Exploring Classification of Topological Priors With Machine Learning for Feature Extraction

Samuel Leventhal, Attila Gyulassy, Mark Heimann, and Valerio Pascucci

***Abstract*— 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. In this article, we describe an approach to creating learnable topological elements, explore the application of ML techniques to classification tasks in a number of areas, and demonstrate this approach as a viable alternative to pixel-level classification, with similar accuracy, improved execution time, and requiring marginal training data.**

***Index Terms*—Computational topology, feature detection, graph learning, graph neural networks, machine learning, Morse-Smale complex, scientific visualization, segmentation, topological data analysis.**

I. 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 og 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 is first of its kind: no pre-existing 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 [33], [42]. Moreover, the need for obtaining or generating 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) [35], [45]. 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 [11], ocean eddies [30], neuron segmentation [39], ligaments in structural foams [47], and atomic structures [8]. 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 which enables higher level measurement and reasoning that answer scientific questions, such as bubble growth in turbulent mixing [35], the relationship between curvature and failure of foam struts [47], or estimating flow through porous materials [61].

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

Manuscript received 6 September 2022; revised 5 February 2023; accepted 8 February 2023. Date of publication 24 February 2023; date of current version 26 June 2024. This work was supported in part by NSF OAC under Grant 2138811, in part by NSF CI CoE under Grant 2127548, in part by Department of Energy (DoE) under Grant DE-FE0031880, in part by the Intel oneAPI Centers of Excellence at University of Utah, in part by the Exascale Computing Project under Grant 17-SC-20-SC, a collaborative effort of the DoE and the NNSA, in part by UT-Battelle, LLC under Grant DE-AC05-00OR22725, in part by the U.S. DoE by Lawrence Livermore National Laboratory under Grant DE-AC52-07NA27344, and in part by LDRD Program under Grant 21-ERD-012. Recommended for acceptance by V. Natarajan. *(Corresponding author: Samuel Leventhal.)*

Samuel Leventhal, Attila Gyulassy, and Valerio Pascucci are with the School of Computing and the Scientific Computing and Imaging Institute, University of Utah, Salt Lake City, UT 84112 USA (e-mail: samlev@cs.utah.edu; jediati@sci.utah.edu; pascucci@sci.utah.edu).

Mark Heimann is with the Lawrence Livermore National Laboratory, Livermore, CA 94550 USA (e-mail: heimann2@llnl.gov).

Digital Object Identifier 10.1109/TVCG.2023.3248632

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 [55], attenuation [38], refraction and absorption [50], 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 [39]. 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 [5], [53]. 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 present an approach for representing objects composed in images as a topologically informative 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 offered here are as follows:

- We demonstrate how topological priors introduce a new space to frame machine learning tasks, offering competitive segmentation accuracy, in both advanced and basic machine learning models, compared to standard or state-of-the-art pixel-based models.
- We design a novel and easily generalized framework to rephrase segmentation as a classification problem that learns to identify a subset graph constructed from topological priors of the MSC computed over an image.
- We present a fast labeling tool to assign class labels to components within the MSC, making manual ground truth labeling of images easier for users.
- We devise a new method for comparing topological prior-object-level predictions to pixel-level predictions and evaluate the performance of our approach across various domains, including medical, neuroscience, and materials science.
- We exhibit a generalized framework based on topological priors to be used by machine learning models, either classically or in an interactive setting, with quick labeling, training, and predicted segmentations as compared to contemporary, standard, or advanced pixel-based methods.

II. Background and Related Work

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.

A. Computational Topology

Topological abstractions have been applied across multiple domains, such as segmentation of neurons [39], structural components of interest in metallic foams [47], eddies in ocean currents [30], bubble formation in mixing fluids [45], and ignition kernels in combustion [35]. 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.

The Morse-Smale Complex: A Morse function $f : \mathcal{M} \to \mathbb{R}$ 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 $f(x) = f(b) \pm x_1^2 \pm \cdots \pm x_d^2$, with d being the dimension of $\mathcal{M}$. 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 $\mathcal{M}$ 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. Fig. 1(a) and (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.

Discrete Morse Theory: Concepts from continuous functions can be applied to a discrete pixel space following an approach based on discrete Morse theory [18]. Instead of a continuous manifold $\mathcal{M}$, 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

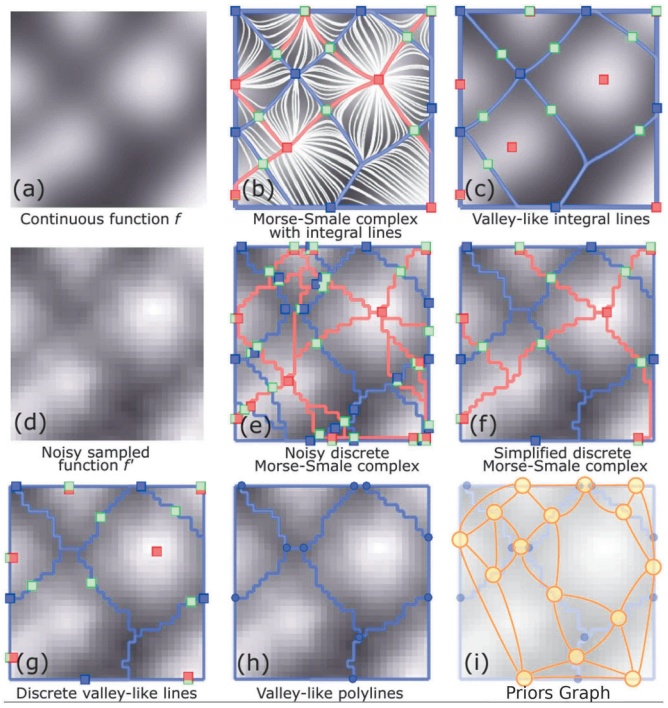


Fig. 1. Morse-Smale complexes are defined for functions with continuous gradients *a-c)*. A smooth function *a)* can be partitioned based on the behavior of integral lines *b)*, with selected integral lines shown in white. This partition forms a cell complex, where integral lines within each cell share 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.

V-paths, starting and ending at critical cells. In this study, we use the open-source MSCEER library [22], implementing steepest-descent [23], [51] and accurate-geometry [24] 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. Often in imaged data, the semantic target consists of higher intensity pixel values. 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 [25].

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* [16]. 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 [1], [58]; the critical cell pairs in the discrete gradient field can be canceled through path reversal [18]; or the MSC can be directly simplified by successively canceling nodes connected by a single arc in the 1-skeleton [10], [16], [25]. 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. Fig. 1(e) and (f) illustrates the use of simplification to remove excess nodes and arcs from the 1-skeleton of the MSC.

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 Fig. 1(g). In many applications, non-overlapping edges are desired, for example, to enable a mapping from image pixels to unique components in a more manageable graph structure. Mcdonald et al. [39] 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. Fig. 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.

B. Image Segmentation

Despite the value of image segmentation to many sciences, medical, and engineering disciplines, it remains a laborious and time-intensive task [37], [43], [48]. 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 [53]. 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 [2], [3]. 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 [34]. 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 [12], [17].

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 greyscale, applying a Sobel filter, and computing watershed regions [4], [7], [52]. Similarly, to apply the topological framework, persistent homology has been used to better understand root architecture from images, identify cells in microscopy images, and much more [6], [15], [56].

Computational Topology for Machine Learning: Several recent works have used topological methods, specifically in the task of segmentation. Banerjee et al. [5] applied the MSC as an image-level prior to be used in a modified U-Net [53] 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 [13].

TDA has also been incorporated into neural network architecture to train deep learning segmentation techniques to conform to higher topological accuracy. Hu et al. [28] 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 [40], [65]. Our work, in contrast to these methods, performs learning in the topological domain over priors, separate from the image space.

III. 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.

A. 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 better suited to the semantic objects, the ridge/valley graph. The edges of the ridge graph are realized by polylines, 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

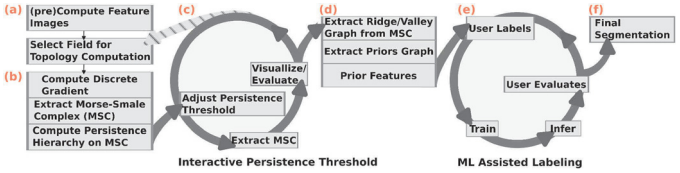


Fig. 2. Illustration of proposed workflow using topological priors with machine learning to accelerate segmentation. Beginning at (a), feature images are precomputed. Next, a topologically informative scalar field is provided by the user. Using the scalar field image, in stage (b), the discrete gradient 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).

specific encoding will likely depend on the application, and we describe an instance of the encoding in Section IV.

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 III-B of the paper.

B. 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 Fig. 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 Fig. 3(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.

IV. A Workflow for Learning and Comparison

We describe an interactive learning workflow that uses topological priors for fast and accurate segmentation. Following the pre-computation 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 Fig. 2.

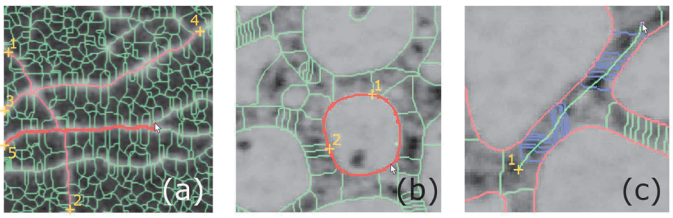


Fig. 3. Interactive tool allowing for human segmentation by labeling topological priors, specifically arcs, as opposed to individual pixels. With the shortest-path tool, a user is six clicks into labeling the foreground neurons (a). Only three clicks are needed to draw a closed loop (b). Finding objects crossing a free-form stroke allows rapid labeling (c).

A. 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.

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:* We convolve the image with a Gaussian function with standard deviation $\sigma \in \{2, 4, 8, 16\}$ as 4 features. Gaussian blurring has been shown to be a representative smoothing kernel for capturing pixel neighborhood information and minimizing fine scale structures while not introducing unrelated structures in higher scales [36], [60].

- *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 [31], [62] 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 [62] over the Gaussian blurred images $\mathbf{X}_r$ and included in the feature set contributing 7 feature attributes [19].

B. 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.

1) User Selection of Scalar Field for Topology: In order to ensure that the semantic object is entirely covered by topological elements, it is necessary to first 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 this 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 properly account for domain-specific noisy artifacts introduced during data acquisition. Our approach is to allow a user to select a scalar field image from the precomputed feature images. Once selected, as shown in Fig. 2, the system computes the discrete gradient and the MSC 1-skeleton, and builds a hierarchy. The results are displayed in an interactive viewer.

2) Simplification Threshold Selection Cycle: Using the precomputed MSC hierarchy, the first interactive cycle shown in Fig. 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 applied to 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 IV-B1).

3) 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 IV-A, 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 resulting in 235 features per topological prior.

C. 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.

1) Interactive Labeling Cycle: 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 [57]. Several interactions are supported for easy selection and "painting" foreground/background labels, as highlighted for various datasets in Fig. 3. 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 [39]. 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.

2) Shallow and Deep Learning: Topological Priors for ML Assisted Labeling: 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 IV-A, IV-B3, and IV-C1. 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.

Deep Learning: We use a conventional feedforward neural network, or multi-layer perceptron (MLP), and an inductive graph neural network, GraphSAGE [26], for node classification of the priors graph.

Multilayer Perceptrons are canonical feed-forward neural networks consisting of three fully connected layers of neurons, each with a nonlinear activation function, trained using backpropagation [54]. 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 becomes $L_{\text{CE}}(\mathbf{z}) = -y \cdot \sigma(\mathbf{z}^T \cdot \mathbf{W}) + log(1 + e^{\sigma(\mathbf{z}^T \cdot \mathbf{W})})$ for labels y.

Graph neural networks, 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 $N(v)$:

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

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 [32], GraphSAGE [26], and many others.

We use the popular GraphSAGE [26] 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 [64]: 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 [66].

V. 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 as shown in (e) of Fig. 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.

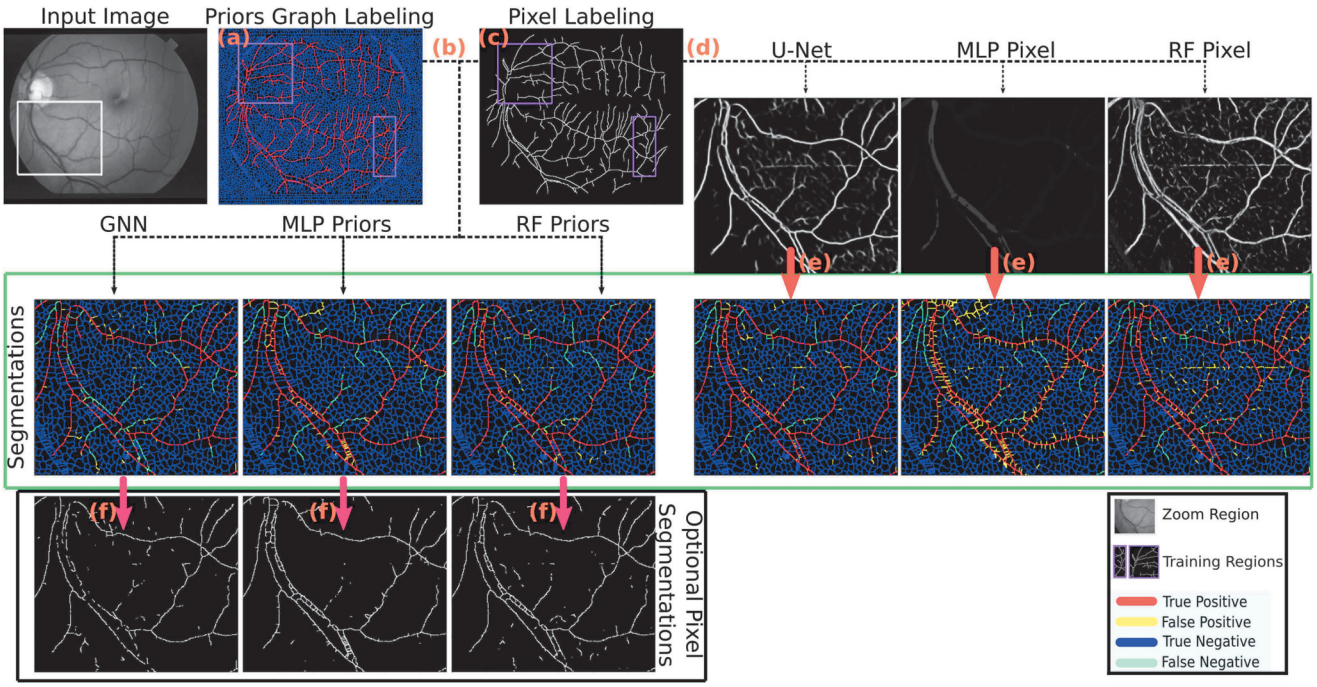


Fig. 4. The priors graph is computed over the image and manually labeled to create a ground truth (a). Subregions to train (pink) the ML models are identified so that the training region reasonably covers representative samples of the semantic object's diverse structures. The topological approaches train 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).

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 [41], [49]. What is more, it has also been shown that reduced human interaction allows for a better user experience [27]. Furthermore, approaches with a high computational accuracy that also enable user steering and editing demonstrate higher overall accuracy and repeatability [41]. 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.

TABLE I
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%

A. Pixel-Level Classification

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.

Priors Graph to Pixel Ground Truth: 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 Fig. 4(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.

Shallow Learning: We train a random forest classifier [9] for our shallow learning model. 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.

Deep Learning: We train two deep learning models to classify pixels: U-Net and MLP-Pixel [53], [54]. The architecture of MLP-Pixel is the same as that described in Section IV-C2, differing only in the learning rate and the number of epochs used for training as given in Table II. For both models, pixel

TABLE II
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

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.

U-Nets are a well-established and notably robust neural network architecture for image segmentation [53]. The canonical U-Net architecture is comprised of two main components following an autoencoder 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 [53]. 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 [53]. 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.

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 Fig. 4(e).

B. Experimental Setup

We outline in Fig. 4 our experimental design for comparing pixel-based methods against the prior-based workflow introduced in this work.

Data: We give more detailed summary statistics and descriptions of the datasets used for evaluation in Table I. 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 IV-B1), 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.

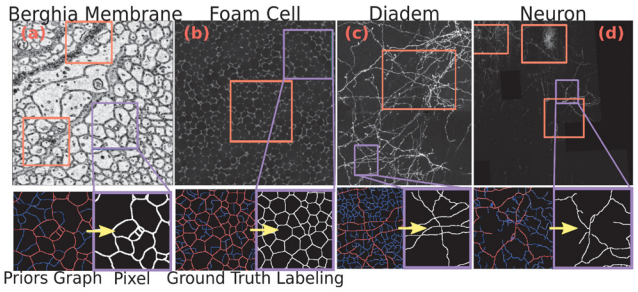


Fig. 5. We provide a summary for each dataset of the training regions (orange) used to achieve the accuracy results provided in Fig. 7 as well as an enlarged view of the prior graph and pixel-level ground truths (pink). Highlighted in orange is the subset used for training all models. Each original image highlights in pink the region expanded for the visualization of results provided in Fig. 8.

Retinal: Imaging and tracing of blood vessel arbors is used to classify disease states of the eye [59]. Obtaining a wire representation of the blood vessels is one of the first diagnostic steps. *Scalar field image:* Laplacian of the image with kernel size 2.0 to better capture faint blood vessels, as in Fig. 4.

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 [63]. Identifying the boundaries of neurons is an open challenge. *Scalar field image:* 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 Fig. 5(a). The user's (and ML) task is to classify priors as either "cell boundary" or not.

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. *Scalar field image:* 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 this field include both cell boundaries and noise, as shown in Fig. 5(b).

Diadem/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 [21] and macaque brain, neurons appear as

ridge-like structures in a noisy background field (Fig. 5(c), and (d) respectively). *Scalar field image:* Gaussian smoothed image with a kernel size of 2.0 to generate topological priors, which are classified as part of neurons or background.

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 $F_1 = 2(\frac{\text{precision}\cdot\text{recall}}{\text{precision}+\text{recall}})$. 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 $= \frac{\text{TP}}{\text{TP}+\text{FP}}$ and recall $= \frac{\text{TP}}{\text{TP}+\text{FN}}$. 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.

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 [64] to combat high class heterophily (see Section IV-C2). For other parameters, we stuck to default values used in official implementations, namely, the ensemble random forest classifier from scikit-learn [46], GraphSAGE's supervised GNN for vertex classification [26] and canonical UNet architecture [53]. 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 II.

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

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 IV-A 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.

TABLE III
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

C. Experimental Results

The main performance and timing results are shown in Fig. 6. 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 III, 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 Fig. 6 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 Fig. 6 we see that RF-Priors consistently lies on

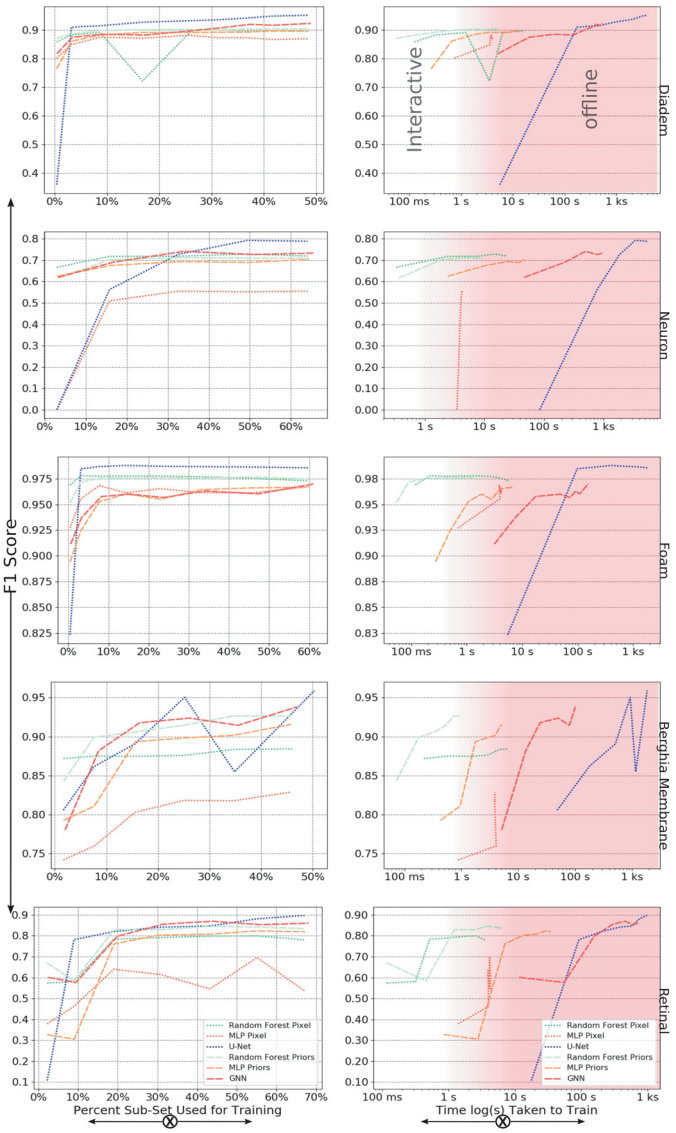


Fig. 6. F_1 scores for all methods based on training region size (left plot column) and the corresponding runtime to train (right plot column). Methods shown are RF-Pixel (dark green), MLP-Pixel (brown), U-Net (blue), RF-Priors (light green), MLP-Priors (amber), and GNN (red) over all datasets (plot 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 versus pixel-level predictions - particularly for small sizes of labeled data, the benefit being less effort is required from an annotator. Meanwhile, RF-Priors is the fastest method and reaches a highly competitive performance. In contrast, the pixel-level U-Net requires a much longer training time to reach higher F_1 scores.

(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.

	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
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
Berghia	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
Retinal	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

Fig. 7. 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.

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 Fig. 6 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.

In Fig. 7, 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 this table 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 [14], [20], [29], [44]. 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 Fig. 8, corresponding to the models trained in Fig. 7. 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

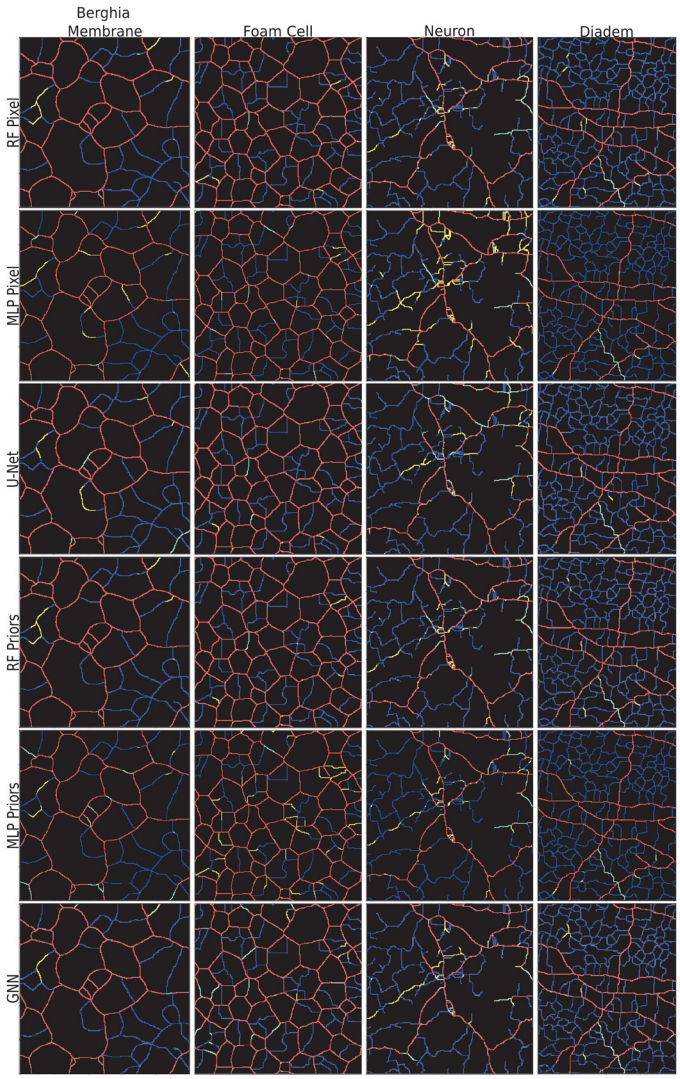


Fig. 8. Segmentation results are shown for each model. The regions shown correspond to the pink boxes in Fig. 5. The region(s) used for the training are those reported in Fig. 7. The segmentations are colored according to the given model's prediction as true positive (red), false positive (yellow), true negative (blue), and false negative (cyan).

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.

D. 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 Fig. 6 can be taken as a proxy for the amount of time a user would need to correct inference results using the labeling tool.

Table III augments the model performance (Fig. 7) 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 Fig. 6. 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.

VI. 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.

VII. Conclusion

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.

REFERENCES

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

[2] R. Achanta, A. Shaji, K. Smith, A. Lucchi, P. Fua, and S. Süsstrunk, "SLIC superpixels," SLIC Superpixels, EPFL Tech. Rep. 149300, Jun., 2010.

[3] R. Achanta, A. Shaji, K. Smith, A. Lucchi, P. Fua, and S. Süsstrunk, "SLIC superpixels compared to state-of-the-art superpixel methods," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 34, no. 11, pp. 2274–2282, Nov. 2012.

[4] P. P. Acharjya and D. Ghoshal, "Watershed segmentation based on distance transform and edge detection techniques," *Int. J. Comput. Appl.*, vol. 52, no. 13, pp. 6–10, 2012.

[5] S. Banerjee et al., "Semantic segmentation of microscopic neuroanatomical data by combining topological priors with encoder–decoder deep networks," *Nature Mach. Intell.*, vol. 2, no. 10, pp. 585–594, 2020.

[6] P. Bendich, D. Cohen-Steiner, H. Edelsbrunner, J. Harer, and D. Morozov, "Inferring local homology from sampled stratified spaces," in *Proc. IEEE 48th Annu. Symp. Found. Comput. Sci.*, 2007, pp. 536–546.

[7] S. Beucher, "Watersheds of functions and picture segmentation," in *Proc. IEEE Int. Conf. Acoust. Speech Signal Process.*, 1982, pp. 1928–1931.

[8] 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," *J. Comput. Chem.*, vol. 39, no. 16, pp. 936–952, 2018.

[9] L. Breiman, "Random forests," *Mach. Learn.*, vol. 45, no. 1, pp. 5–32, 2001.

[10] P.-T. Bremer, H. Edelsbrunner, B. Hamann, and V. Pascucci, "A multi-resolution data structure for two-dimensional Morse-Smale functions," in *Proc. IEEE Visual.*, 2003, pp. 139–146, doi: 10.1109/VISUAL.2003.1250365.

[11] 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 Trans. Vis. Comput. Graph.*, vol. 17, no. 9, pp. 1307–1324, Sep. 2011.

[12] F. Chen, H. Yu, R. Hu, and X. Zeng, "Deep learning shape priors for object segmentation," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2013, pp. 1870–1877.

[13] T. K. Dey, J. Wang, and Y. Wang, "Road network reconstruction from satellite images with machine learning supported by topological methods," in *Proc. 27th ACM SIGSPATIAL Int. Conf. Adv. Geographic Inf. Syst.*, 2019, pp. 520–523.

[14] 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," *J. Microsc.*, vol. 259, no. 2, pp. 114–120, 2015.

[15] H. Edelsbrunner and J. Harer, *Computational Topology: An Introduction*. Providence, RI, USA: Amer. Math. Soc., 2010.

[16] H. Edelsbrunner, D. Letscher, and A. Zomorodian, "Topological persistence and simplification," in *Proc. 41st Annu. Symp. Found. Comput. Sci.*, 2000, pp. 454–463.

[17] S. Eslami, N. Heess, C. K. Williams, and J. Winn, "The shape Boltzmann machine: A strong model of object shape," *Int. J. Comput. Vis.*, vol. 107, no. 2, pp. 155–176, 2014.

[18] R. Forman, "A user's guide to discrete Morse theory," *Sém. Lothar. Combin*, vol. 48, 2002, Art. no. B48c.

[19] A. F. Frangi, W. J. Niessen, K. L. Vincken, and M. A. Viergever, "Multiscale vessel enhancement filtering," in *Proc. Int. Conf. Med. Image Comput. Comput.- Assist. Interv.*, Springer, 1998, pp. 130–137.

[20] J. Funke et al., "Large scale image segmentation with structured loss based deep learning for connectome reconstruction," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 41, no. 7, pp. 1669–1680, Jul. 2019.

[21] 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.

[22] A. Gyulassy, "MSCEER: Morse-Smale complex extraction, exploration, reasoning," 2018. [Online]. Available: https://github.com/scivisus/MSCEER

[23] A. Gyulassy, P.-T. Bremer, B. Hamann, and V. Pascucci, "A practical approach to Morse-Smale complex computation: Scalability and generality," *IEEE Trans. Vis. Comput. Graph.*, vol. 14, no. 6, pp. 1619–1626, Nov./Dec. 2008.

[24] A. Gyulassy, P.-T. Bremer, and V. Pascucci, "Shared-memory parallel computation of Morse-Smale complexes with improved accuracy," *IEEE Trans. Vis. Comput. Graph.*, vol. 25, no. 1, pp. 1183–1192, Jan. 2019.

[25] A. Gyulassy, V. Natarajan, V. Pascucci, P.-T. Bremer, and B. Hamann, "A topological approach to simplification of three-dimensional scalar functions," *IEEE Trans. Vis. Comput. Graph.*, vol. 12, no. 4, pp. 474–484, Jul./Aug. 2006.

[26] W. L. Hamilton, R. Ying, and J. Leskovec, "Inductive representation learning on large graphs," in *Proc. 31st Int. Conf. Neural Inf. Process. Syst.*, 2017, pp. 1025–1035.

[27] 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," in *Proc. 1st IEEE Workshop User-Centered Comput. Vis.*, 2013, pp. 19–24.

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

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

[30] X. Hongjuan and G. Yaolin, "Application of topological analysis in ocean feature extraction," *Comput. Eng.*, vol. 35, no. 3, 2009, Art. no. 263, doi: 10.3969/j.issn.1000-3428.2009.03.090.

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

[32] T. N. Kipf and M. Welling, "Semi-supervised classification with graph convolutional networks," in *Proc. Int. Conf. Learn. Representations*, 2017, pp. 1–14.

[33] A. Klibisz, D. Rose, M. Eicholtz, J. Blundon, and S. Zakharenko, "Fast, simple calcium imaging segmentation with fully convolutional networks," in *Proc. Int. Workshop Deep Learn. Med. Image Anal. Multimodal Learn. Clin. Decis. Support*, Springer, 2017, pp. 285–293.

[34] K. Konyushkova, R. Sznitman, and P. Fua, "Introducing geometry in active learning for image segmentation," in *Proc. IEEE Int. Conf. Comput. Vis.*, 2015, pp. 2974–2982.

[35] D. Laney, P.-T. Bremer, A. Mascarenhas, P. Miller, and V. Pascucci, "Understanding the structure of the turbulent mixing layer in hydrodynamic instabilities," *IEEE Trans. Vis. Comput. Graph.*, vol. 12, no. 5, pp. 1053–1060, Sep./Oct. 2006.

[36] T. Lindeberg, "Scale-space theory: A basic tool for analyzing structures at different scales," *J. Appl. Statist.*, vol. 21, no. 1/2, pp. 225–270, 1994.

[37] D. Liu, Y. Xiong, K. Pulli, and L. Shapiro, "Estimating image segmentation difficulty," in *Proc. Int. Workshop Mach. Learn. Data Mining Pattern Recognit.*, Springer, 2011, pp. 484–495.

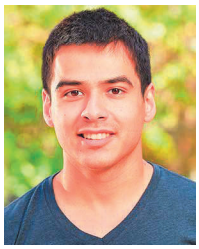
[38] J. Mayer, A. Robert-Moreno, J. Sharpe, and J. Swoger, "Attenuation artifacts in light sheet fluorescence microscopy corrected by OPTiSPIM," *Light Sci. Appl.*, vol. 7, no. 1, pp. 1–13, 2018.

[39] T. McDonald et al., "Improving the usability of virtual reality neuron tracing with topological elements," *IEEE Trans. Vis. Comput. Graph.*, vol. 27, no. 2, pp. 744–754, Feb. 2021.

[40] M. Moor, M. Horn, B. Rieck, and K. Borgwardt, "Topological autoencoders," in *Proc. Int. Conf. Mach. Learn.*, 2020, pp. 7045–7054.
[41] S. D. Olabarriaga and A. W. Smeulders, "Interaction in the segmentation of medical images: A survey," *Med. Image Anal.*, vol. 5, no. 2, pp. 127–142, 2001.
[42] 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," in *Proc. Int. Conf. Neural Inf. Process. Syst.*, 2013, pp. 1745–1753.
[43] N. R. Pal and S. K. Pal, "A review on image segmentation techniques," *Pattern Recognit.*, vol. 26, no. 9, pp. 1277–1294, 1993.
[44] C. Pape et al., "Leveraging domain knowledge to improve microscopy image segmentation with lifted multicuts," *Front. Comput. Sci.*, vol. 1, 2019, Art. no. 6.
[45] V. Pascucci, X. Tricoche, H. Hagen, and J. Tierny, *Topological Methods in Data Analysis and Visualization: Theory, Algorithms, and Applications*. Berlin, Germany: Springer, 2010.
[46] F. Pedregosa et al., "Scikit-learn: Machine learning in Python," *J. Mach. Learn. Res.*, vol. 12, pp. 2825–2830, 2011.
[47] S. Petruzza et al., "High-throughput feature extraction for measuring attributes of deforming open-cell foams," *IEEE Trans. Vis. Comput. Graph.*, vol. 26, no. 1, pp. 140–150, Jan. 2020.
[48] D. L. Pham, C. Xu, and J. L. Prince, "Current methods in medical image segmentation," *Annu. Rev. Biomed. Eng.*, vol. 2, no. 1, pp. 315–337, 2000.
[49] H. Ramadan, C. Lachqar, and H. Tairi, "A survey of recent interactive image segmentation methods," *Comput. Vis. Media*, vol. 6, no. 4, pp. 355–384, 2020.
[50] P. Ricci et al., "Removing striping artifacts in light-sheet fluorescence microscopy: A review," *Prog. Biophys. Mol. Biol.*, vol. 168, pp. 52–65, 2022.
[51] V. Robins, P. J. Wood, and A. P. Sheppard, "Theory and algorithms for constructing discrete Morse complexes from grayscale digital images," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 33, no. 8, pp. 1646–1658, Aug. 2011, doi: 10.1109/TPAMI.2011.95.
[52] 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.
[53] O. Ronneberger, P. Fischer, and T. Brox, "U-Net: Convolutional networks for biomedical image segmentation," in *Proc. Int. Conf. Med. Image Comput. Comput.- Assist. Interv.*, Springer, 2015, pp. 234–241.
[54] F. Rosenblatt, "The perceptron: A probabilistic model for information storage and organization in the brain," *Psychol. Rev.*, vol. 65, no. 6, 1958, Art. no. 386.
[55] C.-T. Shih et al., "NeuroRetriever: Automatic neuron segmentation for connectome assembly," *Front. Syst. Neurosci.*, vol. 15, 2021, Art. no. 687182.
[56] M. Sonka, V. Hlavac, and R. Boyle, *Image Processing, Analysis, and Machine Vision*. Boston, MA, USA: Cengage Learning, 2014.
[57] M. Sweet, C. P. Earls, M. Melcher, and B. Spitzak, "FLTK 1.1. 10 programming manual," FLTK Revision, Oct., 1998. [Online]. Available: http://www.fltk.org
[58] J. Tierny, G. Favelier, J. A. Levine, C. Gueunet, and M. Michaux, "The topology toolkit," *IEEE Trans. Vis. Comput. Graph.*, vol. 24, no. 1, pp. 832–842, Jan. 2018.
[59] E. Van der Merwe and S. Kidson, "Advances in imaging the blood and aqueous vessels of the ocular limbus," *Exp. Eye Res.*, vol. 91, no. 2, pp. 118–126, 2010.
[60] S. van der Walt et al., "Scikit-image: Image processing in Python," *PeerJ*, vol. 2, 2014, Art. no. e453, doi: 10.7717/peerj.453.
[61] A. Venkat et al., "Towards replacing physical testing of granular materials with a topology-based model," *IEEE Trans. Vis. Comput. Graph.*, vol. 28, no. 1, pp. 76–85, Jan. 2022.
[62] P. Virtanen et al., "SciPy 1.0: Fundamental algorithms for scientific computing in Python," *Nature Methods*, vol. 17, no. 3, pp. 261–272, 2020.
[63] D. B. Wilburn and W. J. Swanson, "From molecules to mating: Rapid evolution and biochemical studies of reproductive proteins," *J. Proteomic.*, vol. 135, pp. 12–25, 2016.
[64] K. Xu, C. Li, Y. Tian, T. Sonobe, K. Kawarabayashi, and S. Jegelka, "Representation learning on graphs with jumping knowledge networks," in *Proc. 35th Int. Conf. Mach. Learn.*, 2018, pp. 5449–5458.
[65] Q. Zhao, Z. Ye, C. Chen, and Y. Wang, "Persistence enhanced graph neural network," in *Proc. Int. Conf. Artif. Intell. Statist.*, 2020, pp. 2896–2906.
[66] J. Zhu, Y. Yan, L. Zhao, M. Heimann, L. Akoglu, and D. Koutra, "Beyond homophily in graph neural networks: Current limitations and effective designs," in *Proc. 34th Int. Conf. Neural Inf. Process. Syst.*, 2020, Art. no. 653.

Samuel Leventhal is currently working toward the PhD degree with the University of Utah's Scientific Computing and Imaging Institute under Valerio Pascucci and a research collaborator with Lawrence Livermore National Laboratory. His interests lie in the development of practical and theoretically justified machine learning (ML) models informed topologically or integrated with computational topology (CTo), which includes representation learning of graphs, the development and application of CTo techniques, and the development and application of topologically informed ML models.

Attila Gyulassy is a research computer scientist who works with Dr. Valerio Pascucci at the Scientific Computing and Imaging Institute, University of Utah. His research interests involve developing and applying techniques from computational topology to solve analysis problems in a variety of scientific domains, with other interests in visualization and explainable AI.



Mark Heimann received the PhD degree in computer science from the University of Michigan, in 2020. He is a computer science researcher with the Center for Applied Scientific Computing, Lawrence Livermore National Laboratory. His research focuses on machine learning for graphs and applications to problems in biomedicine and software analysis.

Valerio Pascucci is the Inaugural John R. Parks endowed chair, the founding director of the Center for Extreme Data Management Analysis and Visualization (CEDMAV), a faculty of the Scientific Computing and Imaging Institute, a professor of the School of Computing, University of Utah (UoU), the president of ViSOAR LLC, a UoU spin-off, and the founder of Data Intensive Science, a 501(c) nonprofit providing outreach and training to promote the use of advanced technologies for science and engineering.