for our Hierarchical Successive Training approach) due to a reduction in complexity and size of the coarser lower level graphs in the persistence hierarchy.

Our approaches require further investigation of what scales of graph hierarchies are informative and in what applications or learning frameworks persistence-based graph filtration hierarchies are best applied. To this end, an active direction of our research aims to study how sensitive hierarchical learning is to class imbalance and node (or edge) heterophily (the average proportion of node neighbors with a different class label) within the graph hierarchies, what scales topological priors best model geometric structures, and how best to build more expressive GNNs by leveraging or better understanding hierarchical representations and higher order structures. Moreover, our approaches are limited to obtaining graph hierarchies through persistence within the data domain, i.e., within the image space when computing simplifications of the MSC. Our work would, therefore, benefit from a generalization of our approaches that could be done, for example, by performing persistence graph filtrations.

In our previous studies, we demonstrated benefits in accuracy and complexity, contributing novel and useful approaches based on hierarchical representations of graphs for GNNs - in adapting the message passing itself or through successive training over multiresolution hierarchies of graphs. As has been shown as a limitation to GNNs generally by Morris et al., our approaches suffer in the expressivity of nonlocality (neighborhoods beyond those that are directly adjacent or some number of hops away) and are limited to node classification. Therefore, they cannot represent significantly higher order and subgroup and subgraph structures [149]. One possible direction, which potentially addresses other previously mentioned limitations and uncertainties with our approaches (such as the

role of heterophily and what resolution of graphs within a simplification hierarchy is informative), would be to derive a graph filtration approach based on the heterophily of nodes.

By constructing graph hierarchies through a functional filtration based on heterophily, we could investigate how hierarchical learning may contribute to combating the obstacle of graph learning when there is high heterophily. Some questions of interest that may be answered by improving on our previous works would be how learning with topological graph hierarchies may address common issues GNNs face due to limitations that face local message-passing models, such as a dependence on local graph node neighborhoods in the graph's topology and the aggregation of learned embeddings being myopic to dissimilarities and diversities between neighbors (such as class).

Some recent similar work of interest addresses the issue of nonlocality of GNN message passing, which could be utilized for the construction of filtration-based graph hierarchies, is that of Morris et al., who introduced a hierarchical 'k' Weisfeiler-Lehman (WL) decomposition of graphs for a multiscale resolution of graphs for learning. Rieck et al. also use a persistence-based approach for constructing WL subtrees with desirable topological properties such as cycles and connected components [149]–[151], and notably, Hofer et al. use persistent homology to learn a filtration with a trainable filtration function [152]. As these approaches focus on graph classification to apply the techniques used in a topological setting using our methods, we would first need to reframe the task of node classification as a problem of graph classification.

5.1 Improving Graph Neural Networks with Filtration Learning

5.1.1 Background

Contemporary machine learning has seen significant advances with the introduction of graph neural networks (GNNs), which process and learn from data represented in graphs. GNNs use message passing and aggregation to capture the structural information of the graph. However, several challenges hinder their wider adoption and effectiveness.

Firstly, **scalability** is a significant issue. As graphs increase in size, the computational and memory requirements for message passing grow exponentially, which is especially problematic for large-scale social networks, biological networks, and other complex systems where the graphs can have millions of nodes and edges. Efficiently processing such large graphs without sacrificing the

richness of the graph structure or node features remains an active area of research.

Another challenge is the **oversmoothing** problem, where repeated application of the message passing and aggregation steps causes node representations to become indistinguishable from one another, resulting in a loss of useful information that can diminish the discriminative power of the network, particularly for deep GNNs. Finding a balance between capturing sufficient **local and global graph structures** without oversmoothing is a difficult task.

Heterophily in graphs also poses a problem for many GNN architectures. These networks typically assume that connected nodes are likely to belong to the same class (homophily), but in many real-world graphs, connected nodes may be very different. GNNs can struggle to learn from such heterophilic graphs because their underlying mechanisms are geared toward leveraging homophily to make predictions.

Moreover, GNNs often assume that the graph structure is given and is an accurate representation of the relationships between entities. However, in many cases, the graph structure may be noisy, incomplete, or uncertain. How to learn effectively from **imperfect graph data** is an area that requires further exploration.

In conclusion, although GNNs represent a powerful class of machine learning models for graph-structured data, challenges such as scalability, oversmoothing, heterophily, dynamics, structural imperfections, anisotropy, and interpretability remain obstacles to their full potential. Overcoming these obstacles will require innovative approaches in model design, training algorithms, and theoretical understanding.

Using topology to understand geometric information, connectivity information, and multilevel subgraph hierarchies in graphs can significantly enhance contemporary machine learning approaches that employ message passing and aggregation-based learning, especially for object classification and image segmentation tasks.

Morris et al. demonstrate the need to improve canonical GNNs by revealing that standard GNNs, akin to the 1-WL test, cannot discern complex graph structures, thus limiting expressiveness. The equivalence of 1-WL and GNNs solidifies a stronger understanding of the limitations of GNNs regarding their expressivity of nonlocality, ability to identify and represent significant higher order structures and subgroup and subgraph structures, and their ability to convey higher order group-wise interactions (connectivity). Recent advances aiming to interweave topology and GNNs through scale-dependent simplicial complexes and learned topological filtrations of graph [105], [152] or

even that of our recent works highlight that developing topological machine learning algorithms could advance graph learning.

5.1.2 Motivations

Empirical results from our past works suggest that by better understanding and developing learning models around intrinsic topological adaptations or the extrinsic use of topological representations, we can enhance machine learning models and methods in computational complexity, accuracy, and capability. We suspect, as suggested by the use of topological priors and sequences of complexes growing in resolution, that structure and connectivity illustrated by topological summaries can be further generalized and incorporated to inform learning (Sections 3.2 and 4.2).

By embedding local geometric information into node or edge features, GNN models have the potential to better capture the spatial relationships and structural nuances within data, which is essential for image segmentation. Multilevel sequences of growing resolution could then allow for the encapsulation of local graph structures at different scales, enabling GNNs to capture both fine-grained and coarse-grained patterns within the data. This hierarchical representation may address the oversmoothing problem by preserving node feature diversity across network layers, as information is processed at varying resolutions. Additionally, it may enable GNNs to effectively manage heterophily by distinguishing between different types of node relationships and aggregating messages accordingly. Overall, leveraging multilevel subgraph hierarchies can lead to more powerful, interpretable, and generalizable GNNs that are capable of tackling the complex, structured data encountered in numerous real-world applications.

Future directions of research have emerged from our proposed graph learning method called "Hierarchical Successive Training (HST)," where we use low resolution graphs in a sequence of graphs obtained from persistence filtration to preinitialize higher order graph node embeddings for graph learning as described in Section 4.4.1. We found that preinitialization of node embeddings from learned lower graph complexity node embeddings improved the timing and accuracy performance of GNNs, as shown in Section 4.7, suggesting it is possible to further improve GNN performance with preinitialization. We propose that preinitialization of graph node embeddings with models more robust to heterophily will improve GNNs generally. To test this, we aim to preinitialize graph node embeddings using MLPs and then use the preinitialized graph for node classification with GNNs. Our approach will be similar to that of Han et al. However, their work [153] does

not explore the potential use of MLP initialization to combat heterophily in graph learning for GNNs. As MLPs have been shown to be more resilient to heterophily [154], [155] by using them for preinitialization, it may be possible to train a cheaper (and possibly more robust to heterophily) model for a portion of the training process to be then used to initialize the training of a full GNN model and reach higher accuracy, be more robust to hyperparameter choices, and require less time computationally.

We also propose further exploring the use of a GNN trained on a coarser version of the graph as the initial solution, rather than MLP, as we have shown in Chapter 4 to determine whether sequences of subset graphs obtained from filtration might lead to greater accuracy boosts than MLP in either graphs with homophily or heterophily. We present an approach that uses persistence homology to obtain topologically informative hierarchies of graphs for hierarchical graph neural networks. We propose first formulating learning problems over the edges of graphs where edge attributes are derived from node relations and geometric connectivity. Performing edge classification to infer the probability of relations between nodes (heterophily) would then allow filtration techniques for hierarchical learning, which could then more likely learn expressive connectivity and geometric structures within graph representations based on topological summaries of data as discussed in Sections 3.2 and 4.2, but using a filtration function based on the number of heterophilous edges for each node. Using subset graphs gradually increasing in the sequenced filtration, we can apply our proposed initialization approach by first training on low heterophilous subgraphs, which are then used to initialize higher superlevel graphs in the graph sequence before continuing training.

We suspect further developing the approach of employing and implementing topological summaries in GNNs will provide a means to address the obstacles of the inability of GNNs to express specific local structures, the sparsity of local structures in topological summaries, the impact of heterophily to the quality of learned embeddings based on local neighborhoods, and the impeded expressivity of GNNs to learn global structures.

5.2 Topology-Aware Graph Neural Networks: Enhancing Node Classification Through Filtration Learning

Graph neural networks (GNNs) are a powerful method of learning representations of graph-structured data. They excel at learning class-discriminative representations of nodes in homophilous graphs, where connecting nodes tend to belong to the same class. However, many graph neural networks struggle with heterophilous graphs whose nodes tend to connect to others of different classes since it has been shown that the heterophilous edges can muddy the message passing. Inspired by this finding, we propose to design a topological filtration scheme for the graph based on the probability that a classifier predicts an edge is heterophilous. Using topological filtration based on predicted edge assignments, we can create an ordered sequence of subgraphs. As filtration helps track the evolution of graph connectivity and class connectivity over time, the resulting sequence of nested graphs captures the appearance and merging of connected components and cycles through heterophilous edges. We introduce two methodologies to exploit the sequence of graphs obtained through filtration to train a backbone GNN model. Both methodologies aim to reduce oversmoothing by enhancing the influence of early birth nodes in subgraphs with higher co-class connectivity. The first trains a GNN on each graph in the filtration sequence consecutively for a portion of the total training time, using embeddings from previous graphs to initialize node embeddings in subsequent graphs. The second approach uses a novel message passing scheme to pass messages jointly within each graph and between graph levels with common nodes. The second approach introduced further increases the impact of informative relationships while filtering out uninformative ones through a learnable attention scheme and node filter function for each graph level before multi-scale aggregation. Experiments show that our proposed method improves node classification accuracy on heterophilous and homophilous networks alike.

5.3 Introduction

Graphical representations offer an accessible and actionable means to express and expand our understanding of the world, from observed phenomena to abstract notions, by providing explanations or interpretations based on relationships and connections between elements, circumstances, and complex systems. Graph Neural Networks (GNNs) have emerged as a powerful tool for learning from graph structured data by generalizing the convolution operator to unstructured domains by leveraging message passing or neighborhood aggregation schemes to harness structural information.

Recently, in graph machine learning, the classical task of node classification has demonstrated strong results when employing GNNs.

Despite their potential, GNNs face several known challenges limiting their effectiveness and expressivity; of these obstacles, message passing networks schemes probably can not distinguish specific topologies, such as two triangles from a 6-cycle, and a leading known impediment to GNNs are graphs with high heterophily, namely, graphs whose edges predominantly adjoin nodes of disparate classes, which leads to oversmoothing [156]–[158]. This paper introduces a graph learning methodology that addresses the limitations of conventional GNNs by offering a novel approach employing hierarchical GNNs-leveraging topological insights from persistent homology, namely persistence filtration, to reveal meaningful structure in graphs informed by class connectivity and allowing for a hierarchy of multilevel graph resolutions.

The methodologies introduced here learn class relationships between nodes by first initializing edge embeddings from the combined embeddings of the nodes each edge adjoins and assigns edge-level labels as heterophilous or homophilous, learns edge embeddings, and then identifies the likelihood each edge is heterophilous or homophilous. We then leverage the probability assigned to each edge as separating node classes with persistent homology through a persistence filtration function defined over edge values to obtain a multilevel resolution sequence of graphs well suited for hierarchical GNNs. By formulating the learning problem over graph edges, integrating topological summaries, and learning multilevel subgraph hierarchies based on persistent filtration dependent on class connectivity, our approach offers an ordered sequence of graphs that captures both local and global structural information while also addressing critical issues such as over smoothing, reducing computational complexity, and a paradigm for GNNs more resilient to heterophily. We argue that by providing a more nuanced understanding of graph topology and leveraging hierarchical learning, our proposed approach can lead to more resilient, computationally efficient, and expressive GNNs. Empirically, we show that our approach to persistence filtration learning and hierarchical graph learning compares favorably to previous learning techniques, including standard learning models, standard GNNs, and heterophily-specific models. We also demonstrate competitive performance results in accuracy and complexity over established benchmark datasets for evaluating GNN performance in low and high heterophily scenarios.

Our contributions are as follows:

- We propose a homophily-based topological filtration method for graphs, where heterophilous

edges may be used for learning but given different emphasis compared to homophilous edges. Our filtration preserves topological structure compared to approaches that simply drop heterophilous edges.

- We introduce two different methods for improving backbone GNN models, especially on graphs with heterophily, by allowing them to learn from a multi-scale sequence of graphs based on class connectivity and topological filtration.
- We experimentally demonstrate that our methods lead to improved node classification accuracy relative to previous models and have the potential to filter out edges added in adversarial attacks.
- A relaxed constraint on GNNs to learn a graph representation that a standard GNN can perform best on.

In summary, this chapter highlights the transformative potential of hierarchical graph learning methodologies to address the inherent limitations of conventional Graph Neural Networks, particularly in challenging heterophilous graph structures. By introducing a persistent homology-inspired filtration approach, this work provides a nuanced framework for leveraging topological insights to construct multilevel graph representations. The proposed methods not only enhance the expressivity and resilience of GNNs but also demonstrate superior performance in node classification tasks across varied scenarios, including adversarial settings. By integrating topology-driven edge embeddings and multi-scale graph hierarchies, this approach represents a significant step toward more robust, efficient, and adaptive graph learning paradigms, setting the stage for broader applicability and innovation in graph machine learning.

5.4 Background

5.4.1 Filtration Functions and Topological Filtration

We now provide the necessary background on topological filtration, how it applies to graphs, and recent advancements affording learnable topological filtration schemes.

5.4.1.1 Topological Filtration

Now, let $(K^i)_{i=0}^m$ be a sequence of simplicial complexes such that $\emptyset = K_0 \subseteq K_1 \subseteq \cdots \subseteq K_m = K$. Then, $(K^i)_{i=0}^m$. Then, $(K^i)_{i=0}^m$ is called a filtration of K. In the filtration of K, the sequence of

each simplicial complex K_i is subset to the subsequent $K_{j>i}$.

5.4.1.2 Filter Functions

Let $\mathbb{E}$ be the domain of simplices $\mathbb{K}$ the set of possible simplicial complexes over $\mathbb{E}$, the filter function $f : \mathbb{R} \times \mathbb{K} \times \mathbb{E} \to \mathbb{R}$, $(K,e) \mapsto f(K,e)$ for $K \in \mathbb{K}$, $e \in E$. The filter function then maps each simplex to a real value.

5.4.1.3 Learnable Filter Function

A Learnable filter f_ε function differs in that it is differentiable in ε. A learnable filter function is then $f : \mathbb{R} \times \mathbb{K} \times \mathbb{E} \to \mathbb{R}$, $(\varepsilon,K,e) \mapsto f(\varepsilon,K,e)$ for $K \in \mathbb{K}$, $e \in E$. Then, we call f a learnable vertex filter function with parameter ε. It has been shown that if pairwise simplex values are distinct, such a differentiable filter function can be used in the context of k-persistent homology as a mapping of simplicial complexes that is differentiable end to end [159].

5.4.1.4 Construction of Topological Filtration

The filtration is constructed by including simplices in increasing order of their filter values. Simplices are first assigned values through a filter function or a filter function for the compassion of simplices is derived. Starting with the empty set, simplices are added consecutively in sorted order by increasing the filter function value. Simplices are added such that the resulting simplicial complex contains the previous subset simplicial complexes for lower filter function values. Each step corresponds to subcomplex K_p where p is the threshold value. For filter function $f(e)$ over simplices $e \in \mathbb{E}$ the subcomplex K_p during filtration contains all simplices with filter value less than p. The set of simplices in K_p is then $\{e \in K_p | f(e) \leq p\}$. As the filtration progresses, you can track how topological features like connected components, loops, and voids appear and disappear. These features are summarized using tools like persistent homology.

5.4.1.5 Topological Filtration of Graphs

Graphs are simplicial complexes. We can interpret a graph G as a 1-dimensional simplicial complex whose vertices (0-simplices) are the graph nodes and whose edges (1-simplices) are the edges. In the topological filtration of graphs, we can think of filtration as a growth process of a weighted graph. As the graph grows, new edges and nodes are introduced, introducing new connected components and neighborhood structures.

Node neighborhoods thus share this property: for each node $v_i \in \mathbf{V}_{p_i} \cap \mathbf{V}_{p_j}$, e.g. $\mathcal{N}_{p_i}(v_i)$ for $v_i \in \mathbf{V}_{p_i}$ and $\mathcal{N}_{p_j}(v_i)$ for $v_i \in \mathbf{V}_{p_j}$, we have that $\mathcal{N}_{p_i}(v_i) \subseteq \mathcal{N}_{p_j}(v_i)$. We give all graphs the full node set found at the lowest threshold level of the filtration sequence, but nodes that would not otherwise exist in the graph at a higher persistence level are disconnected. Early birth nodes, which appear for lower value thresholds, then remain in subsequent graphs in the filtration sequence, with neighborhood structure for each node potentially differing for each node between different graphs in the sequence.

5.4.2 Graph Neural Networks for Heterophilous Graphs

For a recent overview of graph neural networks designed to handle graphs with high heterophily, see [160], [161]. Methods designed to handle graphs with heterophily usually focus on one of two challenges posed by heterophily. The first is that in heterophilous graphs, nodes may not have relevant neighbors (e.g. neighbors of the same class) to attend to, and one solution is to expand the receptive field of graph neural networks to allow them to learn non-local dependencies. Such approaches include higher-order neighborhood aggregation, proposed by [126], rewiring the graph based on other forms of node similarity [162], or introducing non-local attention [126]. Such approaches may incur a higher computational cost.

Our work falls into a second category of approaches, which recognize that heterophily also may muddy the message passing of graph neural networks by causing messages to be aggregated from irrelevant neighbors. Such approaches modify the message passing to better suit heterophilous graphs, for example by performing it over signed [163] or directed [164] edges, or by learning a class compatibility matrix that may be used to control message passing [165]. Other approaches include rewiring the graph to be better suited to heterophily, by connecting nodes based on some other measure of similarity or distance [162], [166], [167].

Most related to our work, this rewiring has been performed with a edge classifier that predicts if an edge is heterophilous or homophilous [168], [169]. A backbone graph neural network is trained on a graph that has been preprocessed using the classifier to improve its homophily properties. The edge classifier may be used to filter the graph only by pruning heterophilous edges [168], or new edges may also be added that are predicted to be homophilous [169]. Both methods result in a single output graph which loses any information that the heterophilous edges may contain. In contrast, our approach gives a model access to the input graph at a variety of filtration levels, giving us the best

of both worlds: an ability to use all the information in the original graph, while also being able to emphasize the high-homophily connections in a more filtered graph.

5.4.3 Topological Data Analysis for Graph Neural Networks

Graph filtration learning using persistent homology has been used for graph-level representation learning [159]. For node classification, persistent homology has also been used to guide message passing in graph neural networks [170] for semi-supervised node classification. More related to our work, [171] has used graph neural networks to classify topological objects in image data, learning successively or jointly from hierarchical levels of a topological filtration. Here, we show how to derive such a hierarchical filtration for graphs specifically based on homophily, and we develop a node-level filtration that allows us to attend to different levels of filtered graphs.

While our work focuses on persistent homology, it is possible to use other topological concepts to design neural networks for graphs. Neural sheaf diffusion [172] is another topologically-inspired technique to improve the performance of graph neural networks on heterophily, built on a different set of learned topological structures called sheaves and designing a sheaf convolution operation analogous to graph convolution. A Differentiable Cell Complex Module has also been proposed to learn a cell complex and perform message passing over the resulting cell complex, resulting in a tradeoff between leveraging the expressivity of the resulting topological structure and maintaining computational tractability [173].

5.5 Methodologies for Filtration Learning

5.5.1 Estimating Edge Homophily Scores

Let $G = (\mathcal{N}, \mathcal{E})$ be a graph consisting of node set $v \in \mathcal{N}$ and edge set $\{e = (u,v) | e \in \mathcal{E} \text{and } u,v \in \mathcal{N}\}$ where edge $e = (u,v)$ is said to adjoin nodes u and v. We denote a neighborhood around node v as $N(v) = \{u \in \mathcal{N} | (u,v) \in \mathcal{E}\}$ to consist of all nodes in $\mathcal{N}$ that share an incident edge in $\mathcal{E}$. For $k \geq 1$ we denote the $k-$hop neighborhood of node v as $N_k(v) = \{u \in \mathcal{N}$ such that there is a connected path of k or fewer edges adjoining u and v.

The class balance ratio is defined as the total positively labeled foreground over the total negatively labeled background nodes. For nodes u and v with labels y_u and y_v and edge set $\mathcal{E}_i$ for graph G_i in the filtration sequence, the homophily ratio [126] is given by $h_i = \frac{1}{|\mathcal{E}_i|} \sum_{v \in \mathcal{E}_i} \frac{|\{u \in \mathcal{N}(v) : y_u = y_v\}|}{|\mathcal{N}(v)|}$.

To perform topological filtration of a simplicial complex it is first necessary to have a value

assignment over, or comparative criteria between, simplicial cells that affords an ordering. Based on the filter value and beginning with the lower bound in the range of filter values, simplices can be added in order of increasing value. Considering a graph as a 1-D simplicial complex comprised of 0-cell simplex vertices (graph nodes) and 1-cell simplices (graph edges), if the graph nodes or edges are naturally imbued with a weight or value a topological filtration can be performed directly.

As we aim to employ hierarchical graph learning in a manner to combat oversmoothing our methodology focuses on edges, and specifically, the likelihood an edge adjoins nodes of desperate classes. To accomplish this we first frame the task of node classification as an edge classification problem, where the edge classifier serves as a filter function prescribes filter values to edges through the inferred likelihood that an edge is homopholous.

Given a graph $G = (\mathscr{N}, \mathscr{E})$ containing labeled nodes v with labels y_v and node features h_u and h_v for a subset or all of node set $\mathscr{N}$ we can derive binary edge labels for edge $e = (u,v)$ as $y_{(u,v)} = \left\{ \begin{array}{c} 1 \text{ if } y_u = y_v \\ 0 \text{ if } y_u \neq y_v \end{array} \right\}$ for graphs with binary or multi-class labels. Similarly, a derived edge feature for each edge $e = (u,v)$ can be derived from nodes features h_u and h_v the nodes u and v it adjoins as $\text{COMBINE}(h_u, h_v)$ where combine is a functional operator that merges or aggregates the node features of the nodes made adjacent by the given edge such as an element-wise operation, a weighted sum, the dot product, or concatenation. For our purposes, we use concatenation. The resulting process provides a subset of binary edge labels and features for all edges.

We define an edge filter function to be a multilayer perception with a single logistic output defined as $\text{MLP}\big(\text{COMBINE}(h_u, h_v)\big)$. In our study, we define $\text{COMBINE}(h_u, h_v)$ as the concatenation of the feature vectors h_u and h_v of adjacent nodes' u and v. Defined over edges $e \in \mathscr{E}$ for edge set $\mathscr{E}$ of graph G is then

$$f : \mathbb{R} \times G \times \mathbb{E} \to \mathbb{R} \text{ , } (\mathscr{E}, e = (u,v)) \mapsto f(\mathscr{E}, e)$$

where

$$f(\mathscr{E}, e) := \text{MLP}(\text{CONCAT}(h_u, h_v))$$

We then train the filter function using the derived homophily edge labels $y_{u,v}$ on a subset of the graph and infer values between 0 and 1 to all remaining edges of the graph, predicting whether an edge is homophilous or not.

5.5.2 Homophily-Based Topological Filtration of Graphs

We perform filtration on graphs using a sorted ordering of 1-simplices, or edges, with filter values assigned by $f(\mathscr{E},e)\forall e \in \mathscr{E}$ of graph G, which correspond to the likelihood an edge is homophilous or not. Beginning at filter threshold $p = 0$ we then add edge $e = (u,v)$ and its incident nodes u and v if $f(e) \leq p$ and such that the expanded graph G_p at level p of the filtration sequence remains a simplicial complex containing all $e = (u,v)$ such that $f(e) \leq p$ Once all edges have been assigned a filter function value.

We observe the topological filtration sequence at P levels of filtration thresholds to model multiscale topological information. In contrast to hierarchical graph-level representation learning methods that learn to pool each input graph [142], this gives us several graph representations of the underlying data to use as input to a graph neural network, which we denote the *multi-scale sequence of graphs* or *graph hierarchy*. Computationally, a smaller threshold value produces a graph higher resolution graph, later in the nested subset sequence of graphs in the filtration sequence. Thus, we obtain a hierarchy of graphs $G_1,\ldots,G_P$ where $\forall p_i$ for $i \in [1,\ldots,P], G_{p_{i-1}} \subseteq G_{p_i}$: that is, $G_{p_{i-1}}$ is an induced subgraph defined on a subset of the vertices in G_{p_i}.

5.5.3 Graph Neural Networks with Topological Filtration

We consider two methods for learning node representations for classification using GNNs, which exploit the topological information from the nested sequence of graphs obtained through topological filtration, referred to as the graph hierarchy, in the context of training GNNs. The first method modifies the training of GNN over graph hierarchies, while the second modifies the architecture to pass messages jointly between all hierarchical graph levels. We further introduce two novel methodologies for multi-scale joint aggregation to learn a more informative graph representation for GNNs. The first introduced a novel attention based aggregation that learns to weigh the importance and contribution of nodes within each graph level of the hierarchy during multi-scale joint aggregation, and the second learns a topological filtration of nodes through a learnable filter function that assigns values to nodes used for topological filtration at certain stages of training.

5.5.4 Illustrated Example of Multi-Scale Joint Aggregation

5.5.4.1 Neighbor Messages

We aim to learn **neighbor representations** for each target node, where the target node's embedding representation is obtained from the target node v's neighborhood r, denoted $\mathcal{N}_r(v)$ for node v. In 5.1 node v is highlighted in yellow within each graph of the nested filtration sequence and the dotted blue arrows denote the messages passed to node v in each neighborhood $N_p^r(v)$ where p denotes the graph level in the sequence at filtration value $p \in \{0, \cdots, P\}$ that the neighborhood being considered corresponds to. A single node that exists within multiple graphs of the filtration sequence can have neighborhood structures that differ but are subsets to subsequent neighborhoods later in the sequence of graphs resulting from filtration.

$$h_m^r(u,v) = \text{Message}(h_u^r, h_v^r | v \in \mathcal{N}_p^r(v)))$$

where h_u and h_v are feature vectors of nodes u and nodes v.

5.5.4.2 Aggregation Within Neighborhoods (blue)

Messages are then aggregated for each neighborhood $N_p^r(v)$ across nodes in the target node's neighborhood. In Figure 5.1 this is illustrated with the regions highlighted in blue where, of each graph, messages from each neighborhood r of v in that graph are aggregated.

$$h_{v,p}^r = \text{AGG}_{u \in \mathcal{N}_p^r(v)}(h_m^r(u,v))$$

5.5.4.3 Aggregation Between Neighborhoods (purple)

Messages are then aggregated between all neighborhood $N_p^r(v) \in N_p(v)$ of nodes in target node v's neighborhood in subspace p

$$h_{v,p}^{N_p} = \text{AGG}_{\mathcal{N}_p^r \in N_p}(h_{v,p}^r)$$

5.5.4.4 Multi-Neighborhood Aggregation Across Graph Neighborhoods (orange)

This work introduces the step highlighted in orange in Figure 5.1. Namely, once messages from each neighborhood have been aggregated, messages are then aggregated across all target node neighborhoods in all graphs within the sequence of graphs to which target node v belongs.

$$h_v = \text{AGG}(\{h_{v,p}\}) \tag{5.1}$$

5.5.4.5 Update

Lastly, the feature representation of the target node is updated with aggregated embeddings

$$h_v = \text{Update}(h_{vP}^N, h_a^t)$$

5.5.5 Topological Filtration for Hierarchical Graph Neural Network

5.5.5.1 Multi-Scale Successive Training

For our first method, we use a GNN which learns node features via message passing in the spatial domain [84]. For the i^{th} persistence level p_i (moving from sparsest to densest), the aggregated node embedding for node v_i from neighbors $u_i \in \mathscr{N}_{p_i}(v_i)$ of persistence subgraph G_i is given by

$$h^{k_i}_{\mathscr{N}_{p_i}(v_i)} \;=\; \sigma\big(AGGR(h^{k_i-1}_{u_i} \;:\; \forall\, u_i \in\; \mathscr{N}_{p_i}(v_i))\big)$$

Node v_i's updated embedding is concatenated with the target node embedding from the previous iteration of aggregation $h_{v_i}^{k_i}$ and the aggregated neighbor features, parameterized with target embedding and neighbor embedding weight matrices $W_s^{k_i}$ and $W_n^{k_i}$ respectively. Similarly, the target node embeddings and neighboring node embeddings are passed through a two-layer multilayer perceptron (MLP), which after mean-pooling, are multiplied with weight matrices W_s and W_n The target and neighbor node representations are combined (with concatenation), and the combined embeddings are fed through a nonlinear function σ, specifically a fully connected network with a nonlinear activation function.

Instead of training a GNN for N epochs on the finest graph G_P, we train it for $\frac{N}{P}$ epochs each on graphs $G_1, \ldots, G_P$, using the embeddings from G_{i-1} to initialize the embeddings of corresponding nodes in $G_i \;\forall\, i \in [1 \ldots, P]$. The MLPs and node embedding weight matrices for target and neighboring node embeddings are shared between graph hierarchies. Training begins with the smallest subset graph and iteratively increases in graph size. We also successively share the learned embedding weight matrices and matrices of the MLPs as well.

As a result, when we begin training on graph G_p, the embedding for node $v_p \in \mathscr{V}(G_p)$ is the combined embedding from previous aggregations at lower persistence subgraphs:

$h_{v_p}^{k_p} = COMBINE\big(h_{v_{p-1}}^{k_{p-1}}, h_{v_p}^{k_p-1}\big)$ where we implement COMBINE with concatenation.

5.5.5.2 Multi-Scale Joint Training

For node v_i belonging to node set $\mathbf{V}_{p_i}$ of subgraph $\mathbf{G}_{p_i}$, the feature representation $h_{v_i}^k$ at GNN layer $k \in \{1, \cdots, K\}$ first combines self-representations from the previous level of aggregation with aggregated neighbor information:

$$h_{v_i}^k = COMBINE\big(W_s^{k-1} h_v^{k-1}, \\ AGGR\big(\{W_n^{k-1} h_u^{k-1} : u \in \mathscr{N}_{p_i}(v_i)\}\big)\big)$$

Once this is done, the resulting embedding is combined with the embedding representation from other persistence subgraphs. As an example for two persistence levels p_i and p_j where $v_i \in \mathbf{V}_{p_i} \subset \mathbf{G}_{p_i}$ and $v_j \in \mathbf{V}_{p_j} \subset \mathbf{G}_{p_j}$, we have $h_v^k = \sigma\big(COMBINE\big(h_{v_i}^k, h_{v_j}^k\big)\big)$.

5.5.6 Learned Node Filtrations of Graphs

This work introduces a second, learnable node filter function for learning the filtration of nodes in graphs within the graph hierarchy during training. The purpose of a learnable node filter function in this context is to, given a multi-scale sequence of graphs for learning, obtain a filtration of graphs of a multiscale sequence of graphs in the graph hierarchy. Filtration of nodes in the graph hierarchy aims to learn which graphs (and their respective nodes) in the nested graph sequence obtained from topological filtration are most informative for multi-scale or hierarchical graph learning. Graph Isomorphism Networks (GIN-ε) have been shown to not only be a more expressive learning model, capable of distinguishing different graph structures but also a viable candidate as a learnable filter function for persistent homology when also combined with a multilayer perceptron [16], [159]. We then use a learnable node filter function during Multi-Scale Joint Training defined as

$$f_\varepsilon : \mathbb{R} \times G \times \mathbb{V} \to \mathbb{R}\ ,\ (\varepsilon, \mathscr{N}_p, u) \mapsto f_{p,\varepsilon}(\varepsilon, \mathscr{N}_p, u)$$

where

$$f_{p,\varepsilon}(\mathscr{N}_p, u) := \mathrm{MLP}(\mathrm{GIN}_\varepsilon^p(h_u^p)) \to \mathbb{R}$$

where h_u^p is the embedding representation of node u in $\mathscr{N}_p$ in graph $G_p = (\mathscr{N}_p, \mathscr{E}_p)$ for graph level p of the graph hierarchy and $f_{p,\varepsilon}$, a differentiable node filter function using a $\mathrm{GIN}_\varepsilon^p$ which is learnable and differentiable in ε and MLP is a multilayer perceptron which takes the resulting embedding from

GIN_{ε}^{p} and produces a single dimensional output. The resulting real output of $f_{p,\varepsilon}$ is the learned filter value, denoted q_p, for node u in graph level G_p.

This work introduces a methodology for using the learned filter value q_p to filter graphs within the graph hierarchy. To learn the importance and reduce or increase the contribution of graphs in the graph hierarchy during multi-scale joint aggregation we use the q_p filter value for attention type based aggregation by weighing the aggregated embedding obtained from graph G_p in the hierarchy before multi-scale aggregation across all graphs G_i in the multi-scale filtration sequence of P graphs $G_0, G_1, \cdots, G_P$. Before the combination step of multi-scale aggregation, we then factor each node's resulting embedding from each graph level by its respective filter value q_p In doing so, the learned filter value can learn to eliminate or accentuate the contribution of embeddings from specific graphs in the multi-scale graph sequence.

The learned filtration of graphs in the graph hierarchy during multi-scale joint aggregation can then be expressed as the combined embedding from each subgraph, where each subgraph's embedding is obtained with a back bone GNN and a filter function $f_{p,\varepsilon}(\mathcal{N}_p, u)$, unique to each graph level, for a final embedding that is weighted by a filter value q_p before global combination. For node v_P in the highest resolution graph G_P in the filtration sequence, we then have:

$$\begin{aligned} h_{v_P}^{k} = COMBINE\big(&f_{0,\varepsilon_0}(h_{v_0}^{k-1}) \cdot W_{s,0}^{k-1} h_{v_0} h_{v_0}^{k-1} \\ &, \cdots, f_{i,\varepsilon_i}(h_{v_1}^{k-1}) W_{s,i} h_{v_1}^{k-1} \\ &, \cdots, f_{P,\varepsilon_P}(h_{v_P}^{k-1}) W_{s,P} h_{v_P}^{k-1}\big) \end{aligned}$$

In this setting, each $f_{i,epsilon_i}(\cdot)$ is a learnable filter function for each graph level G_i where the resulting output q_i can serve as a learned attention factor for aggregation or node filter function value. Once a learned filter value is assigned to nodes, performing another level of topological filtration over the initial multiscale graph sequence is also possible. With filter values assigned to nodes, one can easily perform a topological filtration of each subgraph in the graph hierarchy, effectively gaining a more informed multi-scale graph filtration sequence suitable for learning and preserving topological information contained in each graph and consisting of more informative hierarchical graph representations.

We note, this generalizes Equation 5.1 to use the learned filtration weights as attention scores when combining messages from different filtration levels.

From the original precomputed filtration sequence of graphs, we then learn a subset filtration sequence of graphs through the learnable node filter function $f_p(\mathcal{N}_p, u)$, which learns a weighted attention on each node in each graph level to obtain a learned filtration sequence of graphs.

5.6 Experiments

5.6.1 Implementation

We implement all models with PyTorch and all GNNs with Pytorch Geometric [174], [175]. Topological filtration is performed with Dionysus [176].

5.6.2 Hyperparameters and Computing Environment

We employ early stopping based on validation scores (computed every 8 epochs) for a deviation of $1e-3$, with patience of 4 rounds of validation accuracy scores for the edge filter function's training and 10 for the multi-scale model used. For dropout, the edge filter function uses a value of 0.3, and multi-scale models have a value of 0.65. As the size of the datasets tested varied, hidden dimensions varied to account for the limitations of the machine being used and model overfitting. We note that overfitting is a challenge on many common heterophilous graph benchmarks [160], and experienced this especially with our larger joint message passing models, leading us to use aggressive dropout to combat this. We perform a parameter sweep for each of the following hyperparameters over the values provided in Table 5.1.

All experiments were run on a laptop with 3 GB GeForce GTX 970M with 1280 CUDA Cores GPU and 3.5GHz i7-6700HQ processor running Ubuntu, Linux.

5.6.3 Datasets

We now go over the datasets used and the node classification objective. We provide a summary of dataset statistics in Table 5.2.

5.6.3.1 Planetoid (CiteSeer, Cora, and PubMed)

The Planetoid datasets, including CiteSeer, Cora, and PubMed, are widely used citation networks. In these datasets, nodes represent documents, and edges represent citation links between them. The task is typically node classification, where the goal is to predict the category of each document [177].

5.6.3.2 WikipediaNetwork (Chameleon)

is a graph derived from the page-page link network of Wikipedia articles. Nodes represent Wikipedia pages, and edges represent hyperlinks between them. The nodes are classified based on the traffic levels of the web pages. This dataset is used to study graph heterophily [178].

5.6.3.3 WebKB (Cornell, Wisconsin, and Texas)

The WebKB datasets, including Cornell, Wisconsin, and Texas, consist of webpages collected from computer science departments of various universities. Nodes represent webpages, and edges represent hyperlinks between them. The task is to classify the type of webpage (e.g., course, faculty, student) [179].

5.6.4 Node Classification Results

In Table 5.3, we report the test accuracy results of MsST and MsJT in the context of the reported accuracies of other state-of-the-art models with equivalently proportioned train, validation and test splits (we use 60%, 20%, and 20% respectively, following [162] and the subsequent works we compare against)[1].

First of all, we note that our methods excel in all except one dataset, achieving the highest and second highest accuracies. This indicates the potential of our hierarchical filtration to benefit GNNs across the homophily spectrum.

Second of all, while the improvement is modest on the high-homophily datasets Pubmed and Cora (though we do observe a larger percentage improvement on Citeseer), we see a pronounced improvement on several of the high-heterophily datasets. For instance, MsST is over 10% more accurate than the best baseline considered on the Texas dataset, and close to 10% on the Cornell dataset (these are the two datasets with the lowest homophily ratios). This may suggest that indeed our homophily-based filtration is effective especially in high heterophily graphs, where at some levels the filtration uncovers high-homophily subgraphs for the model to learn from.

Finally, we note that on all datasets, our model outperforms SA-SGC, which simply uses the edge classifier to remove heterophilous edges from the graph. This may suggest that by keeping access to them for our model, but simply only at some filtration levels, we can achieve superior

[1]For Geom-GCN, we report results from the original version of the paper: https://openreview.net/attachment?id=S1e2agrFvS&name=original_pdf.

performance. Heterophilous edges are not necessarily just noise.

5.7 Reflection

We propose graph neural network methods that learn from hierarchies of graphs representing a sequence of nested graphs obtained from topological filtration. Treating graphs from the perspective of 1-D simplicial complexes, our filtration is defined as being over 1-cells (graph edges). We obtain a sequence of simplicial complexes, where each complex in the series is subset to the subsequent, from a sorted ordering of 1-cells, thresholded by filtration values prescribed to simplexes through a filter function defined as a learning model trained to infer if edges are homophilous, akin to a class similarity measure between 0-cells (graph nodes). The proposed approach is generalizable to any edge based classification task or graph with weighted edges that afford an ordering. We then present two methods for hierarchical graph learning. The former, achievable for any standard GNN model, performs message passing across subsequent graphs in the graph hierarchy, aggregating node embeddings sequentially moving up the graph filtration sequence. the ladder introduces a novel multi-scale message passing scheme with aggregation performed jointly across graphs in the filtration sequence, with learned attention based aggregation or learned topological filtration of subset graph nodes, effectively learning the degree of contribution graphs within the hierarchy contribute to learned aggregated node embeddings. Our results demonstrate promise for increased accuracy and improvement in training time compared to conventional GNNs while we provide exploratory insights into the effect of class imbalance and heterophily in graph learning.

5.8 Figures and Tables

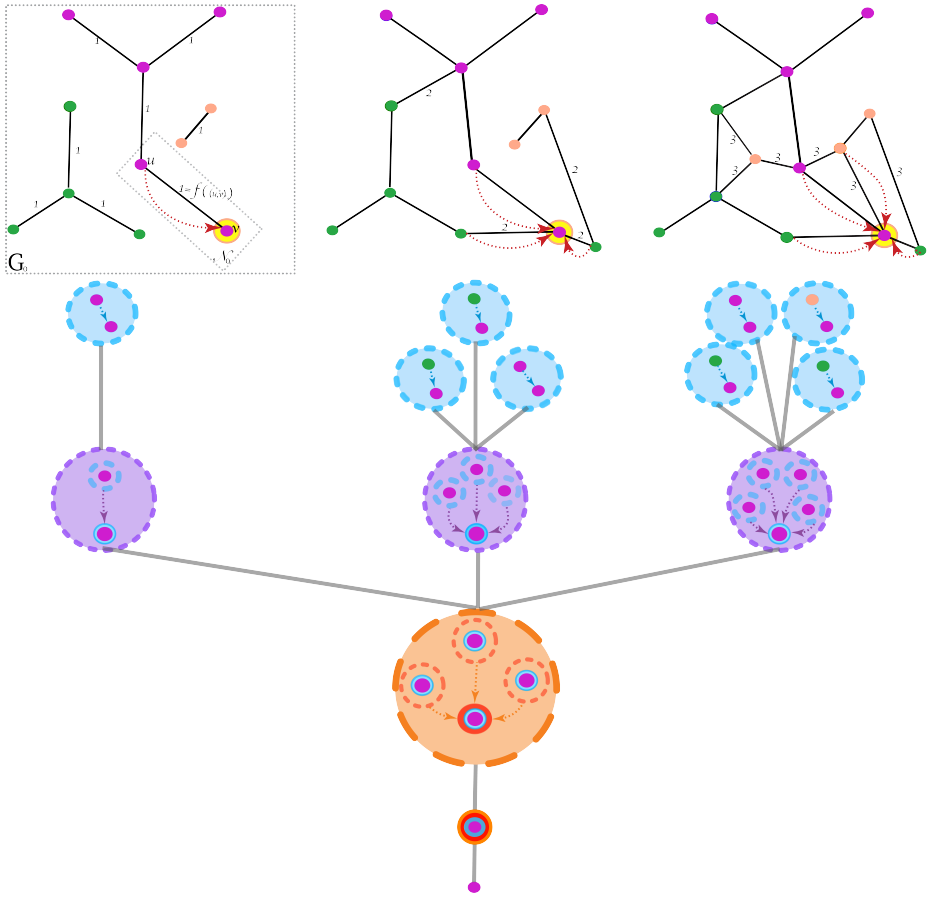


Figure 5.1: A sequence of three nested graphs from topological filtration (top row) offering three levels of graph resolution. Messages are shown being passed (dotted arrow) to the bottom right node that, having survived the filtration, is common to all graphs in the multi-scale sequence, differing in neighborhood connectivity. For each subgraph G_p of the total P graphs in the graph hierarchy $\{G_0, \cdots, G_p, \cdots, G_P\}$, messages are passed from each neighborhood $N^r(v)$ to the target node. Messages at each graph level are then aggregated within neighborhoods (highlighted in blue). Messages from all neighborhoods within each graph are then aggregated between one another (highlighted in purple). We introduce (highlighted in orange) across neighborhood aggregation, where messages are passed across neighborhoods in the multi-scale sequence of graphs. The target node's feature representation is then updated with the aggregated embeddings.

Table 5.1: Values considered for different hyperparameters.

Parameter	Values
Filtration thresholds (in addition to input graph)	(0.5,0.7,0.9), (0.8,0.9), (0.6,0.9), (0.7), (0.8), (0.9), (0.1)
Learning Rate of Multi-Scale GNN (MsST or MJT)	3e-4, 3e-3, 3e-2, 1e-4, 1e-3, 1e-2
Learning rate of edge filter function	1e-3,3e-3, 1e-2, 3e-2
Weight decay	5e-8, 5e-6, 5e-5, 5e-4, 0

Table 5.2: Summary of datasets and their characteristics.

Dataset	Subdataset	Nodes	Edges	Classes
Planetoid	Cora	2,708	5,429	7
	CiteSeer	3,327	4,732	6
	PubMed	19,717	44,338	3
WikipediaNetwork	Chameleon	2,277	31,421	5
WebKB	Cornell	183	295	5
	Wisconsin	251	499	5
	Texas	183	309	5

Table 5.3: Test accuracy results for node classification over datasets, sorted by homophily level, for our methodologies (MsST and MsJT), heterophily oriented model designs, and topological deep learning (TDL) models (last two rows). For our baselines, we report the results of the best-performing variant from the original paper. The highest accuracy scores are highlighted in green, with the second highest outlined in blue. We find that our methods outperform the baselines on almost all datasets across the homophily spectrum.

Model	Texas	Wisconsin	Chameleon	Cornell	CiteSeer	Pubmed	Cora
Hom. Ratio	**0.06**	**0.16**	**0.25**	**0.11**	**0.74**	**0.80**	**0.83**
MsST	96.72%	95.62%	90.12%	95.63%	88.73%	71.04%	93.17%
MsJT	89.27%	93.57%	91.25%	91.71%	93.49%	91.78%	95.39%
Geom-GCN [162]	75.67%	76.47%	89.32%	66.66%	79.12%	90.11%	90.57%
H_2GCN [126]	84.86%	86.67%	59.39%	82.16%	77.07%	89.59%	87.81%
SA-SGC [168]	83.52%	84.52%	64.94%	81.12%	76.88%	87.55%	87.36%
DHGR [166]	85.68%	85.01%	74.57%	82.88%	71.96%	80.09%	83.93%
NSD [172]*	85.95%	89.41%	68.68%	86.49%	77.14%	89.49%	87.30%
DCM [173]	85.71%	89.33%	53.76%	N/A	78.60%	88.61%	86.63%

CHAPTER 6

CONCLUSION

As machine learning models matured, the class of data types that were good candidates for learning techniques expanded, and the application domains for their effective use became more varied. In many cases, the data could be described as a set of discrete elements combined with a set of spatial or functional relationships. When viewed from this perspective, topological methods emerged as an effective class of techniques.

In this dissertation, we introduced new and accessible means to incorporate topological priors in learning while remaining in the topological domain. In Chapter 1, we established the foundations for these methods, including computational topology and its applications. In Chapter 2, we generalized the notion of topological priors graphs with a learnable, model-agnostic data structure representing cells within a complex and their relationships beyond adjacency, such as their affiliation with the connectivity of imaged objects. This graph consisted of dimension-independent representations of cells in the Morse-Smale complex and their incidence relationships, tailored for specific classification tasks for any learning model. We demonstrated the practical benefits of using simplicial complexes for learning, with a focus on image segmentation, where labeling, training, and inference occurred in the topological domain over topological priors derived from cells in the Morse-Smale complex.

The approach presented in Chapter 3 was able to learn topological priors by reframing a segmentation problem as a classification task to identify the subsets of topological cells corresponding to the segmentation of the object of interest. The active learning approach operated without prior knowledge of the complex and its neighborhood relationships, emphasizing the identification and understanding of topological elements and their connectivity. This method allowed us to learn topological structures directly rather than leveraging preexisting topological information to aid the learning models. We also introduced an interactive labeling tool for creating topological summaries and labeling topological cells to populate the training set, enabling practitioners to correct predicted outputs and retrain the model, improving its performance. This work provided a methodology

to compare topological object-level predictions to pixel-level predictions across various domains, including medical, neuroscience, and materials science. In particular, we demonstrated improved accuracy, faster labeling, training, and segmentation compared to traditional pixel-based methods. Our work highlighted the integration of topological elements into learning and showcased their practical benefits.

In Chapter 4, we introduced the hierarchical priors graph (HPG), which combined a sequence of prior graphs obtained with successive simplifications of a Morse-Smale complex PGn (fine resolution priors graph), PGn-1, ... PG1 (coarsest resolution priors graph), with arcs connecting nodes corresponding to the same cells at different resolution levels. Additionally, we introduced the reduced hierarchical prior graph (RHPG), obtained by reducing the HPG to use only arcs connecting cells from level i to level i+1. Using the RHPG, we proposed an initialization-based training paradigm for graph neural networks (GNNs) called Hierarchical Successive Training (HST), where GNNs learned from multiple scales of connectivity by sequential training on the prior graphs for N/P epochs each during all N epochs. We extended our contribution further and introduced a novel message-passing scheme for GNNs based on the HPG, which performed message passing along arcs within each prior graph PGi (same level), followed by passing messages along arcs between different prior graphs (different levels). This Hierarchical Joint Training (HJT) aggregated neighborhood information within and between neighborhoods of each prior graph and across multiple prior graphs of different connectivity scales. Unlike other topologically-informed GNN message-passing approaches, our method constructed an ordered set of graphs from successive topological simplifications of image data, allowing learning from multiple scales of topological information.

In Chapter 5, graphical representations offered an intuitive means of interpreting data based on relationships and connections between elements and systems. Graph Neural Networks (GNNs) emerged as a powerful tool for learning from graph-structured data by generalizing the convolution operator to unstructured domains by leveraging message passing or neighborhood aggregation schemes to harness structural information. Despite their potential, GNNs faced several known challenges limiting their effectiveness and expressivity, especially for graphs with high heterophily. This dissertation introduced a graph learning methodology that addressed the limitations of conventional GNNs by offering a novel approach employing hierarchical GNNs, leveraging topological insights from persistent homology, namely persistence filtration, to reveal meaningful structure in graphs

informed by class connectivity and allowing for a hierarchy of multilevel graph resolutions.

The approach presented treated the graph as a 1-D simplicial complex consisting of 0-simplices (vertices) and 1-simplices (edges), where a subset of the vertices had a label and all had a feature vector. We learned class relationships between vertices by first initializing simplex embeddings from the combined embeddings of the vertices each simplex adjoined, assigning simplex-level labels as heterophilous or homophilous, learning simplex embeddings, and then identifying the likelihood each simplex was heterophilous or homophilous. This work leveraged the probability assigned to each simplex to separate vertex classes with persistent homology through a persistence filtration function defined over simplex values to obtain a multilevel resolution sequence of graphs well suited for hierarchical GNNs. Taken as a node classification problem in graphs, this work began by formulating the learning problem over graph edges, integrating topological summaries, and learning multilevel subgraph hierarchies based on persistent filtration dependent on class connectivity. This approach offered an ordered sequence of graphs that captured both local and global structural information while also addressing critical issues such as oversmoothing, reducing computational complexity, and providing a paradigm for GNNs more resilient to heterophily.

This dissertation demonstrated that providing a more nuanced understanding of graph topology and leveraging hierarchical learning provided an approach that could lead to more resilient, computationally efficient, and expressive GNNs. Empirically, we showed that the introduced approach to persistence filtration learning and hierarchical graph learning compared favorably to previous learning techniques, including standard learning models, standard GNNs, and heterophily-specific models. This work achieved competitive performance in accuracy and complexity over established benchmark datasets for evaluating GNN performance in low and high heterophily scenarios. This dissertation showcased the transformative potential of integrating topological elements into machine learning, ultimately advancing the field towards more robust, efficient, and scalable models.

REFERENCES

[1] A. Klibisz, D. Rose, M. Eicholtz, J. Blundon, and S. Zakharenko, "Fast, simple calcium imaging segmentation with fully convolutional networks," Deep Learning in Medical Image Analysis and Multimodal Learning for Clinical Decision Support, Springer, 2017, pp. 285–293.

[2] M. Pachitariu, A. M. Packer, N. Pettit, H. Dalgleish, M. Hausser, and M. Sahani, "Extracting regions of interest from biological images with convolutional sparse block coding," *Advances in Neural Information Processing Systems*, vol. 26, 2013.

[3] V. Pascucci, X. Tricoche, H. Hagen, and J. Tierny, Topological Methods in Data Analysis and Visualization: Theory, Algorithms, and Applications. Springer Science & Business Media, 2010.

[4] P.-T. Bremer, W. Cabot, A. Cook, *et al.*, Understanding the structure of the turbulent mixing layer in hydrodynamic instabilities, Journal of Physics: Conference Series, IOP Publishing, vol. 46, 2006, p. 077.

[5] P.-T. Bremer, G. Weber, J. Tierny, V. Pascucci, M. Day, and J. Bell, "Interactive exploration and analysis of large-scale simulations using topology-based data segmentation," *IEEE Transactions on Visualization and Computer Graphics*, vol. 17, no. 9, pp. 1307–1324, 2010.

[6] H. Xue and Y. Gu, "Application of topological analysis in ocean feature extraction," *Computer Engineering*, vol. 35, no. 3, 263, p. 263, 2009. DOI: 10.3969/j.issn.1000-3428.2009.03.090. [Online]. Available: http://www.ecice06.com/EN/abstract/article_11349.shtml.

[7] T. McDonald, W. Usher, N. Morrical, *et al.*, "Improving the usability of virtual reality neuron tracing with topological elements," *IEEE Transactions on Visualization and Computer Graphics*, vol. 27, no. 2, pp. 744–754, 2020.

[8] S. Petruzza, A. Gyulassy, S. Leventhal, *et al.*, "High-throughput feature extraction for measuring attributes of deforming open-cell foams," *IEEE Transactions on Visualization and Computer Graphics*, vol. 26, no. 1, pp. 140–150, 2019.

[9] H. Bhatia, A. G. Gyulassy, V. Lordi, J. E. Pask, V. Pascucci, and P.-T. Bremer, "Topoms: Comprehensive topological exploration for molecular and condensed-matter systems," *Journal of Computational Chemistry*, vol. 39, no. 16, pp. 936–952, 2018.

[10] A. Venkat, A. Gyulassy, G. Kosiba, *et al.*, "Towards replacing physical testing of granular materials with a topology-based model," *IEEE Transactions on Visualization and Computer Graphics*, vol. 28, no. 1, pp. 76–85, 2021.

[11] C.-T. Shih, N.-Y. Chen, T.-Y. Wang, *et al.*, "Neuroretriever: Automatic neuron segmentation for connectome assembly," *Frontiers in Systems Neuroscience*, p. 67, 2021.

[12] J. Mayer, A. Robert-Moreno, J. Sharpe, and J. Swoger, "Attenuation artifacts in light sheet fluorescence microscopy corrected by optispim," *Light: Science & Applications*, vol. 7, no. 1, pp. 1–13, 2018.

[13] P. Ricci, V. Gavryusev, C. Müllenbroich, *et al.*, "Removing striping artifacts in light-sheet fluorescence microscopy: A review," *Progress in Biophysics and Molecular Biology*, 2021.

[14] O. Ronneberger, P. Fischer, and T. Brox, U-Net: Convolutional networks for biomedical image segmentation, International Conference on Medical Image Computing and Computer-Assisted Intervention, Springer, 2015, pp. 234–241.

[15] S. Banerjee, L. Magee, D. Wang, *et al.*, "Semantic segmentation of microscopic neuroanatomical data by combining topological priors with encoder–decoder deep networks," *Nature Machine Intelligence*, vol. 2, no. 10, pp. 585–594, 2020.

[16] K. Xu, W. Hu, J. Leskovec, and S. Jegelka, "How powerful are graph neural networks?" *arXiv preprint arXiv:1810.00826*, 2018.

[17] R. Forman, "A user's guide to discrete Morse theory.," *Séminaire Lotharingien de Combinatoire [electronic only]*, vol. 48, B48c–35, 2002.

[18] A. Gyulassy, P.-T. Bremer, and V. Pascucci, "Shared-memory parallel computation of morse-smale complexes with improved accuracy," *IEEE Transactions on Visualization and Computer Graphics*, 2018, MSCEER: Morse-Smale complex extraction, exploration, reasoning. Software available at https://github.com/sci-visus/MSCEER. DOI: 10.1109/TVCG.2018.2864848. [Online]. Available: http://ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=8440824&isnumber=4359476.

[19] V. Robins, P. J. Wood, and A. P. Sheppard, "Theory and algorithms for constructing discrete Morse complexes from grayscale digital images," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 33, no. 8, pp. 1646–1658, 2011. DOI: 10.1109/TPAMI.2011.95.

[20] A. Gyulassy, P.-T. Bremer, B. Hamann, and V. Pascucci, "A practical approach to Morse-Smale complex computation: Scalability and generality," *IEEE Transactions on Visualization and Computer Graphics*, vol. 14, no. 6, pp. 1619–1626, 2008.

[21] A. Gyulassy, T. Bremer, and V. Pascucci, "Shared-memory parallel computation of morse-smale complexes with improved accuracy," *IEEE Transactions on Visualization and Computer Graphics*, vol. 25, pp. 1183–1192, 2019.

[22] A. Gyulassy, V. Natarajan, V. Pascucci, P.-T. Bremer, and B. Hamann, "A topological approach to simplification of three-dimensional scalar functions," *IEEE Transactions on Visualization and Computer Graphics*, vol. 12, no. 4, pp. 474–484, 2006.

[23] H. Edelsbrunner, D. Letscher, and A. Zomorodian, Topological persistence and simplification, Proceedings 41st Annual Symposium on Foundations of Computer Science, IEEE, 2000, pp. 454–463.

[24] J. Tierny, G. Favelier, J. A. Levine, C. Gueunet, and M. Michaux, "The topology toolkit," *IEEE Transactions on Visualization and Computer Graphics*, vol. 24, no. 1, pp. 832–842, 2017.

[25] L. Acciai, P. Soda, and G. Iannello, "Automated neuron tracing methods: An updated account," *Neuroinformatics*, vol. 14, no. 4, pp. 353–367, 2016.

[26] P.-T. Bremer, H. Edelsbrunner, B. Hamann, and V. Pascucci, A multi-resolution data structure for two-dimensional Morse-Smale functions, IEEE Visualization, 2003. VIS 2003., 2003, pp. 139–146. DOI: 10.1109/VISUAL.2003.1250365.

[27] N. R. Pal and S. K. Pal, "A review on image segmentation techniques," *Pattern Recognition*, vol. 26, no. 9, pp. 1277–1294, 1993.

[28] D. L. Pham, C. Xu, and J. L. Prince, "Current methods in medical image segmentation," *Annual Review of Biomedical Engineering*, vol. 2, no. 1, pp. 315–337, 2000.

[29] D. Liu, Y. Xiong, K. Pulli, and L. Shapiro, Estimating image segmentation difficulty, International Workshop on Machine Learning and Data Mining in Pattern Recognition, Springer, 2011, pp. 484–495.

[30] R. Achanta, A. Shaji, K. Smith, A. Lucchi, P. Fua, and S. Süsstrunk, "SLIC superpixels," Tech. Rep., 2010.

[31] R. Achanta, A. Shaji, K. Smith, A. Lucchi, P. Fua, and S. Süsstrunk, "SLIC superpixels compared to state-of-the-art superpixel methods," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 34, no. 11, pp. 2274–2282, 2012.

[32] K. Konyushkova, R. Sznitman, and P. Fua, Introducing geometry in active learning for image segmentation, Proceedings of the IEEE International Conference on Computer Vision, 2015, pp. 2974–2982.

[33] S. Eslami, N. Heess, C. K. Williams, and J. Winn, "The shape boltzmann machine: A strong model of object shape," *International Journal of Computer Vision*, vol. 107, no. 2, pp. 155–176, 2014.

[34] F. Chen, H. Yu, R. Hu, and X. Zeng, Deep learning shape priors for object segmentation, Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2013, pp. 1870–1877.

[35] P. P. Acharjya and D. Ghoshal, "Watershed segmentation based on distance transform and edge detection techniques," *International Journal of Computer Applications*, vol. 52, no. 13, 2012.

[36] S. Beucher, Watersheds of functions and picture segmentation, International Conference on Acoustics, Speech, and Signal Processing, IEEE, vol. 7, 1982, pp. 1928–1931.

[37] J. B. Roerdink and A. Meijster, "The watershed transform: Definitions, algorithms and parallelization strategies," *Fundamenta Informaticae*, vol. 41, no. 1-2, pp. 187–228, 2000.

[38] M. Sonka, V. Hlavac, and R. Boyle, Image Processing, Analysis, and Machine Vision. Cengage Learning, 2014.

[39] P. Bendich, D. Cohen-Steiner, H. Edelsbrunner, J. Harer, and D. Morozov, Inferring local homology from sampled stratified spaces, 48th Annual IEEE Symposium on Foundations of Computer Science, IEEE, 2007, pp. 536–546.

[40] H. Edelsbrunner and J. Harer, Computational Topology: An Introduction. American Mathematical Soc., 2010.

[41] S. Svensson, I. Nyström, and G. Sanniti di Baja, "Curve skeletonization of surface-like objects in 3d images guided voxel classification," *Pattern Recognition Letters*, vol. 23, pp. 1419–1426, Oct. 2002. DOI: 10.1016/S0167-8655(02)00102-2.

[42] G. Borgefors, I. Nyström, and G. Sanniti di Baja, "Computing skeletons in three dimensions," *Pattern Recognition*, vol. 32, pp. 1225–1236, Jul. 1999. DOI: 10.1016/S0031-3203(98)00082-X.

[43] I. Bitter, A. E. Kaufman, and M. Sato, "Penalized-distance volumetric skeleton algorithm,"

IEEE Transactions on Visualization and Computer Graphics, vol. 7, no. 3, pp. 195–206, 2001.

[44] S. Bouix, K. Siddiqi, and A. Tannenbaum, Flux driven fly throughs, 2003 IEEE Computer Society Conference on Computer Vision and Pattern Recognition, 2003. Proceedings., IEEE, vol. 1, 2003, pp. I–I.

[45] N. D. Cornea, D. Silver, X. Yuan, and R. Balasubramanian, "Computing hierarchical curve-skeletons of 3d objects," *The Visual Computer*, vol. 21, no. 11, pp. 945–955, 2005.

[46] Y. Zhou and A. W. Toga, "Efficient skeletonization of volumetric objects," *IEEE Transactions on Visualization and Computer Graphics*, vol. 5, no. 3, pp. 196–209, 1999.

[47] M. Sato, I. Bitter, M. A. Bender, A. E. Kaufman, and M. Nakajima, Teasar: Tree-structure extraction algorithm for accurate and robust skeletons, Proceedings the Eighth Pacific Conference on Computer Graphics and Applications, Oct. 2000, pp. 281–449. DOI: 10.1109/PCCGA.2000.883951.

[48] M. S. Hassouna and A. A. Farag, Robust centerline extraction framework using level sets, 2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'05), IEEE, vol. 1, 2005, pp. 458–465.

[49] T. K. Dey and J. Sun, Defining and computing curve-skeletons with medial geodesic function, Symposium on Geometry Processing, A. Sheffer and K. Polthier, Eds., The Eurographics Association, 2006, ISBN: 3-905673-24-X. DOI: 10.2312/SGP/SGP06/143-152.

[50] H. Blum, "A transformation for extracting new descriptors of shape," Models for the Perception of Speech and Visual Form, W. W. Dunn, Ed., MIT Press, 1967, pp. 362–381.

[51] C. Lantuéjoul and S. Beucher, "On the use of the geodesic metric in image analysis," *Journal of Microscopy*, vol. 121, no. 1, pp. 39–49, 1981.

[52] G. Bertrand, "A parallel thinning algorithm for medial surfaces," *Pattern Recognition Letters*, vol. 16, no. 9, pp. 979–986, Sep. 1995, ISSN: 0167-8655. DOI: 10.1016/0167-8655(95)00034-E. [Online]. Available: http://dx.doi.org/10.1016/0167-8655(95)00034-E.

[53] G. Bertrand and Z. Aktouf, Three-dimensional thinning algorithm using subfields, Vision Geometry III, vol. 2356, 1995. DOI: 10.1117/12.198601. [Online]. Available: https://doi.org/10.1117/12.198601.

[54] G. Borgefors, I. Nyström, and G. Sanniti di Baja, Surface skeletonization of volume objects, International Workshop on Structural and Syntactic Pattern Recognition, Springer, Aug. 1996, pp. 251–259. DOI: 10.1007/3-540-61577-6_26.

[55] P. K. Saha and B. B. Chaudhuri, "Detection of 3-d simple points for topology preserving transformations with application to thinning," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 16, no. 10, pp. 1028–1032, Oct. 1994, ISSN: 0162-8828. DOI: 10.1109/34.329007.

[56] K. Palágyi, E. Sorantin, E. Balogh, *et al.*, A sequential 3d thinning algorithm and its medical applications, 17th International Conference on Information Processing in Medical Imaging, 2001.

[57] C. Lantuéjoul and F. Maisonneuve, "Geodesic methods in quantitative image analysis,"

Pattern Recognition, vol. 17, no. 2, pp. 177–187, 1984.

[58] J. A. Sethian, Level Set Methods and Fast Marching Methods: Evolving Interfaces in Computational Geometry, Fluid Mechanics, Computer Vision, and Materials Science. Cambridge university press, 1999, vol. 3.

[59] R. Cárdenes, S. K. Warfield, E. Macias, and J. Ruiz-Alzola, Occlusion points propagation geodesic distance transformation, Proceedings 2003 International Conference on Image Processing (Cat. No. 03CH37429), IEEE, vol. 1, 2003, pp. I–361.

[60] F. Benmansour, E. Turetken, and P. Fua, "Tubular geodesics using oriented flux: An itk implementation," *The Insight Journal*, vol. 6, pp. 27–29, 2013.

[61] F. Meyer, "Mathematical morphology: From two dimensions to three dimensions," *Journal of Microscopy*, vol. 165, no. 1, pp. 5–28, 1992.

[62] S. Beucher, "Digital skeletons in euclidean and geodesic spaces," *Signal Processing*, vol. 38, no. 1, pp. 127–141, 1994.

[63] P.-T. Bremer, G. Weber, V. Pascucci, M. Day, and J. Bell, "Analyzing and tracking burning structures in lean premixed hydrogen flames," *IEEE Transactions on Visualization and Computer Graphics*, vol. 16, no. 2, pp. 248–260, 2010.

[64] D. Ushizima, D. Morozov, G. H. Weber, A. G. C. Bianchi, J. A. Sethian, and E. W. Bethel, "Augmented topological descriptors of pore networks for material science," *IEEE Transactions on Visualization and Computer Graphics*, vol. 18, no. 12, pp. 2041–2050, Dec. 2012, ISSN: 1077-2626. DOI: 10.1109/TVCG.2012.200.

[65] T. Sousbie, "The persistent cosmic web and its filamentary structure - I. theory and implementation," *Monthly Notices of the Royal Astronomical Society*, vol. 414, no. 1, pp. 350–383, Jun. 2011.

[66] H. Bhatia, A. G. Gyulassy, V. Lordi, J. E. Pask, V. Pascucci, and P.-T. Bremer, "Topoms: Comprehensive topological exploration for molecular and condensed-matter systems," *Journal of Computational Chemistry*, vol. 0, no. 0, 2018. DOI: 10.1002/jcc.25181. eprint: https://onlinelibrary.wiley.com/doi/pdf/10.1002/jcc.25181. [Online]. Available: https://onlinelibrary.wiley.com/doi/abs/10.1002/jcc.25181.

[67] A. Gyulassy, A. Knoll, K. C. Lau, *et al.*, "Interstitial and interlayer ion diffusion geometry extraction in graphitic nanosphere battery materials," *IEEE Transactions on Visualization and Computer Graphics*, vol. 22, no. 1, pp. 916–925, Jan. 2016, ISSN: 1077-2626. DOI: 10.1109/TVCG.2015.2467432.

[68] A. Gyulassy, M. Duchaineau, V. Natarajan, *et al.*, "Topologically clean distance fields," *IEEE Transactions on Computer Graphics and Visualization*, vol. 13, no. 6, pp. 1432–1439, 2007.

[69] H. Edelsbrunner, D. Letscher, and A. Zomorodian, Topological persistence and simplification, Proceedings of the 41st Annual Symposium on Foundations of Computer Science, ser. FOCS '00, Washington, DC, USA: IEEE Computer Society, 2000, pp. 454–, ISBN: 0-7695-0850-2. [Online]. Available: http://dl.acm.org/citation.cfm?id=795666.796607.

[70] D. Silver and X. Wang, "Tracking and visualizing turbulent 3d features," *IEEE Transactions on Visualization and Computer Graphics*, vol. 3, no. 2, pp. 129–141, Apr. 1997, ISSN: 1077-2626. DOI: 10.1109/2945.597796.

[71] D. Silver and X. Wang, Tracking scalar features in unstructured data sets, Visualization '98. Proceedings, Oct. 1998, pp. 79–86. DOI: 10.1109/VISUAL.1998.745288.

[72] C. Bajaj, A. Shamir, and B.-S. Sohn, Progressive tracking of isosurfaces in time-varying scalar fields, 2002.

[73] H. Edelsbrunner, J. Harer, A. Mascarenhas, and V. Pascucci, Time-varying reeb graphs for continuous space-time data, Proceedings of the 20th Annual Symposium on Computational Geometry, Brooklyn, New York, USA: ACM, 2004, pp. 366–372. DOI: 10.1145/997817.997872. [Online]. Available: https://doi.org/10.1145/997817.997872.

[74] G. Weber, P.-T. Bremer, M. Day, J. Bell, and V. Pascucci, "Feature tracking using reeb graphs," Topological Methods in Data Analysis and Visualization, ser. Mathematics and Visualization, Springer, 2011, pp. 241–253. DOI: 10.1007/978-3-642-15014-2_20.

[75] W. Widanagamaachchi, J. Chen, P. Klacansky, *et al.*, Tracking features in embedded surfaces: Understanding extinction in turbulent combustion, 2015 IEEE 5th Symposium on Large Data Analysis and Visualization (LDAV), Oct. 2015, pp. 9–16. DOI: 10.1109/LDAV.2015.7348066.

[76] H. Sundar, D. Silver, N. Gagvani, and S. Dickinson, Skeleton based shape matching and retrieval, 2003 Shape Modeling International., IEEE, 2003, pp. 130–139.

[77] A. Brennecke and T. Isenberg, 3d shape matching using skeleton graphs. SimVis, Jan. 2004, pp. 299–310.

[78] N. Gagvani and D. Silver, "Parameter controlled skeletonization of three dimensional objects," *Department of Electrical and Computational Engineering, Rutgers University, Piscataway, NJ, Tech. Rep. CAIP-TR-216*, 1997.

[79] F. Reinders, F. H. Post, and H. J. Spoelder, "Visualization of time-dependent data with feature tracking and event detection," *The Visual Computer*, vol. 17, no. 1, pp. 55–71, 2001.

[80] A. S. Lowet, C. Firestone, and B. J. Scholl, "Seeing structure: Shape skeletons modulate perceived similarity," *Attention, Perception, & Psychophysics*, vol. 80, no. 5, pp. 1278–1289, 2018.

[81] C. Yang, O. Tiebe, K. Shirahama, and M. Grzegorzek, "Object matching with hierarchical skeletons," *Pattern Recognition*, vol. 55, pp. 183–197, 2016.

[82] F. Rosenblatt, "The perceptron: A probabilistic model for information storage and organization in the brain.," *Psychological Review*, vol. 65, no. 6, p. 386, 1958.

[83] T. N. Kipf and M. Welling, Semi-supervised classification with graph convolutional networks, International Conference on Learning Representations (ICLR), 2017.

[84] W. L. Hamilton, R. Ying, and J. Leskovec, Inductive representation learning on large graphs, Proceedings of the 31st International Conference on Neural Information Processing Systems, 2017, pp. 1025–1035.

[85] T. K. Dey, J. Wang, and Y. Wang, Road network reconstruction from satellite images with machine learning supported by topological methods, Proceedings of the 27th ACM SIGSPATIAL International Conference on Advances in Geographic Information Systems, 2019, pp. 520–523.

[86] X. Hu, Y. Wang, L. Fuxin, D. Samaras, and C. Chen, "Topology-aware segmentation using discrete Morse theory," *arXiv preprint arXiv:2103.09992*, 2021.

[87] M. Moor, M. Horn, B. Rieck, and K. Borgwardt, Topological autoencoders, International Conference on Machine Learning, PMLR, 2020, pp. 7045–7054.

[88] Q. Zhao, Z. Ye, C. Chen, and Y. Wang, Persistence enhanced graph neural network, International Conference on Artificial Intelligence and Statistics, PMLR, 2020, pp. 2896–2906.

[89] F. Hensel, M. Moor, and B. Rieck, "A survey of topological machine learning methods," *Frontiers in Artificial Intelligence*, vol. 4, p. 681 108, 2021.

[90] Y. Umeda, "Time series classification via topological data analysis," *Information and Media Technologies*, vol. 12, pp. 228–239, 2017.

[91] B. Rieck, F. Sadlo, and H. Leitte, Topological machine learning with persistence indicator functions, Topological Methods in Data Analysis and Visualization V, H. Carr, I. Fujishiro, F. Sadlo, and S. Takahashi, Eds., Springer, 2020, pp. 87–101.

[92] M. Carrière, S. Oudot, and M. Ovsjanikov, "Stable topological signatures for points on 3d shapes," *Computer Graphics Forum*, vol. 34, pp. 1–12, 2015. DOI: 10.1111/cgf.12692.

[93] P. Bubenik, "Statistical topological data analysis using persistence landscapes," *Journal of Machine Learning Research*, vol. 16, no. 3, pp. 77–102, 2015.

[94] H. Adams, T. Emerson, M. Kirby, *et al.*, "Persistence images: A stable vector representation of persistent homology," *Journal of Machine Learning Research*, vol. 18, no. 1, pp. 218–252, 2017.

[95] I. Chevyrev, V. Nanda, and H. Oberhauser, "Persistence paths and signature features in topological data analysis," *IEEE Transactions Pattern Analysis Machine Intelligence*, vol. 42, pp. 192–202, 2018. DOI: 10.1109/TPAMI.2018.2885516.

[96] K. Kim, J. Kim, M. Zaheer, J. Kim, F. Chazal, and L. Wasserman, Pllay: Efficient topological layer based on persistent landscapes, Advances in Neural Information Processing Systems, H. Larochelle, M. Ranzato, R. Hadsell, M. F. Balcan, and H. Lin, Eds., vol. 33, Curran Associates, Inc., 2020, pp. 15 965–15 977.

[97] Q. Zhao and Y. Wang, Learning metrics for persistence-based summaries and applications for graph classification, Advances in Neural Information Processing Systems, H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. Fox, and R. Garnett, Eds., vol. 32, Curran Associates, Inc., 2019.

[98] M. Carrière, M. Cuturi, and S. Oudot, Sliced wasserstein kernel for persistence diagrams, Proceedings of the 34th International Conference on Machine Learning, D. Precup and Y. W. The, Eds., PMLR, 2017, pp. 664–673.

[99] G. Kusano, K. Fukumizu, and Y. Hiraoka, "Kernel method for persistence diagrams via kernel embedding and weight factor," *Journal of Machine Learning Research*, vol. 18, pp. 1–41, 2018.

[100] S. Kolouri, Y. Zou, and G. K. Rohde, Sliced wasserstein kernels for probability distributions, IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2016, pp. 5258–5267. DOI: 10.1109/CVPR.2016.568.

[101] C. Hofer, R. Kwitt, M. Niethammer, and A. Uhl, Deep learning with topological signatures, Advances in Neural Information Processing Systems, I. Guyon, U. V. Luxburg, S. Bengio, *et al.*, Eds., vol. 30, Curran Associates, Inc., 2017.

[102] M. Carrière, F. Chazal, Y. Ike, T. Lacombe, M. Royer, and Y. Umeda, Perslay: A neural network layer for persistence diagrams and new graph topological signatures, Proceedings of the Twenty Third International Conference on Artificial Intelligence and Statistics, PMLR, vol. 108, 2020, pp. 2786–2796.

[103] M. Moor, M. Horn, B. Rieck, and K. Borgwardt, Topological autoencoders, Proceedings of the 37th International Conference on Machine Learning, H. Daumé III and A. Singh, Eds., PMLR, 2020, pp. 7045–7054.

[104] C. Chen, X. Ni, Q. Bai, and Y. Wang, A topological regularizer for classifiers via persistent homology, Proceedings of Machine Learning Research, K. Chaudhuri and M. Sugiyama, Eds., 2019, pp. 2573–2582.

[105] Q. Zhao, Z. Ye, C. Chen, and Y. Wang, Persistence enhanced graph neural network, Proceedings of the Twenty Third International Conference on Artificial Intelligence and Statistics, S. Chiappa and R. Calandra, Eds., PMLR, 2020, pp. 2896–2906.

[106] I. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, *et al.*, Generative adversarial nets, Advances in Neural Information Processing Systems, Z. Ghahramani, M. Welling, C. Cortes, N. D. Lawrence, and K. Q. Weinberger, Eds., Curran Associates, Inc., vol. 27, 2014.

[107] C. Fefferman, S. Mitter, and H. Narayanan, "Testing the manifold hypothesis," *Journal of the American Mathematical Society*, vol. 29, pp. 983–1049, 2013. DOI: 10.1090/jams/852.

[108] B. Rieck, M. Togninalli, C. Bock, M. Moor, M. Horn, T. Gumbsch, *et al.*, Neural persistence: A complexity measure for deep neural networks using algebraic topology, International Conference on Learning Representations, 2019.

[109] K. N. Ramamurthy, K. Varshney, and K. Mody, Topological data analysis of decision boundaries with application to model selection, Proceedings of the 36th International Conference on Machine Learning, K. Chaudhuri and R. Salakhutdinov, Eds., PMLR, 2019, pp. 5351–5360.

[110] R. B. Gabrielsson and G. Carlsson, Exposition and interpretation of the topology of neural networks, 2019 18th IEEE International Conference on Machine Learning And Applications (ICMLA), 2019, pp. 1069–1076. DOI: 10.1109/ICMLA.2019.00180.

[111] J. Curry, S. Mukherjee, and K. Turner, "How many directions determine a shape and other sufficiency results for two topological transforms," *arXiv preprint arXiv:1805.09782*, 2018.

[112] C. Hofer, R. Kwitt, M. Niethammer, and M. Dixit, Connectivity-optimized representation learning via persistent homology, Proceedings of the 36th International Conference on Machine Learning, K. Chaudhuri and R. Salakhutdinov, Eds., PMLR, 2019, pp. 2751–2760.

[113] Q. Zhao and Y. Wang, Learning metrics for persistence-based summaries and applications for graph classification, Advances in Neural Information Processing Systems, H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. Fox, and R. Garnett, Eds., vol. 32, Curran Associates, Inc., 2019.

[114] B. Rieck, M. Togninalli, C. Bock, M. Moor, M. Horn, T. Gumbsch, *et al.*, Neural persistence: A complexity measure for deep neural networks using algebraic topology, International Conference on Learning Representations, 2019.

[115] The GUDHI Project, GUDHI User and reference manual, 3.1.1. GUDHI Editorial Board, 2020. [Online]. Available: https://gudhi.inria.fr/doc/3.1.1/.

[116] G. Tauzin, U. Lupo, L. Tunstall, J. B. Perez, M. Caorsi, A. M. Medina-Mardones, *et al.*, "Giotto-tda: A topological data analysis toolkit for machine learning and data exploration," *Journal of Machine Learning Research*, vol. 22, pp. 1–6, 2021.

[117] S. Leventhal, A. Gyulassy, V. Pascucci, and M. Heimann, Modeling hierarchical topological structure in scientific images with graph neural networks, NeurIPS 2022 Workshop: New Frontiers in Graph Learning, 2022.

[118] S. Leventhal, A. Gyulassy, M. Heimann, and V. Pascucci, "Exploring classification of topological priors with machine learning for feature extraction," *IEEE Transactions on Visualization and Computer Graphics*, 2023.

[119] T. Lindeberg, "Scale-space theory: A basic tool for analyzing structures at different scales," *Journal of Applied Statistics*, vol. 21, no. 1-2, pp. 225–270, 1994.

[120] S. van der Walt, J. L. Schönberger, J. Nunez-Iglesias, *et al.*, "Scikit-image: Image processing in Python," *PeerJ*, vol. 2, e453, Jun. 2014, ISSN: 2167-8359. DOI: 10.7717/peerj.453. [Online]. Available: https://doi.org/10.7717/peerj.453.

[121] N. Kanopoulos, N. Vasanthavada, and R. L. Baker, "Design of an image edge detection filter using the Sobel operator," *IEEE Journal of Solid-State Circuits*, vol. 23, no. 2, pp. 358–367, 1988.

[122] P. Virtanen, R. Gommers, T. E. Oliphant, *et al.*, "SciPy 1.0: Fundamental Algorithms for Scientific Computing in Python," *Nature Methods*, vol. 17, pp. 261–272, 2020. DOI: 10.1038/s41592-019-0686-2.

[123] A. F. Frangi, W. J. Niessen, K. L. Vincken, and M. A. Viergever, Multiscale vessel enhancement filtering, International Conference on Medical Image Computing and Computer-Assisted Intervention, Springer, 1998, pp. 130–137.

[124] M. Sweet, C. P. Earls, M. Melcher, and B. Spitzak, "Fltk 1.1. 10 programming manual," *FLTK) Revision*, vol. 10, 1998.

[125] K. Xu, C. Li, Y. Tian, T. Sonobe, K. Kawarabayashi, and S. Jegelka, Representation learning on graphs with jumping knowledge networks, Proceedings of the 35th International Conference on Machine Learning, ICML, vol. 80, PMLR, 2018, pp. 5449–5458.

[126] J. Zhu, Y. Yan, L. Zhao, M. Heimann, L. Akoglu, and D. Koutra, "Beyond homophily in graph neural networks: Current limitations and effective designs," *Advances in Neural Information Processing Systems*, vol. 33, pp. 7793–7804, 2020.

[127] H. Ramadan, C. Lachqar, and H. Tairi, "A survey of recent interactive image segmentation methods," *Computational Visual Media*, vol. 6, no. 4, pp. 355–384, 2020.

[128] S. D. Olabarriaga and A. W. Smeulders, "Interaction in the segmentation of medical images: A survey," *Medical Image Analysis*, vol. 5, no. 2, pp. 127–142, 2001.

[129] R. Hebbalaguppe, K. McGuinness, J. Kuklyte, G. Healy, N. O'Connor, and A. Smeaton, How interaction methods affect image segmentation: User experience in the task, 2013 1st IEEE Workshop on User-Centered Computer Vision (UCCV), IEEE, 2013, pp. 19–24.

[130] L. Breiman, "Random Forests," *Machine learning*, vol. 45, no. 1, pp. 5–32, 2001.

[131] E. Van der Merwe and S. Kidson, "Advances in imaging the blood and aqueous vessels of the ocular limbus," *Experimental Eye Research*, vol. 91, no. 2, pp. 118–126, 2010.

[132] D. B. Wilburn and W. J. Swanson, "From molecules to mating: Rapid evolution and

biochemical studies of reproductive proteins," *Journal of Proteomics*, vol. 135, pp. 12–25, 2016.

[133] T. A. Gillette, K. M. Brown, and G. A. Ascoli, "The DIADEM metric: Comparing multiple reconstructions of the same neuron," *Neuroinformatics*, vol. 9, no. 2, pp. 233–245, 2011.

[134] F. Pedregosa, G. Varoquaux, A. Gramfort, *et al.*, "Scikit-learn: Machine learning in Python," *the Journal of Machine Learning Research*, vol. 12, pp. 2825–2830, 2011.

[135] C. Pape, A. Matskevych, A. Wolny, *et al.*, "Leveraging domain knowledge to improve microscopy image segmentation with lifted multicuts," *Frontiers in Computer Science*, p. 6, 2019.

[136] J. Funke, F. Tschopp, W. Grisaitis, *et al.*, "Large scale image segmentation with structured loss based deep learning for connectome reconstruction," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 41, no. 7, pp. 1669–1680, 2018.

[137] A. Eberle, S. Mikula, R. Schalek, J. Lichtman, M. K. Tate, and D. Zeidler, "High-resolution, high-throughput imaging with a multibeam scanning electron microscope," *Journal of Microscopy*, vol. 259, no. 2, pp. 114–120, 2015.

[138] M. Januszewski, J. Kornfeld, P. H. Li, *et al.*, "High-precision automated reconstruction of neurons with flood-filling networks," *Nature Methods*, vol. 15, no. 8, pp. 605–610, 2018.

[139] M. Horn, E. D. Brouwer, M. Moor, Y. Moreau, B. Rieck, and K. Borgwardt, Topological graph neural networks, International Conference on Learning Representations, 2022.

[140] C. Deng, Z. Zhao, Y. Wang, Z. Zhang, and Z. Feng, Graphzoom: A multi-level spectral approach for accurate and scalable graph embedding, International Conference on Learning Representations, 2020.

[141] Z. Huang, S. Zhang, C. Xi, T. Liu, and M. Zhou, Scaling up graph neural networks via graph coarsening, Proceedings of the 27th ACM SIGKDD Conference on Knowledge Discovery & Data Mining, 2021.

[142] Z. Ying, J. You, C. Morris, X. Ren, W. Hamilton, and J. Leskovec, "Hierarchical graph representation learning with differentiable pooling," *Advances in Neural Information Processing Systems*, vol. 31, 2018.

[143] S. Leventhal, A. Gyulassy, M. Heimann, and V. Pascucci, "Exploring classification of topological priors with machine learning for feature extraction," *IEEE Transactions on Visualization and Computer Graphics*, 2023.

[144] Y. Lahini, O. Gottesman, A. Amir, and S. M. Rubinstein, "Nonmonotonic aging and memory retention in disordered mechanical systems," *Physical Review Letters*, vol. 118, no. 8, p. 085 501, 2017.

[145] T. A. Gillette, K. M. Brown, and G. A. Ascoli, "The diadem metric: Comparing multiple reconstructions of the same neuron," *Neuroinformatics*, vol. 9, pp. 233–245, 2011.

[146] M. Horn, E. De Brouwer, M. Moor, Y. Moreau, B. Rieck, and K. Borgwardt, Topological graph neural networks, https://arxiv.org/abs/2102.07835, 2021.

[147] C. Deng, Z. Zhao, Y. Wang, Z. Zhang, and Z. Feng, Graphzoom: A multi-level spectral approach for accurate and scalable graph embedding, International Conference on Learning Representations, 2020.

[148] Z. Huang, S. Zhang, C. Xi, T. Liu, and M. Zhou, Scaling up graph neural networks via graph

coarsening, Proceedings of the 27th ACM SIGKDD Conference on Knowledge Discovery & Data Mining, 2021, pp. 675–684.

[149] C. Morris, M. Ritzert, M. Fey, *et al.*, Weisfeiler and leman go neural: Higher-order graph neural networks, Proceedings of the AAAI Conference on Artificial Intelligence, vol. 33, 2019, pp. 4602–4609.

[150] B. Rieck, C. Bock, and K. Borgwardt, A persistent weisfeiler-lehman procedure for graph classification, International Conference on Machine Learning, PMLR, May 2019, pp. 5448–5458.

[151] F. Hensel, M. Moor, and B. Rieck, "A survey of topological machine learning methods," *Frontiers in Artificial Intelligence*, vol. 4, p. 681 108, 2021.

[152] C. Hofer, F. Graf, B. Rieck, M. Niethammer, and R. Kwitt, Graph filtration learning, Proceedings of the 37th International Conference on Machine Learning, H. Daumé III and A. Singh, Eds., PMLR, 2020, pp. 4314–4323.

[153] X. Han, T. Zhao, Y. Liu, X. Hu, and N. Shah, "Mlpinit: Embarrassingly simple gnn training acceleration with mlp initialization," *arXiv preprint arXiv:2210.00102*, 2022.

[154] J. Zhu, Y. Yan, L. Zhao, M. Heimann, L. Akoglu, and D. Koutra, "Beyond homophily in graph neural networks: Current limitations and effective designs," *Advances in Neural Information Processing Systems*, vol. 33, pp. 7793–7804, 2020.

[155] Z. Zhong, S. Ivanov, and J. Pang, "Simplifying node classification on heterophilous graphs with compatible label propagation," *arXiv preprint arXiv:2205.09389*, 2022.

[156] R. Murphy, B. Srinivasan, V. Rao, and B. Ribeiro, Relational pooling for graph representations, International Conference on Machine Learning, PMLR, 2019, pp. 4663–4673.

[157] R. Sato, M. Yamada, and H. Kashima, Random features strengthen graph neural networks, Proceedings of the 2021 SIAM International Conference on Data Mining (SDM), SIAM, 2021, pp. 333–341.

[158] Q. Li, Z. Han, and X.-M. Wu, Deeper insights into graph convolutional networks for semi-supervised learning, Proceedings of the AAAI Conference on Artificial Intelligence, vol. 32, 2018.

[159] C. Hofer, F. Graf, B. Rieck, M. Niethammer, and R. Kwitt, Graph filtration learning, International Conference on Machine Learning, PMLR, 2020, pp. 4314–4323.

[160] J. Zhu, Y. Yan, M. Heimann, L. Zhao, L. Akoglu, and D. Koutra, "Heterophily and graph neural networks: Past, present and future," *IEEE Data Engineering Bulletin*, 2023.

[161] S. Luan, C. Hua, Q. Lu, *et al.*, "The heterophilic graph learning handbook: Benchmarks, models, theoretical analysis, applications and challenges," *arXiv preprint arXiv:2407.09618*, 2024.

[162] H. Pei, B. Wei, K. Chang, Y. Lei, and B. Yang, Geom-gcn: Geometric graph convolutional networks, International Conference on Learning Representations, 2020.

[163] Y. Yan, M. Hashemi, K. Swersky, Y. Yang, and D. Koutra, Two sides of the same coin: Heterophily and oversmoothing in graph convolutional neural networks, 2022 IEEE International Conference on Data Mining (ICDM), IEEE, 2022, pp. 1287–1292.

[164] E. Rossi, B. Charpentier, F. Di Giovanni, F. Frasca, S. Günnemann, and M. M.

Bronstein, Edge directionality improves learning on heterophilic graphs, Learning on Graphs Conference, Proceedings of Machine Learning Research, 2023, pp. 25–1.

[165] J. Zhu, R. A. Rossi, A. Rao, *et al.*, Graph neural networks with heterophily, Proceedings of the AAAI Conference on Artificial Intelligence, vol. 35, 2021, pp. 11 168–11 176.

[166] W. Bi, L. Du, Q. Fu, Y. Wang, S. Han, and D. Zhang, "Make heterophilic graphs better fit gnn: A graph rewiring approach," *IEEE Transactions on Knowledge and Data Engineering*, 2024.

[167] S. Li, D. Kim, and Q. Wang, Restructuring graph for higher homophily via adaptive spectral clustering, Proceedings of the AAAI Conference on Artificial Intelligence, vol. 37, 2023, pp. 8622–8630.

[168] M. Huang, C. Cai, Y. Liang, J. Wang, and L. Song, Revisiting the role of heterophily in graph representation learning: An edge classification perspective, Advances in Neural Information Processing Systems, 2022.

[169] Y. Xue, Z. Jin, and W. Gao, "A data-centric graph neural network for node classification of heterophilic networks," *International Journal of Machine Learning and Cybernetics*, pp. 1–11, 2024.

[170] Q. Zhao, Z. Ye, C. Chen, and Y. Wang, Persistence enhanced graph neural network, International Conference on Artificial Intelligence and Statistics, PMLR, 2020, pp. 2896–2906.

[171] S. Leventhal, A. Gyulassy, V. Pascucci, and M. Heimann, Modeling hierarchical topological structure in scientific images with graph neural networks, 2023 IEEE International Conference on Image Processing (ICIP), IEEE, 2023, pp. 2995–2999.

[172] C. Bodnar, A. T. Vasilache, M. Rüttgers, *et al.*, "Neural sheaf diffusion: A topological perspective on heterophily and oversmoothing in gnns," *arXiv preprint arXiv:2202.04579*, 2022.

[173] C. Battiloro, I. Spinelli, L. Telyatnikov, M. Bronstein, S. Scardapane, and P. Di Lorenzo, "From latent graph to latent topology inference: Differentiable cell complex module," *arXiv preprint arXiv:2305.16174*, 2023.

[174] A. Paszke, S. Gross, F. Massa, *et al.*, Pytorch: An imperative style, high-performance deep learning library, Advances in Neural Information Processing Systems, vol. 32, 2019, pp. 8024–8035.

[175] M. Fey and J. E. Lenssen, Fast graph representation learning with pytorch geometric, ICLR Workshop on Representation Learning on Graphs and Manifolds, 2019. [Online]. Available: https://arxiv.org/abs/1903.02428.

[176] D. Morozov, Dionysus - a library for computing persistent homology, http://www.mrzv.org/software/dionysus/, Accessed: 2024-08-10, 2007–.

[177] Z. Yang, W. W. Cohen, and R. Salakhutdinov, Revisiting semi-supervised learning with graph embeddings, International Conference on Machine Learning, PMLR, 2016, pp. 40–48.

[178] B. Rozemberczki, R. Davies, R. Sarkar, and C. Sutton, Multi-scale attributed node embedding, Proceedings of the 27th ACM SIGKDD Conference on Knowledge Discovery & Data Mining, 2021, pp. 1235–1243.

[179] T. P. Peixoto and F. Emmert-Streib, Network cartography of university websites: A comparative analysis, Proceedings of the 30th International Conference on Computer Networks, 2019, pp. 1–11.