Topologically Faithful Image Segmentation via Induced Matching of Persistence Barcodes

Nico Stucki [*1 2 3] **Johannes C. Paetzold** [*4] **Suprosanna Shit** [*1 5 6] **Bjoern H. Menze** [6] **Ulrich Bauer** [1 2 3]

Abstract

Segmentation models predominantly optimize pixel-overlap-based loss, an objective that is actually inadequate for many segmentation tasks. In recent years, their limitations fueled a growing interest in topology-aware methods, which aim to recover the topology of the segmented structures. However, so far, existing methods only consider global topological properties, ignoring the need to preserve topological features spatially, which is crucial for accurate segmentation. We introduce the concept of induced matchings from persistent homology to achieve a spatially correct matching between persistence barcodes in a segmentation setting. Based on this concept, we define the *Betti matching error* as an interpretable, topologically and feature-wise accurate metric for image segmentations, which resolves the limitations of the *Betti number error*. Our Betti matching error is differentiable and efficient to use as a loss function. We demonstrate that it improves the topological performance of segmentation networks significantly across six diverse datasets while preserving the performance with respect to traditional scores. Our code is publicly available[1].

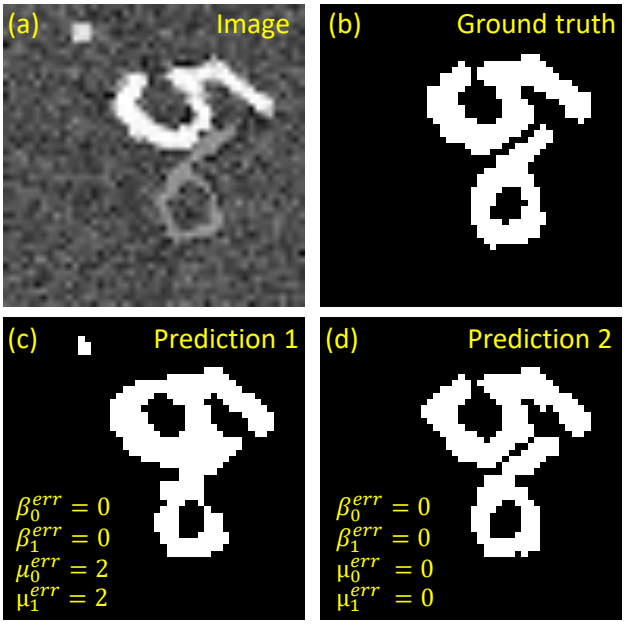


Figure 1: Exemplary segmentations of identical Dice scores from models trained with *Wasserstein loss* (c) and Betti matching loss (d). Dice and Betti number error (β^{err}) are indecisive between both predictions. On the other hand, our Betti matching error (μ^{err}) favors the superior segmentation in (d).

1. Introduction

Topology studies properties of shapes that are related to their connectivity and that remain unchanged under deformations, translations, and twisting. Some topological concepts, such as *cubical complexes*, *homology*, and *Betti numbers* (which in dimenion 2 count connected components and holes), form interpretable descriptions of shapes in space that can be efficiently computed. Naturally, the topology of physical structures is highly relevant in machine learning tasks, where the preservation of its connectivity is crucial, a prominent example being image segmentation. Recently, a number of methods have been proposed to improve topology preservation in image segmentation for a wide range of applications.

While spatial agreement between image and segmentation is a critical aspect of segmentations, all existing topology-

[*]Equal contribution [1]TUM School of Computation, Information and Technology, Technical University of Munich, Germany [2]Munich Data Science Institute, Germany [3]Munich Center for Machine Learning, Germany [4]Department of Computing, Imperial College London, United Kingdom [5]Department of Neuroradiology, Klinikum rechts der Isar, Germany [6]Department of Quantitative Biomedicine, University of Zurich, Switzerland. Correspondence to: Nico Stucki <nico.stucki@tum.de>, Johannes C. Paetzold <j.paetzold@ic.ac.uk>, Suprosanna Shit <suprosanna.shit@tum.de>.

Proceedings of the 40^{th} International Conference on Machine Learning, Honolulu, Hawaii, USA. PMLR 202, 2023.

[1]https://github.com/nstucki/Betti-matching/

aware methods so far only consider the global topology but ignore the spatial correspondence of features across images (see Fig. 1).

Our contribution

In this work we overcome this key limitation by introducing a rigorous framework for faithfully quantifying the preservation of local topological properties in the context of image segmentation, see Fig. 1. Our method builds on *induced matchings* between *persistence barcodes*, a concept from algebraic topology introduced by (Bauer & Lesnick, 2015). The introduction of these matchings to a machine learning setting allows us to formalize precisely the spatial correspondences between topological features of two grayscale images, by embedding both images into a common *comparison image*. Put in simple terms, our central contribution is an efficient, differentiable solution for localized topological error identification, which serves as:

- a *topological loss* to train segmentation networks, which guarantees to correctly, in a spatial sense, emphasize and penalize the topological structures during training (see Sec. 3.2);
- an *interpretable topological metric* for image segmentation, which is not only sensitive to the number of topological features but also to their location within the respective images (see Sec. 3.3).

Experimentally, we demonstrate that using our Betti matching loss function leads to vastly improved segmentations across six diverse datasets.

1.1. Related work

Stability of persistence and induced matchings Several proofs for the stability of persistence can be found in the literature. In 2005, (Cohen-Steiner et al., 2005) established a first stability result for *persistent homology* of real-valued functions. The result states that the map sending a function to the *barcode* of its sublevel sets is 1-Lipschitz with respect to suitable metrics. In 2008 this result was generalized by (Chazal et al., 2009b) and formulated in purely algebraic terms, in what is now known as the algebraic stability theorem. It states that the existence of a δ-interleaving (a sort of approximate isomorphism) between two pointwise finite-dimensional persistence modules implies the existence of a δ-matching between their respective barcodes. This theorem provides the justification for the use of persistent homology to study noisy data. In (Bauer & Lesnick, 2015), the authors present a constructive proof of this theorem, which associates to a given δ-interleaving between persistence modules a specific δ-matching between their barcodes. For this purpose, they introduce the notion of induced matchings, which form the foundation of our proposed Betti matching framework. Beyond their theoretical use in the proof of stability, induced matchings have been utilized in a computational setting for idenyifying corresponding topological features (Reani & Bobrowski, 2022; García-Redondo et al., 2022).

Topology aware segmentation Various publications have highlighted the importance of topologically correct segmentations in computer vision and image analysis applications. *Persistent homology* is a popular framework from algebraic topology that has been utilitzed in this context. A key publication by (Hu et al., 2019) proposes to improve image segmentation using a loss function that we refer to as the *Wasserstein loss*, based on a variation of the *Wasserstein distance* between *persistence diagrams* (an alternative to barcodes as descriptor of persistent homolgy). Specifically, the authors propose to match features in the persistence diagrams of dimension 1 between ground truth and prediction so as to minimize the squared distance of matched points. However, this approach has a fundamental limitation, in that it cannot guarantee that the matched structures are spatially related in any sense (see Fig. 2 and App. A). Put succinctly, the cycles are matched irrespective of the location within the image, which frequently has an adverse impact during training (see App. G). (Clough et al., 2020) follows a similar approach, not computing the barcode of the ground truth segmentation, but using only the Betti numbers it ought to have. Furthermore, persistent homology has also been used in other similar problems, in particular, for crowd localization (Abousamra et al., 2021) and for reconstructing 3D cell shapes from 2D images (Waibel et al., 2022).

Other topology-aware segmentation methods incorporate pixel-overlaps of topologically relevant structures. For example, the *clDice* score, introduced by (Shit et al., 2021), targets the segmentation of tubular structures such as vascular networks. It is a variant of the commonly used *Dice* similarity coefficient (Dice, 1945), incorporating a skeleton for foreground and background in such a way that a score of 0 guarantees topological equivalence of ground truth and prediction. (Hu & Chen, 2021) and (Jain et al., 2010) use *homotopy warping* to identify critical pixels and measure the topological difference between grayscale images. (Hu et al., 2021) utilizes *discrete Morse theory* (see (Delgado-Friedrichs et al., 2014)) to compare critical topological structures within prediction and ground truth. (Wang et al., 2022) incorperate a *marker loss*, which is based on the Dice loss between a predicted marker map and the ground truth marker map, to improve fine antomical structure segmentation topologically. Generally, these overlap-based approaches are computationally efficient but do not explicitly guarantee the spatial correspondence of the topological features. Other approaches aim at enforcing topologically motivated priors, e.g., connectivity priors (Chen et al., 2011; Sasaki et al., 2017; Wang & Jiang, 2018). (Mosinska et al.,

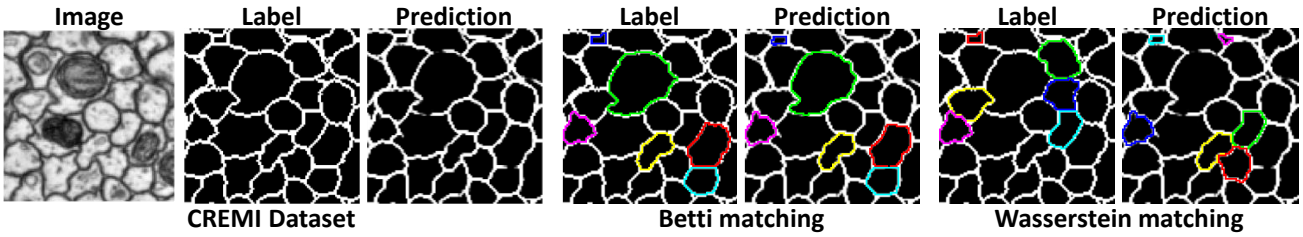


Figure 2: Comparison of our Betti matching and *Wasserstein matching* ((Hu et al., 2019)). We match cycles between label and prediction for a CREMI image and highlight matched pairs in the same color. We visualize only six (randomly selected out of the total 23 matches for both methods) matched pairs for presentation clarity. Note that Betti matching always matches spatially correctly while the Wasserstein matching gets most matches wrong. For more examples please consider Figures 10-13 in the Appendix.

2018) applied task-specific pre-trained filters to improve connected components. (Zhang & Lui, 2022) uses template masks as an input to enforce the diffeomorphism of a specific shape. (Cheng et al., 2021) jointly models connectivity and features based on iterative feedback learning. (Oner et al., 2020) aims to improve the topological performance by enforcing region separation of curvilinear structures.

2. Background on algebraic topology

We introduce the necessary concepts from algebraic topology to describe the construction of induced matchings for images. For the basic definitions, we refer to Appendix N.

2.1. Images as filtered cubical complexes

The topology of an image $\boldsymbol{I} \in \mathbb{R}^{m \times n}$ (e.g., a prediction or ground truth segmentation) is best captured by *filtered cubical complexes*. In order to filter a *cubical complex* K we consider an *order preserving* function $f\colon K \to \mathbb{R}$. Its **sublevel sets** $D(f)_r := f^{-1}((-\infty, r])$ assemble to the **sublevel filtration** $D(f) = \{D(f)_r\}_{r\in\mathbb{R}}$ of K. Since f can only take finitely many values $\{f_1 < \ldots < f_l\}$, the filtered cubical complex K_* given by $K_i = D(f)_{f_i}$ for $i = 1, \ldots, l$, encodes all the information about the filtration.

We consider the **cubical grid complex** $K^{m,n}$ consisting of all *cubical cells* contained in $[1, m] \times [1, n] \subseteq \mathbb{R}^2$. The **filter function** f_I of $\boldsymbol{I}$ is defined on the vertices of $K^{m,n}$ by the corresponding entry in $\boldsymbol{I}$, and on all higher-dimensional cubes as the maximum value of its vertices. Note that $f_{\boldsymbol{I}}$ is order preserving, so we can associate the sublevel filtration of $f_{\boldsymbol{I}}$ and its corresponding filtered cubical complex to the image $\boldsymbol{I}$ and denote them by $D(\boldsymbol{I})$ and $K_*(\boldsymbol{I})$, respectively. This construction is called the **V-construction** since pixels are treated as vertices in the cubical complex, see Fig. 4b. An alternative, the **T-construction**, considers pixels as top-dimensional cells of a 2-dimensional cubical complex (see (Heiss & Wagner, 2017)). We implemented both, V- and T-construction, in Betti matching and encode them in the *ValueMap* array inside the `CubicalPersistence` class in Algo. 1.

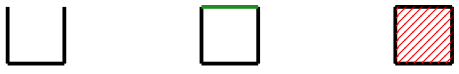
2.2. Homology and induced maps

Homology is a powerful concept involving local computations to capture information about the global structure of a topological space X. For each $d \in \mathbb{N}_0$ it assigns an abelian **Homology group** $H_d(X)$ to X, which encodes its topological features in dimension d. A feature in dimension 0 describes a connected component, and in dimension 1, it describes a hole. Considering coefficients in $\mathbb{F}_2$, these abelian groups form $\mathbb{F}_2$-vector spaces and the dimension of $H_d(X)$ is called the dth **Betti number** of X, which is denoted by $\beta_d(X)$.

Homology is a functor, i.e., it does not only act on spaces, but also on maps between spaces. Therefor, a continuous map $g\colon X \to Y$ (e.g., an inclusion) induces linear maps $H_d(g)\colon H_d(X) \to H_d(Y)$ in each dimension $d \in \mathbb{N}_0$, which allow us to identify homological features of one space as homological features of another space and the map g induces the identification. For more details, we refer to (Kaczynski et al., 2004) and App. N.2, where we recap the homology of cubical complexes with coefficients in $\mathbb{F}_2$.

2.3. Persistent homology and its barcode

Persistent homology considers sublevel filtrations of spaces and observes the lifetime of topological features within the filtration in form of *persistence modules*. The basic premise is that features that persist for a long time are significant, whereas features with a short lifetime are likely to be caused by noise.

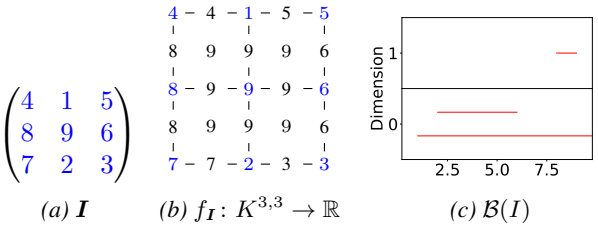


(a) K_1 *(b)* K_2 *(c)* K_3

Figure 3: A filtered cubical complex with varying homology in degree 1. Adding the green 1-cell in (b) creates homology (birth) and adding the red 2-cell in (c) turns homology trivial (death). Together they form a *persistence pair*.

The **persistent homology** $H_d(f)$ of an order preserving function $f\colon K \to \mathbb{R}$ in dimension $d \in \mathbb{N}_0$ consists of vector spaces $H_d(f)_r = H_d(D(f)_r)$ and transition maps $H_d(f)_{r,s}\colon H_d(D(f)_r) \to H_d(D(f)_s)$ induced by the inclusions $D(f)_r \hookrightarrow D(f)_s$ for $r \leq s$. Note that $H_d(f)$ is a pointwise finite-dimensional (p.f.d.) persistence module, and by a result of (Crawley-Boevey, 2015), any p.f.d. persistence module M is isomorphic to a direct sum of interval modules $C(\mathcal{I})$: $M \cong \bigoplus_{\mathcal{I}\in\mathcal{B}(M)} C(\mathcal{I})$. Here, $\mathcal{B}(M)$ denotes the **barcode** of M, given by a *multiset* of intervals. Note that the persistent homology is **continuous from above**: all intervals in the barcode are of the form $[s, t)$.

Barcodes of Images For an image $\boldsymbol{I} \in \mathbb{R}^{m\times n}$ with associated filter function $f_{\boldsymbol{I}}\colon K^{m,n} \to \mathbb{R}$, we will refer to the persistent homology of $f_{\boldsymbol{I}}$ in dimension d as the persistent homology of the image $\boldsymbol{I}$ in dimension d and denote it by $H_d(\boldsymbol{I})$. Its associated barcode in dimension d will be denoted by $\mathcal{B}_d(\boldsymbol{I})$ and we call $\mathcal{B}(\boldsymbol{I}) = \bigcup_{d\in\mathbb{N}_0} \mathcal{B}_d(\boldsymbol{I})$ the barcode of $\boldsymbol{I}$.



(a) $\boldsymbol{I}$ *(b)* $f_{\boldsymbol{I}}\colon K^{3,3} \to \mathbb{R}$ *(c)* $\mathcal{B}(I)$

Figure 4: (a) shows an image $\boldsymbol{I}$, (b) visualizes the V-construction and (c) shows the associated barcode $\mathcal{B}(\boldsymbol{I})$. The sublevel filtration squentially adds pixels in order of increasing value. By adding 1 and 2, two connected components (represented by the two bars in dimension 0) are born. Adding 6 merges the two coonected components ending the finite interval $[2, 6)$. The other connected component persists forever, which is represented by the essential interval $[1, \infty)$. By adding 8 a hole is formed, which is filled by adding 9. This is represented by the interval $[8, 9)$ in dimension 1.

In order to compute the barcode $\mathcal{B}(\boldsymbol{I})$, we make use of the *reduction algorithm* described in (Edelsbrunner et al., 2008). It starts by sorting the cells of the associated filtered cubical complex $K_*(\boldsymbol{I})$ to obtain a *compatible* ordering $c_1, \ldots, c_l$, which defines a **cell-wise refinement** $L_*(\boldsymbol{I})$. Here, compatible means that the cells in K_i preceed the cells in $K \setminus K_i$, and the faces of a cell preceed the cell. We encode this ordering in the *IndexMap* array inside the `CubicalPersistence` class in Algo. 1. The algoritm then performs a variant of Gaussian elimination on the *boundary matrix* of $K^{m,n}$, where rows and columns are indexed with respect to the compatible ordering. Adding a d-cell c_k to the complex will either create new homology classes in dimension d or turn homology classes trivial in dimension $d-1$ (see Figure 3). In the latter case, assuming that the classes that become trivial have been created by adding cell c_j, we pair the cells c_j and c_k to a **persistence pair** (c_j, c_k). The unpaired cells are called **singletons**. Each pair (c_j, c_k) satisfying $f_{\boldsymbol{I}}(c_j) f_{\boldsymbol{I}}(c_k)$ gives rise to a **finite interval** $[f_{\boldsymbol{I}}(c_j), f_{\boldsymbol{I}}(c_k)) \in \mathcal{B}(\boldsymbol{I})$, and each singleton c_i gives rise to an **essential interval** $[f_{\boldsymbol{I}}(c_i), \infty) \in \mathcal{B}(\boldsymbol{I})$.

Note that a finite interval $[f_{\boldsymbol{I}}(c_j), f_{\boldsymbol{I}}(c_k)) \in \mathcal{B}_d(\boldsymbol{I})$ determines a **refined (finite) interval** $[j, k)$ and an essential interval $[f_{\boldsymbol{I}}(c_i), \infty) \in \mathcal{B}_d(\boldsymbol{I})$ determines a **refined (essential) interval** $[i, \infty)$. Collectively, we call the set $\mathcal{B}_d^{\text{fine}}(\boldsymbol{I})$ consisting of refined intervals in dimension d the **refined barcode** in dimension d of $\boldsymbol{I}$ and $\mathcal{B}^{\text{fine}}(\boldsymbol{I}) = \bigcup_{d\in\mathbb{N}_0} \mathcal{B}_d^{\text{fine}}(\boldsymbol{I})$ the refined barcode of $\boldsymbol{I}$. Therefore, we can consider $H_d(\boldsymbol{I})$ as **staggered** persistence module, meaning that the intervals in its barcode have unique endpoints.

2.4. Induced matchings between persistence barcodes

Following the idea of induced maps in homology (see section 2.2), (Bauer & Lesnick, 2015) introduces the notion of induced matchings of persistence barcodes, which allow us to identify correspondances of intervals in the barcodes of images and play a central role in our Betti matching. The following theorem (paraphrased as a special case of the general Theorem 4.2 in (Bauer & Lesnick, 2015)) is key to the definition of induced matchings:

Theorem 2.1. *Let $\Phi\colon M \to N$ be a morphism of p.f.d., staggered persistence modules that are continuous from above. Then there are unique injective maps $\mathcal{B}(\operatorname{im}\Phi) \hookrightarrow \mathcal{B}(M)$ and $\mathcal{B}(\operatorname{im}\Phi) \hookrightarrow \mathcal{B}(N)$, which map an interval $[b, c) \in \mathcal{B}(\operatorname{im}\Phi)$ to an interval $[b, d) \in \mathcal{B}(M)$ with $c \leq d$, and to an interval $[a, c) \in \mathcal{B}(N)$ with $a \leq b$, respectively.*

Note that $\operatorname{im}\Phi$ is a p.f.d. submodule of N, and we will refer to its barcode as the **image barcode** of Φ. Obviously, the injections in Theorem 2.1 determine matchings $\mathcal{B}(M) \xrightarrow{\sigma_M} \mathcal{B}(\operatorname{im}\Phi) \xrightarrow{\sigma_N} \mathcal{B}(N)$. The **induced matching** of Φ is then given by the composition $\sigma(\Phi) = \sigma_N \circ \sigma_M$.

Induced matchings of images Let $\boldsymbol{I}, \boldsymbol{J} \in \mathbb{R}^{m\times n}$ be images such that $\boldsymbol{I} \geq \boldsymbol{J}$ (entry-wise). Then the sublevel sets of $\boldsymbol{I}$ form subcomplexes of the sublevel sets of $\boldsymbol{J}$ and the inclusions $D(\boldsymbol{I})_r \hookrightarrow D(\boldsymbol{J})_r$ induce linear maps $H_d(\boldsymbol{I})_r \to H_d(\boldsymbol{J})_r$ in homology. These assemble to a morphism $\Phi_d(\boldsymbol{I}, \boldsymbol{J})\colon H_d(\boldsymbol{I}) \to H_d(\boldsymbol{J})$ between p.f.d. persistence modules, which are continuous from above. Furthermore, the refined barcodes $\mathcal{B}_d^{\text{fine}}(\boldsymbol{I}), \mathcal{B}_d^{\text{fine}}(\boldsymbol{J})$ allow us to consider $H_d(\boldsymbol{I})$ and $H_d(\boldsymbol{J})$ as staggered persistence modules and apply Theorem 2.1.

In the following we will denote the image barcode of $\Phi_d(\boldsymbol{I}, \boldsymbol{J})$ by $\mathcal{B}_d(\boldsymbol{I}, \boldsymbol{J})$. For the computation of the image barcode, we follow the algorithm described in (Bauer & Schmahl, 2022). It involves the *reduction* of the boundary matrix of $K^{m,n}$ with rows indexed by the ordering $c_1, \ldots, c_l$ in $L_*(\boldsymbol{I})$ and columns indexed by the ordering $d_1, \ldots, d_l$ in $L_*(\boldsymbol{J})$. The resulting reduced matrix yields

image persistence pairs (c_i, d_j), which satisfy $f_{\boldsymbol{I}}(c_i) < f_{\boldsymbol{J}}(d_j)$ and correspond to finite intervals $[f_{\boldsymbol{I}}(c_i), f_{\boldsymbol{J}}(d_j)) \in \mathcal{B}_d(\boldsymbol{I}, \boldsymbol{J})$. Following the structure of the induced matchings obtained by Theorem 2.1, we match a refined interval $[h, i) \in \mathcal{B}_d^{\text{fine}}(\boldsymbol{I})$ to a refined interval $[j, k) \in \mathcal{B}_d^{\text{fine}}(\boldsymbol{J})$ if the pair (c_h, d_k) is an image persistence pair. This way we obtain a matching $\sigma^{\text{fine}} \colon \mathcal{B}^{\text{fine}}(\boldsymbol{I}) \to \mathcal{B}^{\text{fine}}(\boldsymbol{J})$ between the refined barcodes, which yields the **induced matching** $\sigma(\boldsymbol{I}, \boldsymbol{J}) \colon \mathcal{B}(\boldsymbol{I}) \to \mathcal{B}(\boldsymbol{J})$ by replacing refined intervals with the corresponding intervals in $\mathcal{B}(\boldsymbol{I}), \mathcal{B}(\boldsymbol{J})$.

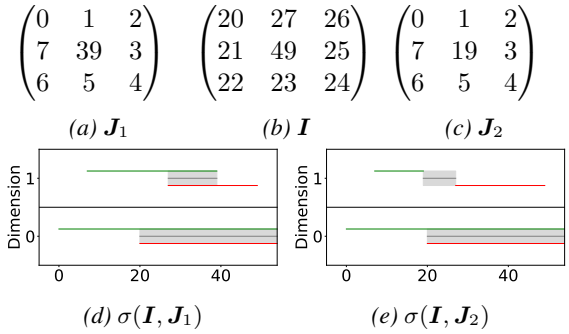


Figure 5: (a), (b) and (c) show images which satisfy $\boldsymbol{I} \geq \boldsymbol{J}_1, \boldsymbol{J}_2$. (d) and (e) visualize the induced matchings. Red bars correspond to the barcode of $\boldsymbol{I}$, green bars to the barcodes of $\boldsymbol{J}_1, \boldsymbol{J}_2$ and gray bars to the image barcodes $\mathcal{B}(\boldsymbol{I}, \boldsymbol{J}_1), \mathcal{B}(\boldsymbol{I}, \boldsymbol{J}_2)$, which are used to define the induced matchings $\sigma(\boldsymbol{I}, \boldsymbol{J}_1), \sigma(\boldsymbol{I}, \boldsymbol{J}_2)$. The shaded gray area indicates matched intervals (red and green bars) according to the agreement of endpoints.

In the present work, we augment this induced matching by additionally considering **reverse persistence pairs**, i.e., pairs (c_i, d_j), obtained by the reduction, that do not satisfy $f_{\boldsymbol{I}}(c_i) < f_{\boldsymbol{J}}(d_j)$ (see Figure 5e). When this is the case, we also match the corresponding intervals in $\mathcal{B}_{\text{fine}}(\boldsymbol{I})$ and $\mathcal{B}_{\text{fine}}(\boldsymbol{J})$ according to Theorem 2.1. Note that this is a slight variation of the induced matching defined in (Bauer & Lesnick, 2015). This extension satisfies similar properties and is a natural adaptation in this context.

3. Betti matching

In general, the structure of interest in segmentation tasks is given by the foreground. Therefore, we consider *superlevel filtrations* instead of sublevel filtrations in applications. For simplicity, we stick to sublevel filtrations to describe the theoretical background. Throughout this section, we denote by $\boldsymbol{L} \in [0, 1]^{m \times n}$ a **likelihood map** predicted by a deep neural network, by $\boldsymbol{P} \in \{0, 1\}^{m \times n}$ the binarized **prediction** of $\boldsymbol{L}$, and by $\boldsymbol{G} \in \{0, 1\}^{m \times n}$ the **ground truth** segmentation.

3.1. Matching by comparison in ambient space

In order to visualize that two objects in two different images are at the same location, we can simply move one image ontop of the other one and observe that the locations of the objects now agree. Thereby, we are constructing a common ambient space for both images which allows us to identify locations. Following this idea, in order to find a matching between $\mathcal{B}(\boldsymbol{L})$ and $\mathcal{B}(\boldsymbol{G})$ that takes the location of represented topological features into account, we are looking for a common *ambient filtration* of $K^{m,n}$, which is

(a) big enough to contain the sublevel sets of $\boldsymbol{L}$ and $\boldsymbol{G}$;

(b) fine enough to capture the topologies of $\boldsymbol{L}$ and $\boldsymbol{G}$.

Here, (a) guarantees that we can compute induced matchings of the respective inclusions and (b) guarantees that the identification of features by the induced matchings are non-trivial (discriminative). The most natural candidate which comes into mind is given by the union $D(\boldsymbol{L})_r \cup D(\boldsymbol{G})_r$ of sublevel sets. Therefore, we introduce the comparison image $\boldsymbol{C} = \min(\boldsymbol{L}, \boldsymbol{G})$ (entry-wise minimum) and observe that $D(\boldsymbol{C})_r = D(\boldsymbol{L})_r \cup D(\boldsymbol{G})_r$. By construction, we have $\boldsymbol{C} \leq \boldsymbol{L}, \boldsymbol{G}$ and obtain induced matchings $\sigma(\boldsymbol{L}, \boldsymbol{C}) \colon \mathcal{B}(\boldsymbol{L}) \to \mathcal{B}(\boldsymbol{C})$ and $\sigma(\boldsymbol{G}, \boldsymbol{C}) \colon \mathcal{B}(\boldsymbol{G}) \to \mathcal{B}(\boldsymbol{C})$ (see Sec. 2.4). The **Betti matching** $\mu(\boldsymbol{L}, \boldsymbol{G}) \colon \mathcal{B}(\boldsymbol{L}) \to \mathcal{B}(\boldsymbol{G})$ is then given by the composition

$$\mu(\boldsymbol{L}, \boldsymbol{G}) = \sigma(\boldsymbol{G}, \boldsymbol{C})^{-1} \circ \sigma(\boldsymbol{L}, \boldsymbol{C}), \tag{1}$$

where $\sigma(\boldsymbol{G}, \boldsymbol{C})^{-1}$ denotes the inverse of the matching $\sigma(\boldsymbol{G}, \boldsymbol{C})$. Working with superlevel sets yields an analogous construction. In the superlevel-setting we choose $\boldsymbol{C} = \max(\boldsymbol{L}, \boldsymbol{G})$ as the comparison image to guarantee that each superlevel set of the comparison image is the union of the corresponding superlevel sets of ground truth and likelihood map.

3.2. Betti matching defines topological loss

We denote by $\overline{\mathbb{R}}$ the **extended real line** $\mathbb{R} \cup \{-\infty, \infty\}$. A barcode $\mathcal{B}$ consisting of intervals $[a, b)$ can then equivalently be seen as a multiset $\mathrm{Dgm}(\mathcal{B})$ of points $(a, b) \in \overline{\mathbb{R}}^2$ which lie above the diagonal $\Delta = \{(x, x) \mid x \in \mathbb{R}\}$. Furthermore, we add all the points on the diagonal Δ with infinite multiplicity to $\mathrm{Dgm}(\mathcal{B})$ and thus define the **persistence diagram** of $\mathcal{B}$. A matching $\tau \colon \mathcal{B}_1 \to \mathcal{B}_2$ between barcodes then corresponds to a bijection $\tau \colon \mathrm{Dgm}(\mathcal{B}_1) \to \mathrm{Dgm}(\mathcal{B}_2)$ between persistence diagrams, by mapping unmatched points (a, b) to their closest point $((a + b)/2, (a + b)/2)$ on the diagonal Δ. We use these perspectives interchangeably (see Fig. 19). For simplicity, we denote by $\mathrm{Dgm}(\boldsymbol{I})$ the persistence diagram associated to the barcode of a grayscale image $\boldsymbol{I}$.

Persistent homology is stable, i.e., there exist metrics on the set of persistence diagrams for which slight variations in the input result in small variations of the corresponding persistence diagram (Chazal et al., 2009a). Therefore, it is natural to require $\mathrm{Dgm}(\boldsymbol{L})$ to be similar to $\mathrm{Dgm}(\boldsymbol{G})$. A frequently used metric to measure the difference between persistence

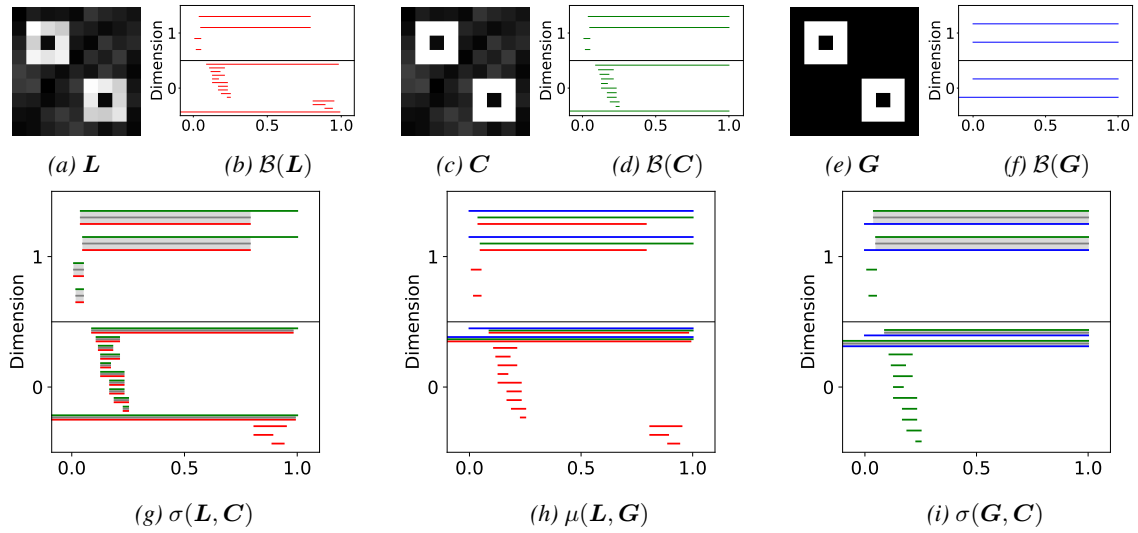


Figure 6: **Betti matching.** (a)–(f) show a likelihood map $\boldsymbol{L}$, a ground truth $\boldsymbol{G}$, the comparison image $\boldsymbol{C}$ and their barcodes. (g) and (i) show the induced matchings $\sigma(\boldsymbol{L}, \boldsymbol{C})\colon \mathcal{B}(\boldsymbol{L}) \to \mathcal{B}(\boldsymbol{C})$ and $\sigma(\boldsymbol{G}, \boldsymbol{C})\colon \mathcal{B}(\boldsymbol{G}) \to \mathcal{B}(\boldsymbol{C})$ (matchings indicated in gray) and (h) shows the resulting Betti matching $\mu(\boldsymbol{L}, \boldsymbol{G})\colon \mathcal{B}(\boldsymbol{L}) \to \mathcal{B}(\boldsymbol{G})$, which matches a red interval to a blue interval if there is a green interval in between. We use this matching to define our loss and metric.

diagrams is the Wasserstein distance (Cohen-Steiner et al., 2010), and it has been adapted to train segmentation networks (Hu et al., 2019). Because of the shortcomings described in Fig. 2,8b and App. A,G, we propose to replace the Wasserstein matching γ_* by the Betti matching $\mu(\boldsymbol{L}, \boldsymbol{G})$ and define the **Betti matching loss**

$$l_{\text{BM}}(\boldsymbol{L}, \boldsymbol{G}) = \sum_{q \in \text{Dgm}(\boldsymbol{L})} 2\|q - \mu(\boldsymbol{L}, \boldsymbol{G})(q)\|_2^2. \quad (2)$$

The factor 2 is added to simplify its interpretation as Betti matching error (see Sec. 3.3). Since the values in $\boldsymbol{L}$ and $\boldsymbol{G}$ are contained in $[0, 1]$, we replace the essential intervals $[a, \infty)$ with the finite interval $[a, 1]$, to obtain a well-defined expression. To efficiently train segmentation networks, we combine our Betti matching loss with a standard volumetric loss, specifically, the *Dice Loss*, to

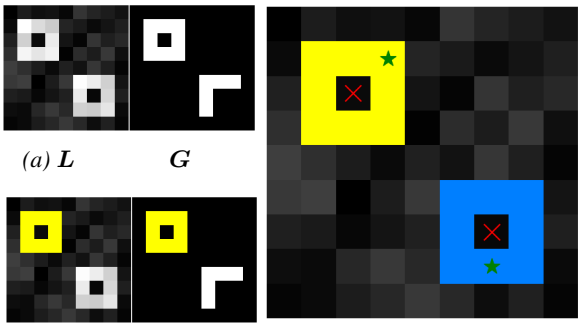
$$l_{train} = \alpha l_{\text{BM}}(\boldsymbol{L}, \boldsymbol{G}) + l_{\text{dice}}(\boldsymbol{L}, \boldsymbol{G}). \quad (3)$$

Gradient of Betti matching loss Note that we can see $\boldsymbol{L} = \boldsymbol{L}(\boldsymbol{I}, \omega)$ as a function that assigns the predicted likelihood map to an image $\boldsymbol{I} \in \mathbb{R}^{m \times n}$ and the segmentation network parameters $\omega \in \mathbb{R}^l$. A point $q = (q_1, q_2) \in \text{Dgm}(\boldsymbol{L})$ describes a topological feature that is born by adding pixel $b(q)$ (**birth** of q) and killed by adding pixel $d(q)$ (**death** of q) to the filtration. The coordinates of q are then determined by their values $q_1 = \boldsymbol{L}_{d(q)}$ and $q_2 = \boldsymbol{L}_{b(q)}$. Assuming that the Betti matching is constant in a sufficiently small neighborhood around the given predicted likelihood map $\boldsymbol{L}$, the Betti matching loss is differentiable in ω and the chain rule yields the gradient

$$\nabla_\omega l_{\text{BM}}(\boldsymbol{L}, \boldsymbol{G}) = \sum_{q \in \text{Dgm}(\boldsymbol{L})} 4(q_1 - \mu(\boldsymbol{L}, \boldsymbol{G})(q)_1) \frac{\partial \boldsymbol{L}_{d(q)}}{\partial \omega} + 4(q_2 - \mu(\boldsymbol{L}, \boldsymbol{G})(q)_2) \frac{\partial \boldsymbol{L}_{b(q)}}{\partial \omega}. \quad (4)$$

Note that likelihood maps for which this assumption is not satisfied may exist. But this requires $\boldsymbol{L}$ to have at least two entries with the exact same value, and the set of such likelihood maps has *Lebesgue measure* zero. Therefore, the gradient is well-defined almost everywhere, and in the edge cases, we consider it as a sub-gradient, which still reduces the loss and has a positive effect on the topology of the segmentation.

Physical meaning of the gradient To understand the effect of the Betti matching gradient during training, consider the example in Fig. 7. Let $x, y \in \text{Dgm}(\boldsymbol{L})$ denote the points corresponding to the yellow and blue cycle in (c), respectively. (b) shows that x is matched and y is unmatched. Since, all points in $\text{Dgm}(\boldsymbol{G})$ are of the form (0,1), Betti matching maps x to $(0, 1)$ and y to its closest point $(\frac{y_1+y_2}{2}, \frac{y_1+y_2}{2})$ on the diagonal Δ. Therefore, the gradient will enforce the segmentation network to move x closer to $(0, 1)$ (i.e., decrease $x_1 = \boldsymbol{L}_{d(x)}$ and increase $x_2 = \boldsymbol{L}_{b(x)}$) and y closer to $(\frac{y_1+y_2}{2}, \frac{y_1+y_2}{2})$ (i.e., increase $y_1 = \boldsymbol{L}_{d(y)}$ and decrease $y_2 = \boldsymbol{L}_{b(y)}$). This results in an amplification of the local contrast between $\star$ and $\times$ of the yellow cycle and a reduction of the local contrast between $\star$ and $\times$ of the blue cycle, which improves the topological performance of the segmentation.

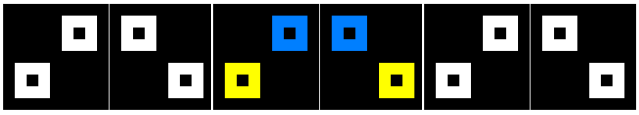


(a) $\boldsymbol{L}$ $\boldsymbol{G}$ *(b)* Betti matching *(c)* 1-cycles in $\boldsymbol{L}$

Figure 7: (a) $\boldsymbol{L}$ shows a Topological error (bottom right). (b) Matched cycles in Betti matching are shown in yellow. (c) For both cycles in $\boldsymbol{L}$, the birth ($b(q)$) and death pixels ($d(q)$) are marked with $\star$ and $\times$, respectively.

Summarized, we can say that matched features get emphasized, and unmatched features get suppressed during training, which highlights the importance of finding a spatially correct matching (see App. G for further discussion).

3.3. Betti matching error as topological metric



(a) $\beta_1^{\text{err}}(\boldsymbol{P},\boldsymbol{G})=0$ *(b)* $l_{\text{W}}(\boldsymbol{P},\boldsymbol{G})=0$ *(c)* $\mu_1^{\text{err}}(\boldsymbol{P},\boldsymbol{G})=4$

Figure 8: **Advantages of our Betti matching error over the Betti number error.** (a) shows a prediction $\boldsymbol{P}$ (left), ground truth $\boldsymbol{G}$ (right) and the corresponding Betti number error in dim 1. (b) shows the Wasserstein matching in dim 1 (same color indicates a matching) with its corresponding loss and (c) shows the Betti matching in dim 1 (no features are matched) with the corresponding Betti matching error. Note that both Betti number error and Wasserstein loss fail to represent the spatial mistake in the prediction, while the Betti matching correctly does not match any cycles resulting in an error of 4.

Betti number error The Betti number error β^{err} (see App. M) compares the topological complexity of the binarized prediction $\boldsymbol{P}$ and the ground truth $\boldsymbol{G}$. However, it is limited as it only compares the number of topological features in both images, while ignoring their spatial correspondence (see Fig. 8). In terms of persistence diagrams, the Betti number error can be expressed by considering a maximal matching $\beta\colon \text{Dgm}(\boldsymbol{P}) \to \text{Dgm}(\boldsymbol{G})$, e.g., the Wasserstein matching (see App. G), and counting the number of unmatched points:

$$\beta^{\text{err}}(\boldsymbol{P},\boldsymbol{G}) = \#\ker(\beta) + \#\,\text{coker}(\beta), \tag{5}$$

where for a matching σ we denote by $\#\ker(\sigma)$ and $\#\,\text{coker}(\sigma)$ the number of unmatched points in the domain of σ and in codomain of σ, respectively (see App. N.4).

We denote by $\mu^{err}(\boldsymbol{P},\boldsymbol{G}) := l_{\text{BM}}(\boldsymbol{P},\boldsymbol{G})$ the **Betti matching error** between $\boldsymbol{P}$ and $\boldsymbol{G}$. It can be seen as a refinement of the Betti number error, which also takes the location of the features within their respective images into account (see Fig. 8). Since the entries of $\boldsymbol{P}$ and $\boldsymbol{G}$ take values in $\{0,1\}$, the only point appearing in their persistence diagrams is $(0,1)$ and its multiplicity coincides with the number of features in the respective image. Observe that an unmatched point – with respect to the Betti matching – contributes with $2(0-\frac{1}{2})^2 + 2(1-\frac{1}{2})^2 = 1$ to $\mu^{\text{err}}(\boldsymbol{P},\boldsymbol{G})$, while a matched pair of points contributes with 0. Hence, the Betti matching error takes values in $\mathbb{N}_0$ and represents the number of unmatched features in both $\boldsymbol{P}$ and $\boldsymbol{G}$, i.e.,

$$\mu^{\text{err}}(\boldsymbol{P},\boldsymbol{G}) = \#\ker(\mu(\boldsymbol{P},\boldsymbol{G})) + \#\,\text{coker}(\mu(\boldsymbol{P},\boldsymbol{G})). \tag{6}$$

4. Experiments with Betti matching

Datasets We employ a set of six datasets with diverse topological features for our validation experimentation. Two datasets, the Massachusetts roads dataset and the CREMI neuron segmentation dataset, exhibit frequently connected curvilinear, network-like structures, which form a large number of cycles in the foreground. The C.elegans infection live/dead image dataset (Elegans) from the Broad Bioimage Benchmark Collection (Ljosa et al., 2012) and our synthetic, modified MNIST dataset (LeCun, 1998) (synMnist) consist of a balanced number of dimension 0 and dimension 1 features. And third, the colon cancer cell dataset (Colon) from the Broad Bioimage Benchmark Collection (Carpenter et al., 2006; Ljosa et al., 2012) and the Massachusetts buildings dataset (Buildings) (Mnih, 2013) have "blob-like" foreground structures. They contain very few dimension 1 features but every instance of a cell or building forms a dimension 0 feature.

Training of the segmentation networks For implementation details, e.g., the training splits, please refer to App. K and L. We train all our models for a fixed, dataset-specific number of epochs and evaluate the final model on an unseen test set. We train all models on an Nvidia P8000 GPU using Adam optimizer. We run experiments on a range of alpha-parameters for clDice (Shit et al., 2021), the Wasserstein matching (Hu et al., 2019), and Betti matching; we choose to present the top performing model in Table 1; extended results are given in tables 3, 4, 5, 6, 7 in App. J.

4.1. Results

Main Results Our proposed Betti matching loss improves the topological accuracy of the segmentations across all datasets (Table 1), irrespective of the choice of hyperparameters (Table 4) compared to all baselines. We show superior scores for the topological metrics Betti matching error (μ^{err}) and Betti number error (β^{err}) in both dimension 0 and dimension 1. Furthermore, the volumetric metrics

Table 1: Main results for Betti matching and three baselines on six datasets. Green columns indicate the topological metrics. Bold numbers highlight the best performance for a given dataset if it appears substantial (i.e. the second best performance is not within std/8). We find that Betti matching improves the segmentations in all topological metrics for all datasets. We further observe a constantly high performance in volumetric metrics. ↑ indicates higher value wins and ↓ the opposite. More metrics are given in the supplementary Tables.

	Loss	Dice ↑	clDice ↑	Acc. ↑	μ^{err} ↓	μ_0^{err} ↓	μ_1^{err} ↓	β^{err} ↓	β_0^{err} ↓	β_1^{err} ↓
CREMI	Dice	0.894	0.939	0.959	149.64	39.68	109.96	114.12	39.12	75.00
	clDice	0.879	**0.944**	0.952	147.04	34.36	112.68	103.92	33.64	70.28
	Hu et al.	0.888	0.935	0.957	162.48	44.24	118.24	118.16	43.68	74.48
	Ours	0.893	0.941	0.959	**129.80**	**31.00**	**98.80**	**79.16**	**30.36**	**48.80**
Roads	Dice	0.663	0.698	0.974	117.80	87.04	30.76	113.96	86.54	27.42
	clDice	0.668	0.704	0.975	131.00	102.08	28.92	125.83	101.67	24.17
	Hu et al.	0.674	0.712	0.974	101.00	73.04	27.96	95.83	72.54	23.29
	Ours	0.663	0.713	0.972	**83.00**	**56.30**	26.70	**75.08**	**55.79**	**19.29**
synMnist	Dice	0.871	0.907	0.962	3.70	1.96	1.74	2.590	1.674	0.916
	clDice	0.875	0.921	0.963	2.54	0.87	1.67	1.640	0.700	0.940
	Hu et al.	0.866	0.915	0.960	2.85	1.00	1.85	1.802	0.764	1.038
	Ours	0.849	0.915	0.954	**2.28**	**0.53**	1.75	**1.348**	**0.426**	0.922
Elegans	Dice	0.922	0.959	0.984	4.10	2.60	1.50	2.60	1.40	1.20
	clDice	0.917	**0.964**	0.982	3.90	2.20	1.70	2.20	1.20	1.00
	Hu et al.	0.921	0.959	0.984	4.30	2.84	1.45	2.50	1.35	1.15
	Ours	0.919	0.960	0.983	**3.40**	2.10	**1.30**	**1.90**	**0.80**	1.10
Colon	Dice	0.899	0.863	0.970	44.26	21.76	22.50	33.75	13.75	20.00
	clDice	0.907	0.871	0.974	47.26	18.76	28.50	37.75	11.75	26.00
	Hu et al.	0.902	0.876	0.972	34.50	15.50	19.00	22.00	7.00	15.00
	Ours	0.907	0.871	0.975	**32.00**	**14.26**	17.76	21.50	**6.25**	15.25
Buildings	Dice	0.623	0.672	0.934	572.44	551.00	21.46	162.95	151.70	11.25
	clDice	0.632	**0.693**	0.931	571.20	535.96	35.26	175.50	155.05	20.45
	Hu et al.	0.625	0.677	0.934	556.60	537.50	19.10	181.10	169.60	11.50
	Ours	0.625	0.685	**0.937**	**489.16**	**471.26**	17.90	**118.45**	**107.75**	10.70

of the segmentations (Accuracy, Dice, and clDice) show equivalent if not superior quantitative results for our method. Our method can be trained from scratch or used to refine pre-trained networks. Importantly, our method improves the topological correctness of curvilinear segmentation problems (Roads, CREMI), blob-segmentation problems (Buildings, Colon), and mixed problems (SynMnist, Elegans). We confidently attribute this to the theoretical guarantees of induced matchings, which hold for the foreground and the background classes in dim 0 and dim 1. For illustration, please consider the Roads and Buildings dataset; essentially, the topology of the background of the Buildings dataset is very similar to the foreground in Roads, i.e., the foreground of the Roads and the background of the Buildings dataset are interesting in dim 1, whereas the background of the roads and the foreground of the Buildings are interesting in dimension 0. As our method can efficiently leverage the topological features of both foreground and background when we apply *sub- and superlevel set*-matching and it is intuitive that our method prevails in both. It is of note that for some datasets, the method by (Hu et al., 2019) is the best performing baseline and for some (Shit et al., 2021).

Ablation experiments In order to study the effectiveness of the Betti matching loss, we conduct various ablation experiments. First, we study the effect of the α parameter in our method, see Table 4. We find that increasing α improves the topological metrics. For some datasets, e.g., synMnist, the Dice metric is compromised if α is chosen too big. Therefore, we conclude that α is a tunable and dataset-specific parameter. Ostensibly, the effect of the α parameter cannot be compared directly. Nonetheless, it appears that our method is more robust towards variation in α. Second, we study the effect of considering both the foreground and the background (bothlevel) versus solely the foreground (superlevel). We find that bothlevel is particularly useful if the background has a complex topology (e.g., Elegans), whereas superlevel shows a similar performance if the foreground has a more complex topology (e.g., CREMI), see Table 3. Third, we test the effect of pre-training and training from scratch for Betti matching, and the method by (Hu et al., 2019). Table 7 shows that our method can be trained from scratch efficiently, if not superiorly, whereas the baseline method struggles in that setting – especially on more complex datasets such as CREMI. We attribute this to the spatially correct matching of Betti matching and its consequences on the gradient (see Sec. 3.2). Training-from-scratch means that there is a lot of potential for *false positives* and *false negatives* in the Wasserstein matching

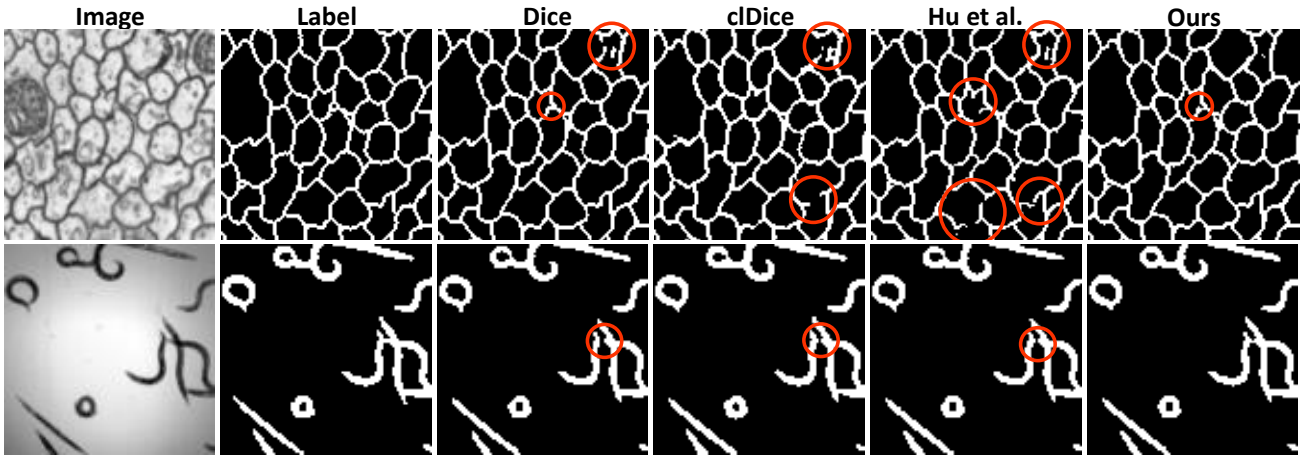


Figure 9: Qualitative Results on the CREMI (top row) and Elegans (bottom row) dataset using the same models as in Table 1. Topological errors are indicated by red circles. Our method leads to less topological errors in the segmentation. For more results, please consider Figures 14-16 in the Appendix.

(see App. G) since there are a lot noisy features when the network is still uncertain. For example, for CREMI, we found that the Wasserstein matching matches cycles incorrectly in more than 99 % of the cases. Moreover, we observe that Betti matching optimizes the Wasserstein loss more efficiently We also experiment with adding a boundary to images in order to close loops that cross the image border, similar to (Hu et al., 2019), and term this **relative Betti matching**. Table 5 shows a negligible effect on all metrics. For additional ablation and more metrics on the ablation studies, please refer to App. J. The computational complexity of Betti matching is $\mathcal{O}(n^3)$, see App. D for details.

5. Discussion

Concluding remarks In this paper, we propose a rigorous method called *Betti matching*, which enables the faithful quantification of topological errors in image segmentations. Herein, our method is the first to guarantee the correct matching of persistence barcodes in image segmentation according to their spatial correspondence. We show that Betti matching error is an interpretable segmentation metric, which can be understood as a sharpened variant of the Betti number error. Further, we show how our method can be used to train segmentation networks. Training networks using Betti matching loss is stable and leads to improvements on all 6 datasets. We foresee vast application potential in challenging tasks such as road network, vascular network and Neuron instance segmentation. We are thus hopeful that our method's theory and experimentation will stimulate future research in this area.

Limitations In the general setting of persistent homology of functions on arbitrary topological spaces, there are instances where maps of persistence modules cannot be written as matchings. This is somewhat analogous to the fact that in linear algebra, certain linear transformations cannot be diagonalized. We did not observe any such case in our specific segmentation setting. A theoretical investigation of this question will be the subject of future work. Further, we understand application-specific experimental limitations. Our method's computational complexity is beyond widely used loss functions such as BCE (see App. D); moreover, our current implementation is only available in 2D, whereas the theoretical guarantees trivially generalize to 3D.

Acknowledgments

The authors gratefully acknowledge Maximilian Schmahl for the initial idea of using induced matchings in the image segmentation settings. The authors are also indebted to Prof. Daniel Rueckert and Ivan Ezhov for all-out support throughout the project. N. Stucki and U. Bauer are supported by the Munich Data Science Institute (MDSI) at Technical University of Munich (TUM) via the MDSI Seed Funds program (ATPL4IS) and by the Munich Center for Machine Learning (MCML). U. Bauer is supported by the Deutsche Forschungsgemeinschaft (DFG – German Research Foundation) – Project-ID 195170736 – TRR1 09. J. C. Paetzold and S. Shit are supported by DFG project DCoMEX 16HPC010 (Grant agreement ID: 956201). B. Menze is supported by the Helmut Horten Foundation.

References

Abousamra, S., Hoai, M., Samaras, D., and Chen, C. Localization in the crowd with topological constraints. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 35, pp. 872–881, 2021.

Bauer, U. Ripser: efficient computation of vietoris–rips persistence barcodes. *Journal of Applied and Computational Topology*, 5(3):391–423, 2021.

Bauer, U. and Lesnick, M. Induced matchings and the algebraic stability of persistence barcodes. *Journal of Computational Geometry*, 6(2):162–191, 2015.

Bauer, U. and Schmahl, M. Efficient computation of image persistence. *arXiv preprint arXiv:2201.04170*, 2022.

Carpenter, A. E., Jones, T. R., Lamprecht, M. R., Clarke, C., Kang, I. H., Friman, O., Guertin, D. A., Chang, J. H., Lindquist, R. A., Moffat, J., et al. Cellprofiler: image analysis software for identifying and quantifying cell phenotypes. *Genome biology*, 7(10):1–11, 2006.

Chazal, F., Cohen-Steiner, D., Glisse, M., Guibas, L. J., and Oudot, S. Y. Proximity of persistence modules and their diagrams. In *Proceedings of the twenty-fifth annual symposium on Computational geometry*, pp. 237–246, 2009a.

Chazal, F., Cohen-Steiner, D., Glisse, M., Guibas, L. J., and Oudot, S. Y. Proximity of persistence modules and their diagrams. In *Proceedings of the twenty-fifth annual symposium on Computational geometry*, pp. 237–246, 2009b.

Chen, C., Freedman, D., and Lampert, C. H. Enforcing topological constraints in random field image segmentation. In *The 24th IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2011, Colorado Springs, CO, USA, 20-25 June 2011*, pp. 2089–2096. IEEE Computer Society, 2011. doi: 10.1109/CVPR.2011.5995503. URL https://doi.org/10.1109/CVPR.2011.5995503.

Cheng, M., Zhao, K., Guo, X., Xu, Y., and Guo, J. Joint topology-preserving and feature-refinement network for curvilinear structure segmentation. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 7147–7156, 2021.

Clough, J., Byrne, N., Oksuz, I., Zimmer, V. A., Schnabel, J. A., and King, A. A topological loss function for deep-learning based image segmentation using persistent homology. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2020.

Cohen-Steiner, D., Edelsbrunner, H., and Harer, J. Stability of persistence diagrams. In *Proceedings of the twenty-first annual symposium on Computational geometry*, pp. 263–271, 2005.

Cohen-Steiner, D., Edelsbrunner, H., Harer, J., and Mileyko, Y. Lipschitz functions have l p-stable persistence. *Foundations of computational mathematics*, 10(2):127–139, 2010.

Crawley-Boevey, W. Decomposition of pointwise finite-dimensional persistence modules. *J. Algebra Appl.*, 14 (5):1550066, 8, 2015. ISSN 0219-4988. doi: 10.1142/S0219498815500668. URL https://doi.org/10.1142/S0219498815500668.

Delgado-Friedrichs, O., Robins, V., and Sheppard, A. Skeletonization and partitioning of digital images using discrete morse theory. *IEEE transactions on pattern analysis and machine intelligence*, 37(3):654–666, 2014.

Dice, L. R. Measures of the amount of ecologic association between species. *Ecology*, 26(3):297–302, 1945. ISSN 00129658, 19399170. URL http://www.jstor.org/stable/1932409.

Edelsbrunner, H., Harer, J., et al. Persistent homology-a survey. *Contemporary mathematics*, 453:257–282, 2008.

Funke, J., Tschopp, F., Grisaitis, W., Sheridan, A., Singh, C., Saalfeld, S., and Turaga, S. C. Large scale image segmentation with structured loss based deep learning for connectome reconstruction. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 41(7):1669–1680, Jul 2019. ISSN 1939-3539. doi: 10.1109/tpami.2018.2835450. URL http://dx.doi.org/10.1109/TPAMI.2018.2835450.

García-Redondo, I., Monod, A., and Song, A. Fast topological signal identification and persistent cohomological cycle matching. Preprint, 2022.

Garin, A., Heiss, T., Maggs, K., Bleile, B., and Robins, V. Duality in persistent homology of images. *arXiv preprint arXiv:2005.04597*, 2020.

Heiss, T. and Wagner, H. Streaming algorithm for euler characteristic curves of multidimensional images. *CoRR*, abs/1705.02045, 2017. URL http://arxiv.org/abs/1705.02045.

Hu, X. and Chen, C. Image segmentation with homotopy warping. *arXiv preprint arXiv:2112.07812*, 2021.

Hu, X., Li, F., Samaras, D., and Chen, C. Topology-preserving deep image segmentation. *Advances in neural information processing systems*, 32, 2019.

Hu, X., Wang, Y., Li, F., Samaras, D., and Chen, C. Topology-aware segmentation using discrete morse theory. In *International Conference on Learning Representations (ICLRR)*, 2021.

Jain, V., Bollmann, B., Richardson, M., Berger, D. R., Helmstaedter, M. N., Briggman, K. L., Denk, W., Bowden, J. B., Mendenhall, J. M., Abraham, W. C., et al. Boundary learning by optimization with topological constraints. In *2010 IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, pp. 2488–2495. IEEE, 2010.

Kaczynski, T., Mischaikow, K., and Mrozek, M. *Cubical Homology*, pp. 39–92. Springer New York, New York, NY, 2004. ISBN 978-0-387-21597-6. doi: 10.1007/0-387-21597-2_2. URL https://doi.org/10.1007/0-387-21597-2_2.

LeCun, Y. The mnist database of handwritten digits. *http://yann. lecun. com/exdb/mnist/*, 1998.

Ljosa, V., Sokolnicki, K. L., and Carpenter, A. E. Annotated high-throughput microscopy image sets for validation. *Nature methods*, 9(7):637–637, 2012.

Mnih, V. *Machine Learning for Aerial Image Labeling*. PhD thesis, University of Toronto, 2013.

Mosinska, A. et al. Beyond the pixel-wise loss for topology-aware delineation. In *CVPR*, pp. 3136–3145, 2018.

Oner, D., Koziński, M., Citraro, L., Dadap, N. C., Konings, A. G., and Fua, P. Promoting connectivity of network-like structures by enforcing region separation. *arXiv preprint arXiv:2009.07011*, 2020.

Otter, N., Porter, M. A., Tillmann, U., Grindrod, P., and Harrington, H. A. A roadmap for the computation of persistent homology. *EPJ Data Science*, 6:1–38, 2017.

Reani, Y. and Bobrowski, O. Cycle registration in persistent homology with applications in topological bootstrap. *IEEE Transactions on Pattern Analysis & Machine Intelligence*, 2022. ISSN 1939-3539. doi: 10.1109/TPAMI.2022.3217443.

Sasaki, K., Iizuka, S., Simo-Serra, E., and Ishikawa, H. Joint gap detection and inpainting of line drawings. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 5725–5733, 2017.

Shit, S., Paetzold, J. C., Sekuboyina, A., Ezhov, I., Unger, A., Zhylka, A., Pluim, J. P., Bauer, U., and Menze, B. H. cldice-a novel topology-preserving loss function for tubular structure segmentation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 16560–16569, 2021.

Wagner, H., Chen, C., and Vuçini, E. Efficient computation of persistent homology for cubical data. In *Topological methods in data analysis and visualization II*, pp. 91–106. Springer, 2012.

Waibel, D. J., Atwell, S., Meier, M., Marr, C., and Rieck, B. Capturing shape information with multi-scale topological loss terms for 3d reconstruction. *arXiv preprint arXiv:2203.01703*, 2022.

Wang, H., Xian, M., and Vakanski, A. Ta-net: Topology-aware network for gland segmentation. In *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision*, pp. 1556–1564, 2022.

Wang, X. and Jiang, X. Post-processing for retinal vessel detection. In *Tenth International Conference on Digital Image Processing (ICDIP 2018)*, volume 10806, pp. 1442–1446. SPIE, 2018.

Yeung, M., Sala, E., Schönlieb, C.-B., and Rundo, L. Unified focal loss: Generalising dice and cross entropy-based losses to handle class imbalanced medical image segmentation. *Computerized Medical Imaging and Graphics*, 95:102026, 2022.

Zhang, H. and Lui, L. M. Topology-preserving segmentation network: A deep learning segmentation framework for connected component. *arXiv preprint arXiv:2202.13331*, 2022.

A. Illustrating additional examples of topological matching performance.

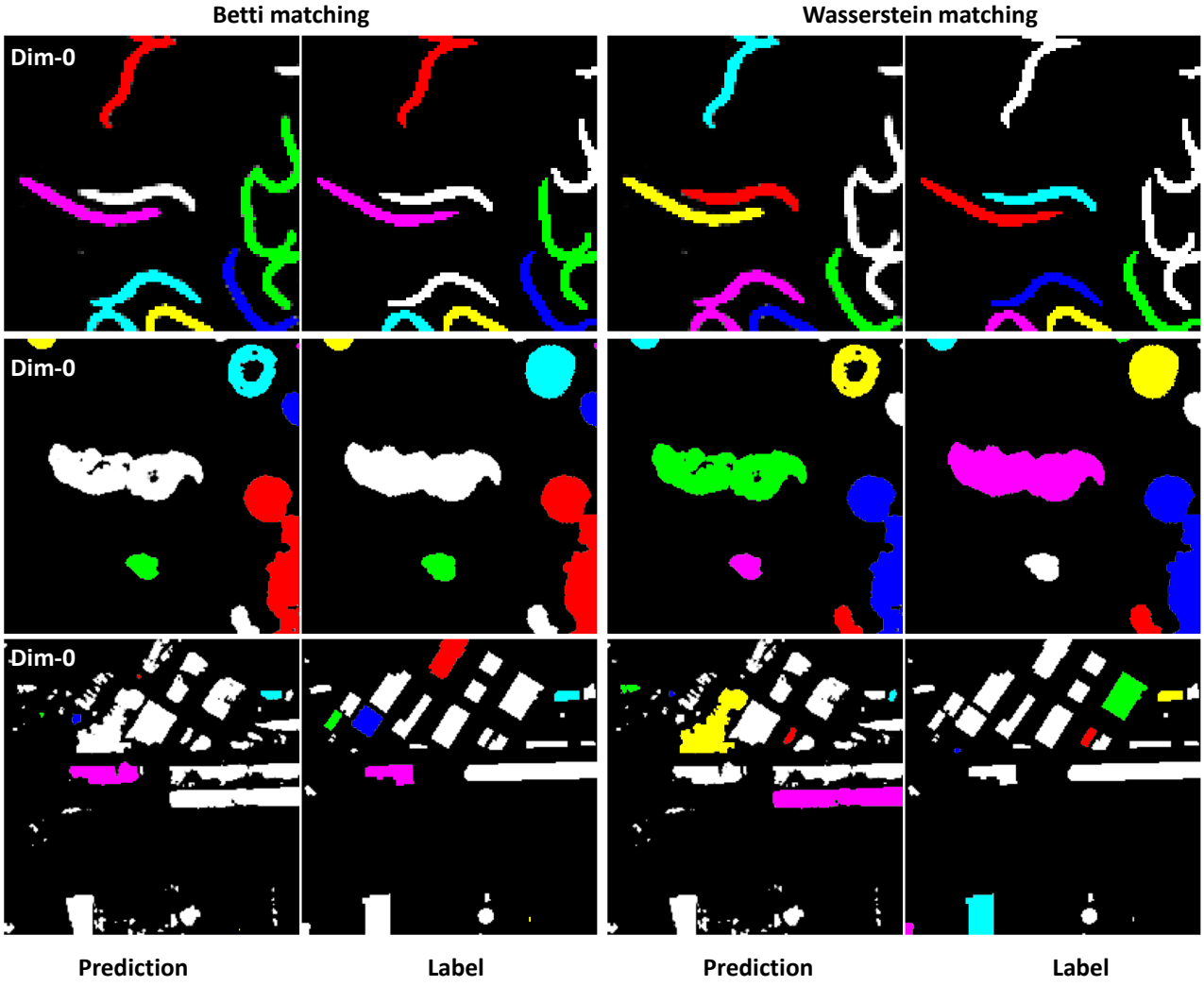


Figure 10: Motivation. Our Betti matching and the Wasserstein matching ((Hu et al., 2019)) for Elegans, Colon and Buildings label-prediction pairs. Here we match the connected components (dim 0). The matched components (according to the matching methods) are represented in the same color. We randomly sample 6 matched pairs.

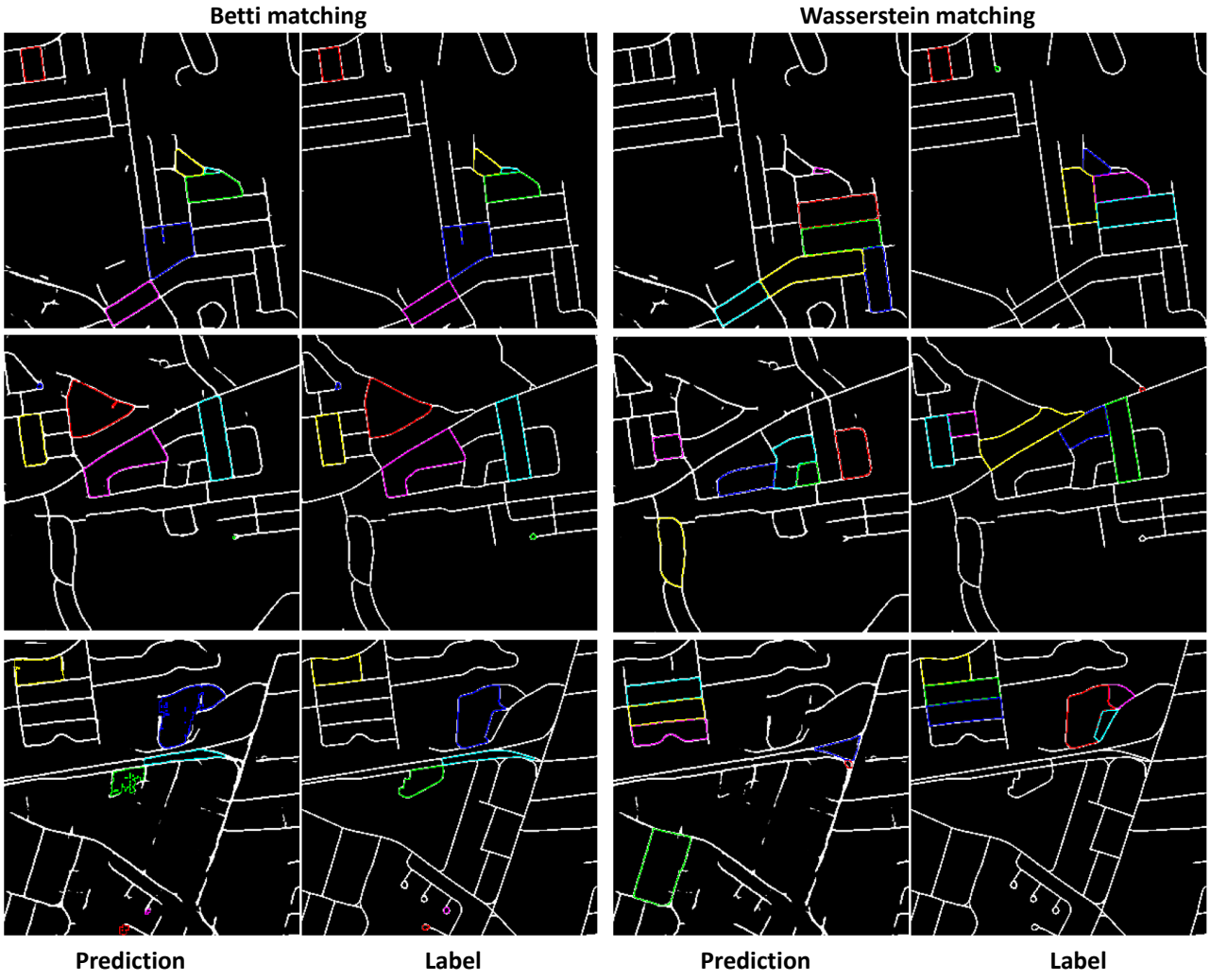


Figure 11: Motivation. Our Betti matching and the Wasserstein matching ((Hu et al., 2019)) for Roads label-prediction pairs. The matched 1-cycles (according to the matching methods) are represented in the same color. We randomly sample 6 matched pairs. We observe that our method correctly matches the cycles in the first two rows. The third row represents an example early in Training. Here we observe that our method correctly matches some ”finished” cycles but also provides a correct matching to the blue and green cycles which still have to be closed. Essentially, one can observe here that our Betti matching leads to a correct loss.

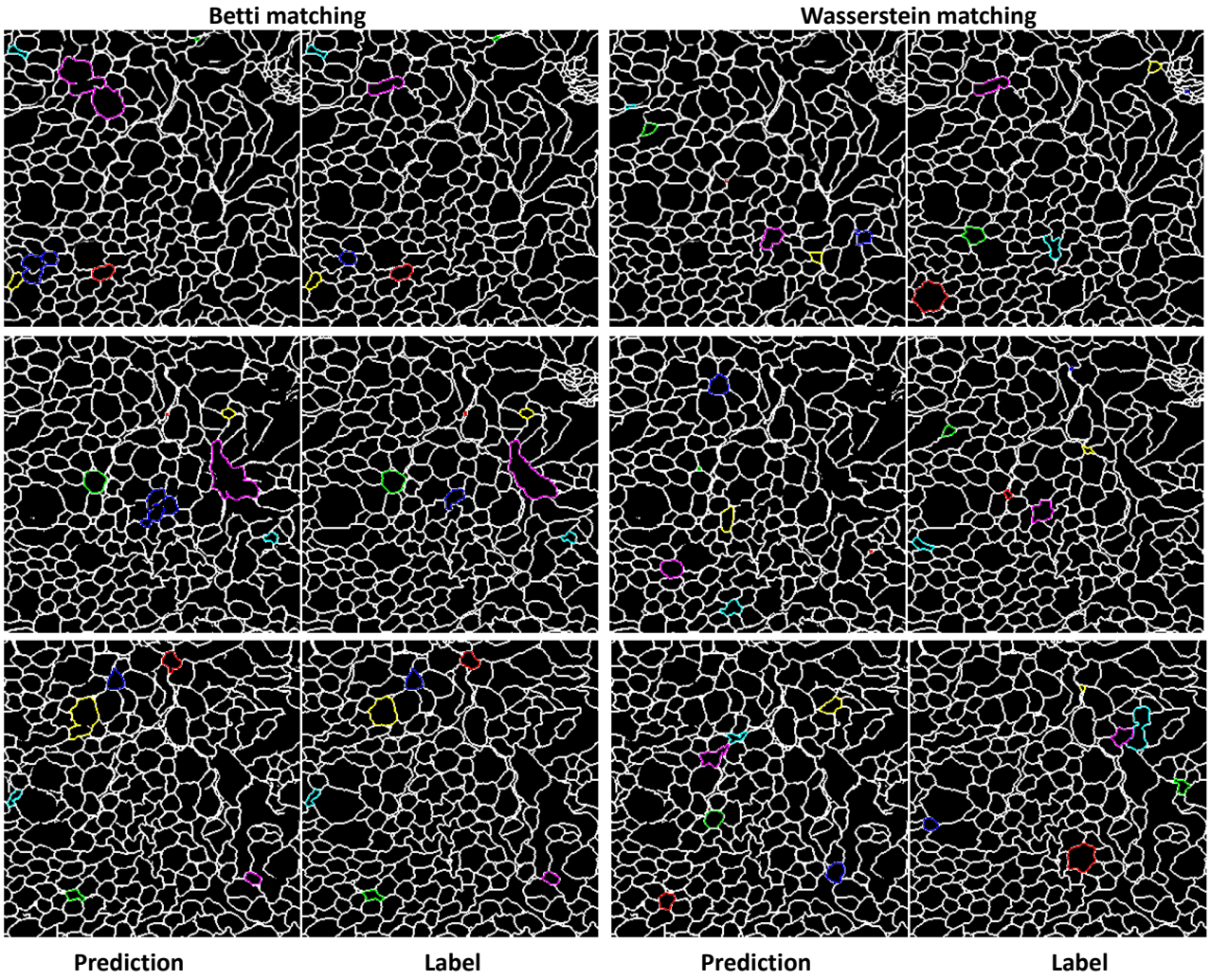


Figure 12: Motivation. Our Betti matching and the Wasserstein matching ((Hu et al., 2019)) for CREMI label-prediction pairs. The matched cycles (according to the matching methods) are represented in the same color. We randomly sample 6 matched pairs.

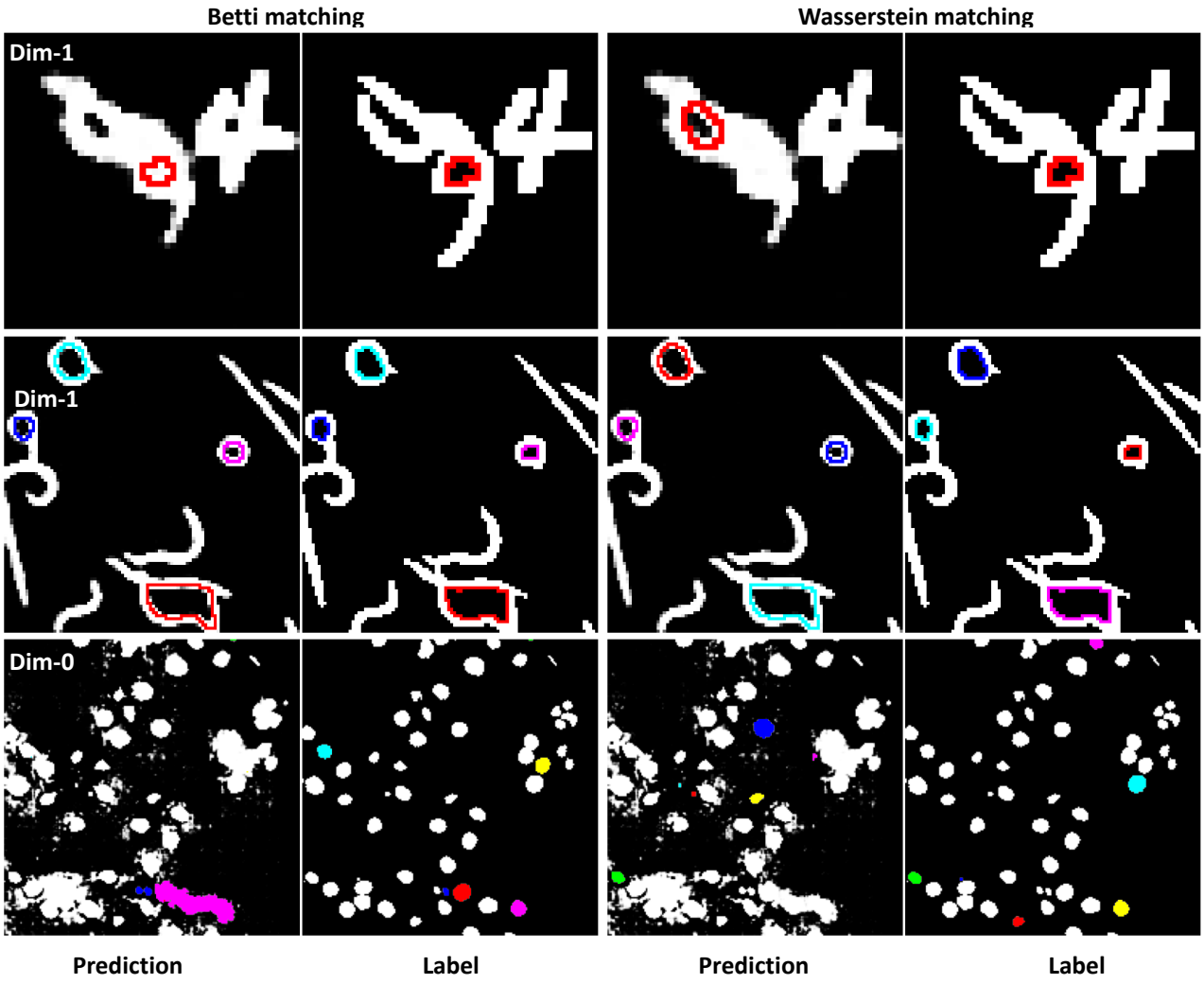


Figure 13: Motivation. Our Betti matching and the Wasserstein matching ((Hu et al., 2019)) for synMnist label-prediction pairs (top row), colon cells (middle row) and the Elegans dataset (lower row). The matched connected components (dim 0) and cycles (dim 1) (according to the matching methods) are represented in the same color. We randomly sample 6 matched pairs. Importantly, in the last row, one limitation of our proposed matching can be observed. Consider the features (buildings) matched by our method in red; here, the matching could be considered suboptimal, because the actual building is poorly segmented, and our method does not match the feature from the label to the largest segmentation component in the prediction. Please note that this scenario is only relevant for really poor predictions.

B. Additional qualitative results

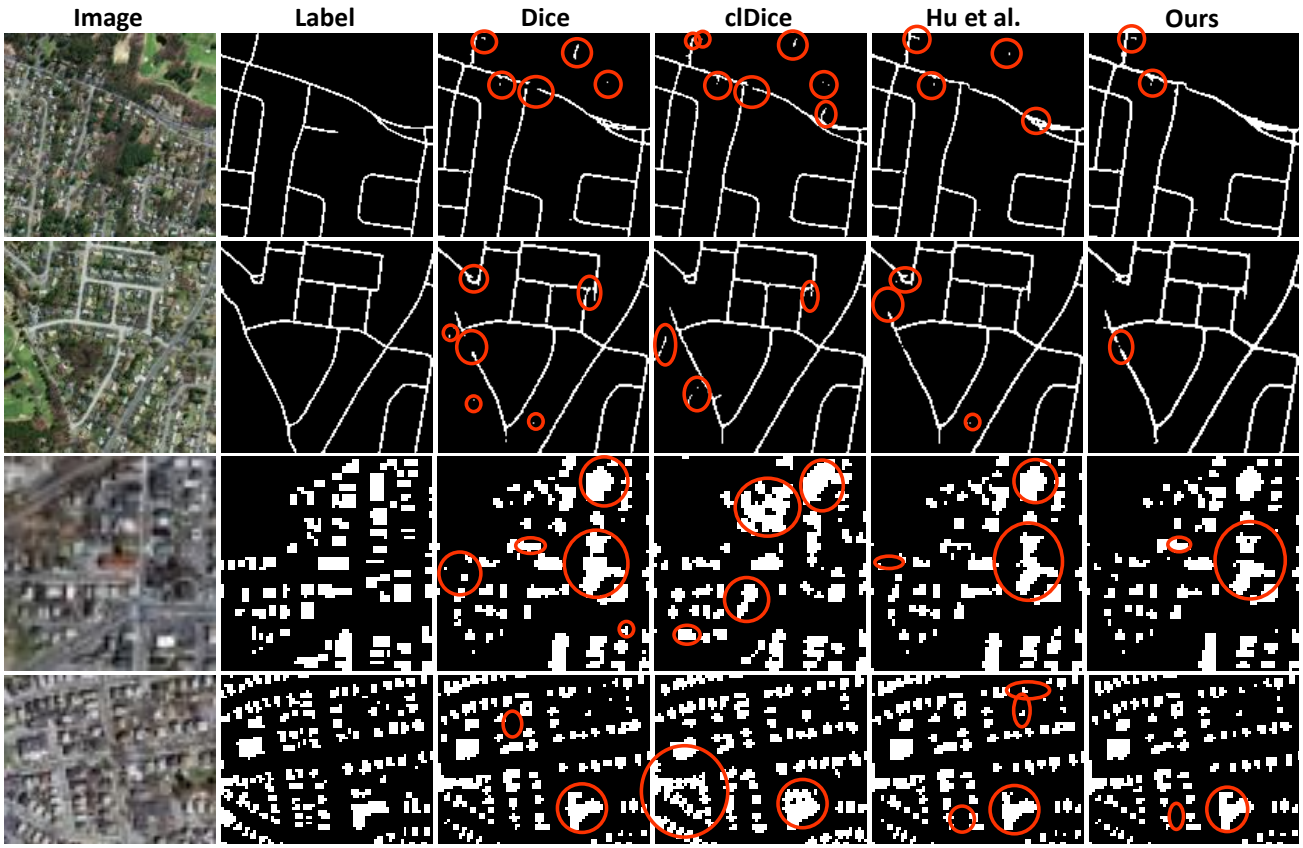


Figure 14: Qualitative Results on Roads and Buildings dataset. Image, Label, and different segmentations (same models as table 1). Topological errors are indicated by red circles. Our method leads to improved topology compared to the baselines.

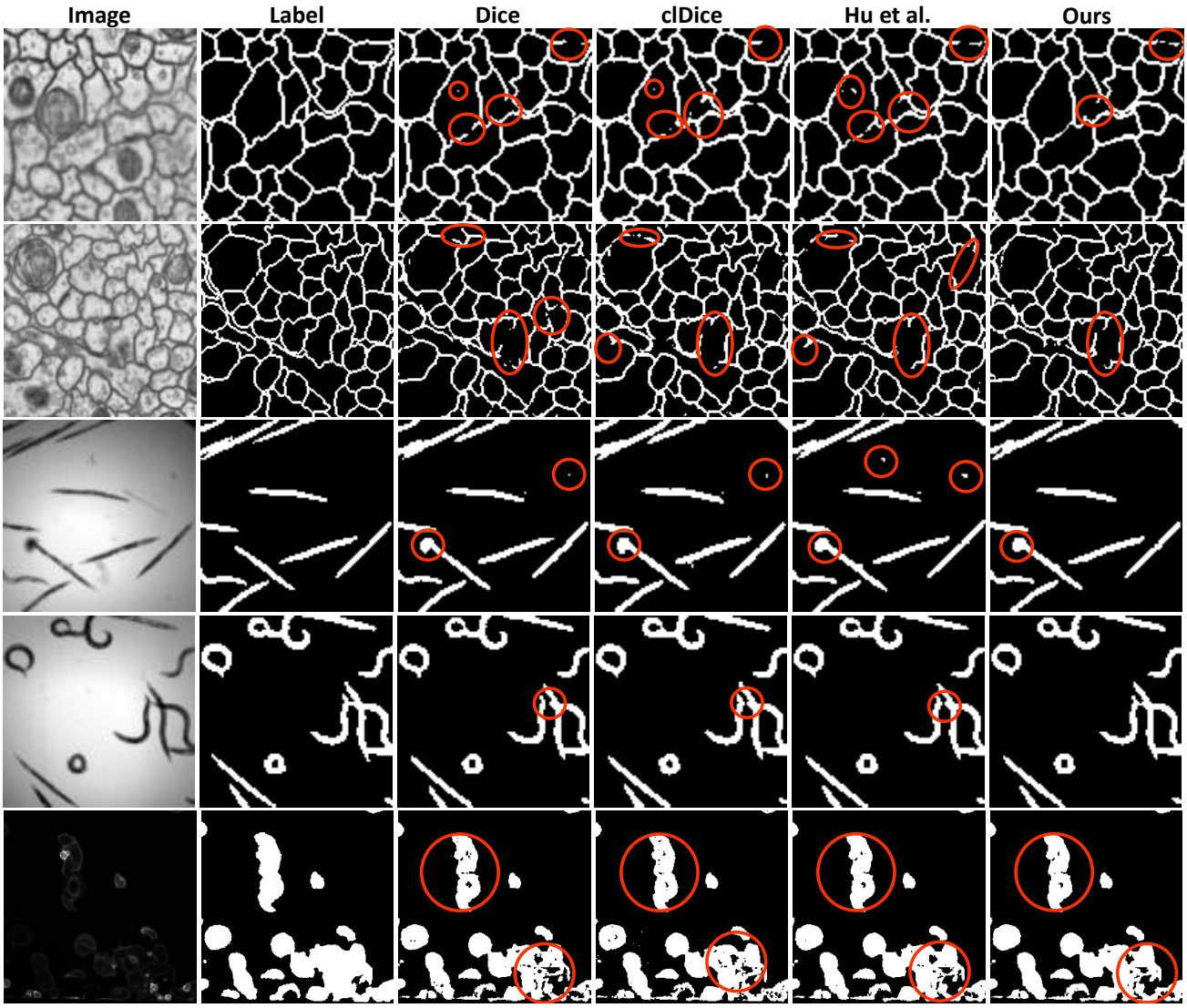


Figure 15: Qualitative Results on CREMI, Elegans and Colon dataset. Image, Label, and different segmentations (same models as table 1). Topological errors are indicated by red circles. Our method leads to improved topology compared to the baselines.

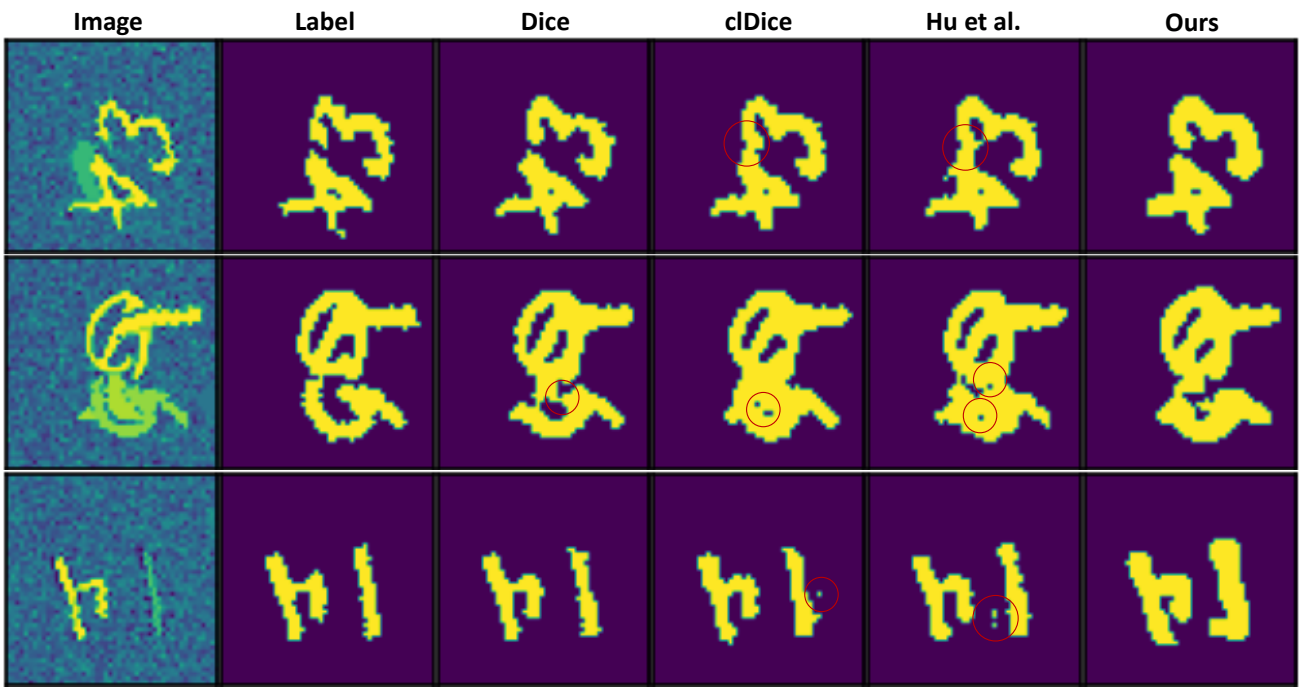


Figure 16: Qualitative Results on SynMnist. Image, Label, and different segmentations (same models as table 1) on examples of the SynMnist testset. Topological errors are indicated by red circles. Our method always segments the correct topology.

C. Implementing Betti matching in an algorithm

Below, we provide the pseudocode for an efficient realization of the Betti matching. For the computation of the barcodes in dimension 0 we leverage the *Union-Find* datastructure, which is very efficient at managing equivalence classes. Alexander duality allows us to use it in dimension 1, as well (see (Garin et al., 2020)). Moreover, it can also be used for the computation of the image barcodes in both dimensions. Note that we adapt the Union Find class to manage the birth of equivalence classes. We use *clearing* (as proposed in (Bauer, 2021)) by keeping track of *critical-edges* and *columns-to-reduce*, in order to reduce the amount of operations during the reductions (see sections 2.3, 2.4).

Algorithm 1: Betti matching

Data: $\boldsymbol{G}, \boldsymbol{L}$
Option: $relative = False, filtration = \text{'}superlevel\text{'}$
Result: $\mathcal{L}_0$, $\mathcal{L}_1$, $\mathcal{L}$

begin
  if *filtration='superlevel'* **then** // Construction of comparison image
    $\boldsymbol{C} \leftarrow max(\boldsymbol{G}, \boldsymbol{L})$
  else
    $\boldsymbol{C} \leftarrow min(\boldsymbol{G}, \boldsymbol{L})$
  end
  $\mathcal{B}(\boldsymbol{G}), \mathbb{D}_G, \boldsymbol{V}_G, \boldsymbol{X}_G \leftarrow CubicalPersistence(\boldsymbol{G}, relative, filtration, True)$;
  $\mathcal{B}(\boldsymbol{L}), \mathbb{D}_L, \boldsymbol{V}_L, \boldsymbol{X}_L \leftarrow CubicalPersistence(\boldsymbol{L}, relative, filtration, True)$;
  $\mathcal{B}(\boldsymbol{C}), \mathbb{C}_C, \boldsymbol{V}_C, \boldsymbol{X}_C \leftarrow CubicalPersistence(\boldsymbol{C}, relative, filtration, False)$;
  $\mathcal{B}(\boldsymbol{G}, \boldsymbol{C}) \leftarrow ImagePersistence(\mathbb{D}_G, \boldsymbol{X}_G, \mathbb{C}_C, \boldsymbol{X}_C)$;
  $\mathcal{B}(\boldsymbol{L}, \boldsymbol{C}) \leftarrow ImagePersistence(\mathbb{D}_L, \boldsymbol{X}_L, \mathbb{C}_C, \boldsymbol{X}_C)$;
  $\sigma(\boldsymbol{G}, \boldsymbol{C}) \leftarrow InducedMatching(\mathcal{B}(\boldsymbol{G}, \boldsymbol{C}), \mathcal{B}(\boldsymbol{G}), \mathcal{B}(\boldsymbol{C}))$;
  $\sigma(\boldsymbol{L}, \boldsymbol{C}) \leftarrow InducedMatching(\mathcal{B}(\boldsymbol{L}, \boldsymbol{C}), \mathcal{B}(\boldsymbol{L}), \mathcal{B}(\boldsymbol{C}))$;
  $\mu(\boldsymbol{L}, \boldsymbol{G}) = \phi$; // Initialize matched refined intervals
  $\mathbb{U}_0, \mathbb{U}_1 = \mathcal{B}(\boldsymbol{G})_0, \mathcal{B}(\boldsymbol{G})_1$; // Initialize unmatched refined intervals for ground truth
  $\mathbb{V}_0, \mathbb{V}_1 = \mathcal{B}(\boldsymbol{L})_0, \mathcal{B}(\boldsymbol{L})_1$; // Initialize unmatched refined intervals for prediction
  $\mathcal{L}_0 = \mathcal{L}_1 = 0$; // Initialize Betti matching loss
  for $d \leftarrow 0$ **to** 1 **by** 1 **do** // Loop over dimension d
    foreach $m_0 \in \sigma(\boldsymbol{G}, \boldsymbol{C})_d$ **do**
      foreach $m_1 \in \sigma(\boldsymbol{L}, \boldsymbol{C})_d$ **do**
        if $m_0[2] = m_1[2]$ **then** // Check for same image persistence pair
          Add $((m_0[0], m_0[2], m_1[0]))$ to $\mu(\boldsymbol{L}, \boldsymbol{G})_d$;
          Remove $(m_0[0])$ from $\mathbb{U}_d$;
          Remove $(m_1[0])$ from $\mathbb{V}_d$;
          Remove (m_1) from $\sigma(\boldsymbol{L}, \boldsymbol{C})_d$;
          $p, q = m_0[0], m_1[0]$;
          $I_0, I_1 = \boldsymbol{V}_G(Index2Coord(p[0])), \boldsymbol{V}_G(Index2Coord(p[1]))$; // Map index to value
          $J_0, J_1 = \boldsymbol{V}_L(Index2Coord(q[0])), \boldsymbol{V}_L(Index2Coord(q[1]))$; // Map index to value
          $\mathcal{L}_d = \mathcal{L}_d + (I_0 - J_0)^2 + (I_1 - J_1)^2$; // Loss for matched intervals
          break
        end
      end
    end
    foreach $p \in \mathbb{U}_d$ **do**
      $I_0, I_1 = \boldsymbol{V}_G(Index2Coord(p[0])), \boldsymbol{V}_G(Index2Coord(p[1]))$; // Map index to value
      $\mathcal{L}_d = \mathcal{L}_d + \frac{(I_0 - I_1)^2}{2}$; // Loss for unmatched intervals in ground truth
    end
    foreach $p \in \mathbb{V}_d$ **do**
      $I_0, I_1 = \boldsymbol{V}_L(Index2Coord(p[0])), \boldsymbol{V}_L(Index2Coord(p[1]))$; // Map index to value
      $\mathcal{L}_d = \mathcal{L}_d + \frac{(I_0 - I_1)^2}{2}$; // Loss for unmatched intervals in prediction
    end
  end
  $\mathcal{L} \leftarrow \mathcal{L}_0 + \mathcal{L}_1$; // Total Betti matching loss
end

Procedure `CubicalPersistence(`$\boldsymbol{I}$*, relative, filtration, critical*`)`

if *relative=True* **then**
  $\boldsymbol{I} \leftarrow AddBoundary(\boldsymbol{I})$; `// Add image boundary`
end
$\boldsymbol{V}, \boldsymbol{X}, \mathbb{E} \leftarrow FilterCubeMap(\boldsymbol{I}, filtration)$; `// Valuemap, Indexmap & edges are computed using the CubeMap datastructure as in (Wagner et al., 2012)`
$\mathcal{B}(\boldsymbol{I})_0, \mathcal{B}(\boldsymbol{I})_1 = \phi$; `// Initialize refined barcodes`
$\mathbb{C} = \phi$; `// Initialize columns-to-reduce for the clearning trick`
if *critical=True* **then**
  $\mathbb{D} = \phi$; `// Initialize critical-edges for the clearing trick`
end
$\mathcal{U} = UnionFind(\#cubes + 1)$; `// Instantiate a Union-Find class`
foreach $e \in \mathbb{E}$ **do** `// Compute refined intervals in dimension 1`
  $b_0, b_1 \leftarrow DualBoundary(\boldsymbol{X}, e)$; `// Find dual boundary of an edge`
  $x, y \leftarrow \mathcal{U}.find(b_0), \mathcal{U}.find(b_1)$;
  if $x = y$ **then**
    Add e to $\mathbb{C}$, **continue**
  end
  $b = min(\mathcal{U}.getbirth(x), \mathcal{U}.getbirth(y))$; `// Retrieve birth`
  if *critical=True* **then**
    Add e to $\mathbb{D}$;
  end
  if (e, b) *is valid* **then** `// Check for positive interval`
    Add (e, b) to $\mathcal{B}(\boldsymbol{I})_1$
  end
  $\mathcal{U}.union(x, y)$
end
$\mathcal{U} = UnionFind(\#cubes)$; `// Instantiate a Union-Find class`
foreach $e \in \mathbb{C}$ **do** `// Compute refined intervals in dimension 0`
  $b_0, b_1 \leftarrow Boundary(\boldsymbol{X}, e)$; `// Find boundary of an edge`
  $x, y \leftarrow \mathcal{U}.find(b_0), \mathcal{U}.find(b_1)$;
  if $x = y$ **then**
    continue
  end
  $b = max(\mathcal{U}.getbirth(x), \mathcal{U}.getbirth(y))$; `// Retrieve birth`
  if (b, e) *is valid* **then** `// Check for positive interval`
    Add (b, e) to$\mathcal{B}(\boldsymbol{I})_0$;
  end
  $\mathcal{U}.union(x, y)$
end
if *critical=True* **then**
  return $(\mathcal{B}(\boldsymbol{I})_0, \mathcal{B}(\boldsymbol{I})_1), \mathbb{D}, \boldsymbol{V}, \boldsymbol{X}$; `// Return refined barcodes, critical-edges, Valuemap & Indexmap`
else
  return $(\mathcal{B}(\boldsymbol{I})_0, \mathcal{B}(\boldsymbol{I})_1), \mathbb{C}, \boldsymbol{V}, \boldsymbol{X}$; `// Return refined barcodes, columns-to-reduce, Valuemap & Indexmap`
end

Procedure `ImagePersistence` $(\mathbb{D}, \boldsymbol{X}_I, \mathbb{C}, \boldsymbol{X}_J)$
- $\mathcal{B}(\boldsymbol{I},\boldsymbol{J})_0, \mathcal{B}(\boldsymbol{I},\boldsymbol{J})_1 = \phi$; // Initialize image persistence pairs
- $\mathcal{U} = UnionFind(\#cubes)$; // Instantiate a Union-Find class
- **foreach** $e \in \mathbb{C}$ **do** // Compute pairs in dimension 0
  - $b_0, b_1 \leftarrow Boundary(\boldsymbol{X}_I, e)$; // Find boundary of an edge
  - $x, y \leftarrow \mathcal{U}.find(b_0), \mathcal{U}.find(b_1)$;
  - **if** $x = y$ **then**
    - **continue**
  - **end**
  - $b = max(\mathcal{U}.getbirth(x), \mathcal{U}.getbirth(y))$; // Retrieve birth
  - Add (b, e) to $\mathcal{B}(\boldsymbol{I},\boldsymbol{J})_0$; `// All pairs for extended induced matching (see Sec. 2.4)`
  - $\mathcal{U}.union(x, y)$
- **end**
- $\mathcal{U} = UnionFind(\#cubes + 1)$; // Instantiate a Union-Find class
- **foreach** $e \in \mathbb{D}$ **do** // Compute pairs in dimension 1
  - $b_0, b_1 \leftarrow DualBoundary(\boldsymbol{X}_J, e)$; // Find dual boundary of an edge
  - $x, y \leftarrow \mathcal{U}.find(b_0), \mathcal{U}.find(b_1)$;
  - **if** $x = y$ **then**
    - **continue**
  - **end**
  - $b = min(\mathcal{U}.getbirth(x), \mathcal{U}.getbirth(y))$; // Retrieve birth
  - Add (e, b) to $\mathcal{B}(\boldsymbol{I},\boldsymbol{J})_1$; `// All pairs for extended induced matching (see Sec. 2.4)`
  - $\mathcal{U}.union(x, y)$
- **end**
- **return** $(\mathcal{B}(\boldsymbol{I},\boldsymbol{J})_0, \mathcal{B}(\boldsymbol{I},\boldsymbol{J})_1)$; // Return image persistence pairs

Procedure `InducedMatching` $(\mathcal{B}(\boldsymbol{I},\boldsymbol{J}), \mathcal{B}(\boldsymbol{I}), \mathcal{B}(\boldsymbol{J}))$
- $\sigma(\boldsymbol{I},\boldsymbol{J})_0, \sigma(\boldsymbol{I},\boldsymbol{J})_1 = \phi$; // Initialize matched refined intervals
- **for** $d \leftarrow 0$ **to** 1 **by** 1 **do** // Loop over dimension d
  - **foreach** $(a, b) \in \mathcal{B}(\boldsymbol{I},\boldsymbol{J})_d$ **do** // For each image persistence pair
    - $m_i, m_j = None$;
    - **foreach** $(c, d) \in \mathcal{B}(\boldsymbol{I})_d$ **do** // Match left endpoints
      - **if** $c = a$ **then**
        - $m_i = (c, d)$;
        - **break**
      - **end**
    - **end**
    - **if** $m_i = None$ **then** // Skip search if no match found
      - **continue**
    - **end**
    - **foreach** $(c, d) \in \mathcal{B}(\boldsymbol{J})_d$ **do** // Match right endpoints
      - **if** $d = b$ **then**
        - $m_j = (c, d)$;
        - **break**
      - **end**
    - **end**
    - **if** $m_j = None$ **then** // Skip search if no match found
      - **continue**
    - **end**
    - Add $(m_i, (a, b), m_j)$ to $\sigma(\boldsymbol{I},\boldsymbol{J})_d$;
  - **end**
- **end**
- **return** $(\sigma(\boldsymbol{I},\boldsymbol{J})_0, \sigma(\boldsymbol{I},\boldsymbol{J})_1)$

D. Computational complexity

For a grayscale image represented by a matrix $\boldsymbol{I} \in \mathbb{R}^{M,N}$, we have $n = MN$ number of pixels and form a cubical grid complex of dimension $d = 2$. The computation of the filtration and the boundary matrix can be done efficiently using the CubeMap data structure (see (Wagner et al., 2012)) with $\mathcal{O}(3^d n + d^2 n)$ time and $\mathcal{O}(d^2 n)$ space complexity. Computing the barcodes by means of the reduction algorithm requires cubic complexity in the number of pixels $\mathcal{O}(n^3)$ (see (Otter et al., 2017)). Despite our empirical acceleration due to the Union-Find class and clearing tricks (as described in (Bauer & Schmahl, 2022; Bauer, 2021)), the order complexity remains $\mathcal{O}(n^3)$. We need $\mathcal{O}(n^2)$ time complexity for computing the final matching and loss. It is noteworthy that (Hu et al., 2019) also needs $\mathcal{O}(n^3)$ time complexity to compute the barcode and $\mathcal{O}(n^2)$ for the matching, whereas (Shit et al., 2021) requires relatively lower complexity $\mathcal{O}(n)$ due to the overlap based loss formulation.

E. Runtime Experiments

We have added experiments that show the runtime of our training and the calculation of the metric compared to the baseline of Hu et al. and the Betti number metric.

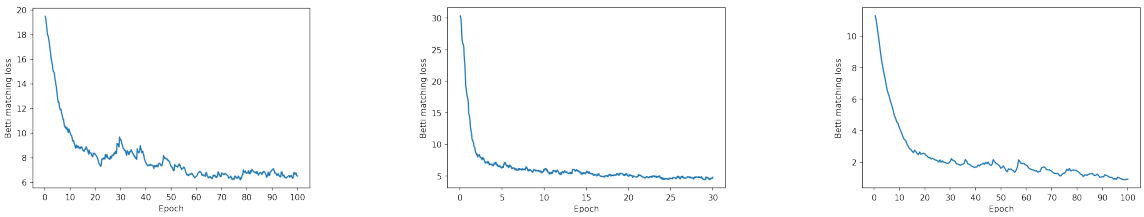
Loss Computation	Runtime (Training)
Hu et al.	5.38 seconds per iteration
Ours Betti matching	7.24 seconds per iteration

We report the loss computation time per iteration for a batch size of 8 on a Quadro P6000.

Metric Computation	Runtime (Evaluation)
Betti number error	20.55 seconds per image
Ours Betti matching	35.25 seconds per image

We report the metric computation time for an image size of 312x312 on a Quadro P6000.

F. Convergence of Betti matching loss



(a) CREMI *(b)* synMnist *(c)* ELEGANS

Figure 17: Plot of the empirical convergence curves of our Betti matching loss for the CREMI, MNIST, and ELEGANS datasets. We plot the Betti matching contribution in the training loss for a varying number of epochs, which is dependent on the dataset size. Please note that we train our model on 48×48 size image patch. We show that Betti matching loss efficiently converges for the different datasets. The absolute magnitude of the loss varies from dataset to dataset because Betti matching is a real interpretable measure of dim 0 and dim 1 topological features in the training images. For example, CREMI has a substantially higher number of features, especially cycles, than Elegans, therefore, the absolute magnitude of the loss is likely higher.

G. Wasserstein matching

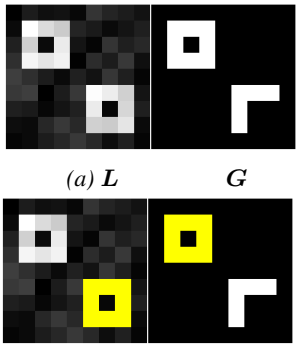
The pth **Wasserstein distance** is frequently used to measure the difference between persistence diagrams; it is given by

$$d_p(\mathcal{B}_1, \mathcal{B}_2) = \inf_{\gamma} \left(\sum_{q \in \mathrm{Dgm}(\mathcal{B}_1)} \|q - \gamma(q)\|_\infty^p \right)^{1/p}$$

for $p \geq 1$, where γ runs over all bijections $\mathrm{Dgm}(\mathcal{B}_1) \to \mathrm{Dgm}(\mathcal{B}_2)$ that respect the dimension. For a likelihood map $\boldsymbol{L} \in [0,1]^{m \times n}$ and a ground truth $\boldsymbol{G} \in \{0,1\}^{m \times n}$, the authors of (Hu et al., 2019) adopt this metric to define the **Wasserstein loss**

$$l_{\mathrm{W}}(\boldsymbol{L}, \boldsymbol{G}) = \min_{\gamma} \sum_{q \in \mathrm{Dgm}(\boldsymbol{L})} \|q - \gamma(q)\|_2^2,$$

where γ runs over all bijections $\mathrm{Dgm}(\boldsymbol{L}) \to \mathrm{Dgm}(\boldsymbol{G})$ that respect the dimension. The bijection γ_* achieving the minimum corresponds to the **Wasserstein matching** $\mathrm{Dgm}(\boldsymbol{L}) \to \mathrm{Dgm}(\boldsymbol{G})$, which minimizes the total distance of matched points. For the represented topological features this means that the matching is purely based on their local contrast within their respective images. Furthermore, note that $\mathrm{Dgm}(\boldsymbol{G})$ contains exclusively the point $(0,1)$ since the entries of $\boldsymbol{G}$ are contained in $\{0,1\}$. Hence, γ_* matches points in $\mathrm{Dgm}(\boldsymbol{L})$ representing features in $\boldsymbol{L}$ with enough local contrast in descending order until $\mathrm{Dgm}(\boldsymbol{G})$ runs out of points. This procedure results in a matching of topological features, which potentially exhibit no spatial relation within their respective images (see Fig. 2,8b,18b and App. A) and can have a negative impact on the training of segmentation networks. To see this, we distinguish two cases for a fixed point $q = (q_1, q_2) \in \mathrm{Dgm}(\boldsymbol{L})$:



(a) $\boldsymbol{L}$ $\boldsymbol{G}$

(b) Wasserstein matching

Figure 18: (a) A predicted likelihood map $\boldsymbol{L}$ and ground truth segmentation $\boldsymbol{G}$. (b) visualizes the Wasserstein matching γ_* (only the yellow cycles are matched), i.e. the top-left cycle in $\boldsymbol{L}$ is a false negative and the bottom-right cycle in $\boldsymbol{L}$ is a false positive.

case 1: (**false positive**) q is matched but there is no spatially corresponding feature in $\boldsymbol{G}$:
Since q is matched to the point $(0,1) \in \mathrm{Dgm}(\boldsymbol{G})$, the loss l_{W} will be reduced by decreasing the value q_1 and increasing the value q_2. Hence, the segmentation network will learn to increase the local contrast of the feature described by the q (see Sec. 3.2), but it should be decreased.

case 2: (**false negative**) q is unmatched but there is a spatially corresponding feature in $\boldsymbol{G}$:
Since q is unmatched, the bijection γ_* maps it to its closest point $((q_1+q_2)/2, (q_1+q_2)/2)$ on the diagonal Δ and the loss l_{W} will be reduced by increasing the value q_1 and decreasing the value q_2. Hence, the segmentation network will learn to decrease the local contrast of the feature described by q (see Sec. 3.2), but it should be increased.

G.1. Frequency of incorrect Wasserstein matching

Next, we study how frequently these two cases occur. Assuming that the Betti matching is correct, we evaluate the quality of the Wasserstein matching on the CREMI dataset. Therefor, we choose a segmentation model to obtain label-prediction pairs for every image in the CREMI dataset and compute both matchings. Among the 37243 matched intervals in the barcodes of the predictions by the Wasserstein matching, only 224 have been matched correctly, i.e. it achieves a *precision* of 0.6%.

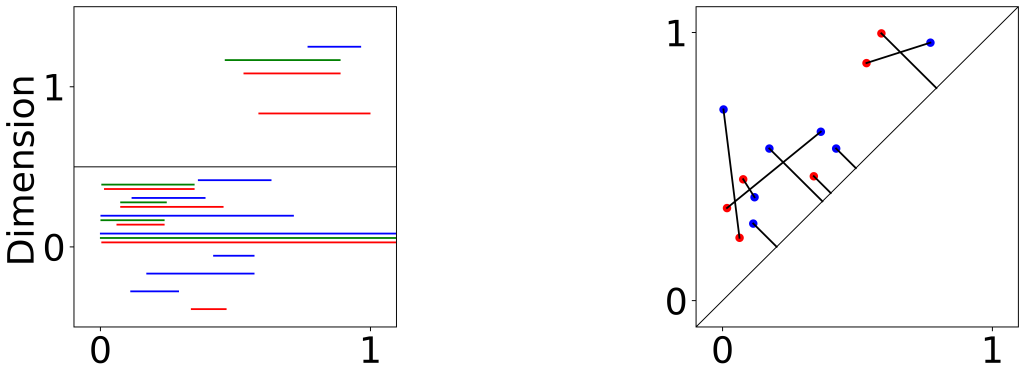
G.2. Wasserstein loss as Betti number error

For a binarized output $\boldsymbol{P}$ and ground truth $\boldsymbol{G}$, the Wasserstein loss and the Betti number error are closely related. A similar argumentation as in Sec. 3.3 for the Betti matching loss shows that

$$\beta^{\mathrm{err}}(\boldsymbol{P}, \boldsymbol{G}) = 2 l_{\mathrm{W}}(\boldsymbol{P}, \boldsymbol{G}).$$

A lower Betti number error of a model trained with our Betti matching loss compared to a model trained with the Wasserstein loss asserts that the Betti matching loss produces more *faithful* gradients during the training of segmentation networks. Note that, empirically, models trained with Betti matching loss consistently outperform models trained with Wasserstein loss with regard to the Betti number error (see Tables 1,4).

H. Persistence Diagrams and Barcodes



(a) Matching between Barcodes

(b) Bijection between persistence diagram

Figure 19: Illustrations of how to translate a matching between barcodes (a) into a bijection between persistence diagram (b) and vice versa. A red or blue line in (a) is a dot of the same color in (b). In (a), a green interval in between a blue and a red interval indicates that they are matched. In (b), a line connecting two points indicates that they are matched. For detail, please refer to Section 3.2.

I. Additional comparison experiments

Table 2: We find that our method outperforms the unified focal loss proposed by Weung et al. (Yeung et al., 2022) in all metrics on CREMI dataset. It is important to note that the main contribution of our method is the faithful matching of topological features between label and prediction. This sets us apart from all voxel based losses designed to handle class imbalance.

Methods	Dice	clDice	Acc.	μ^{err}	μ_0^{err}	μ_1^{err}	β^{err}	β_0^{err}	β_1^{err}
Yeung et al.	0.867	0.926	0.949	472.16	152.88	319.28	184.72	75.88	108.84
Ours (Betti matching)	0.893	0.941	0.959	259.60	62.00	197.60	79.16	30.36	48.80

J. Additional ablation experiments

Table 3: bothlevel versus superlevel matching of our method on the Elegans dataset and the CREMI dataset. The bothlevel matching appears to have a more pronounced contribution in the scenario of topologically complex background

level	α	Dice	clDice	Acc.	μ^{err}	μ_0^{err}	μ_1^{err}	β^{err}	β_0^{err}	β_1^{err}
Elegans bothlevel	0.005	0.92	0.96	0.98	3.40	2.10	1.30	1.90	0.80	1.10
Elegans superlevel	0.005	0.92	0.95	0.98	4.30	2.70	1.60	2.40	1.30	1.10
CREMI bothlevel	0.5	0.89	0.95	0.95	120.96	25.84	95.12	52.08	25.28	26.80
CREMI superlevel	0.5	0.89	0.95	0.96	118.40	28.80	89.60	52.24	28.16	24.08

Table 4: α ablation on the synMnist dataset and the Roads dataset.

		α	Dice	clDice	Acc.	μ^{err}	μ_0^{err}	μ_1^{err}	β^{err}	β_0^{err}	β_1^{err}	ARI	VOI
Roads	Ours	0.0005	0.670	0.706	0.974	107.92	79.79	28.13	103.917	79.375	24.542	0.643	0.847
		0.005	0.670	0.708	0.974	102.50	74.54	27.96	97.583	74.042	23.542	0.647	0.839
		0.05	0.667	0.709	0.974	90.79	63.33	27.46	85.042	62.833	22.208	0.655	0.828
		0.5	0.663	0.713	0.972	83.00	56.29	26.71	75.083	55.792	19.292	0.690	0.791
	clDice	0.05	0.664	0.701	0.975	154.58	123.38	31.21	151.250	123.125	28.125	0.588	0.895
		0.1	0.663	0.697	0.975	162.38	131.96	30.42	158.375	131.708	26.667	0.599	0.885
		0.25	0.667	0.701	0.975	158.38	128.25	30.13	154.208	127.917	26.292	0.622	0.879
		0.75	0.668	0.704	0.975	131.00	102.08	28.92	125.833	101.667	24.167	0.615	0.873
		0.5	0.663	0.696	0.975	157.25	127.50	29.75	152.417	127.000	25.417	0.618	0.874
	Hu et al.	0.0005	0.669	0.706	0.974	107.50	79.04	28.46	102.500	78.542	23.958	0.651	0.838
		0.005	0.674	0.712	0.974	101.00	73.04	27.96	95.833	72.542	23.292	0.660	0.836
		0.05	0.669	0.707	0.974	105.79	78.42	27.38	99.875	77.917	21.958	0.654	0.832
		0.5	0.656	0.699	0.970	117.17	89.21	27.96	105.417	88.708	16.708	0.709	0.787
synMnist	Ours	0.0005	0.866	0.907	0.962	3.37	1.69	1.69	2.302	1.370	0.932	0.844	0.537
		0.005	0.871	0.920	0.962	2.54	0.92	1.62	1.596	0.732	0.864	0.873	0.481
		0.05	0.849	0.916	0.955	2.28	0.53	1.75	1.348	0.426	0.922	0.868	0.491
		0.5	0.796	0.888	0.939	2.30	0.58	1.72	1.428	0.466	0.962	0.805	0.612
	clDice	0.05	0.871	0.911	0.963	3.23	1.59	1.64	2.264	1.328	0.936	0.871	0.483
		0.1	0.872	0.912	0.963	3.28	1.63	1.65	2.320	1.384	0.936	0.862	0.506
		0.25	0.874	0.917	0.963	2.68	1.04	1.65	1.764	0.826	0.938	0.877	0.461
		0.5	0.875	0.922	0.963	2.54	0.87	1.67	1.640	0.700	0.940	0.881	0.454
		0.75	0.869	0.921	0.961	2.39	0.80	1.59	1.580	0.622	0.958	0.888	0.429
	Hu et al.	0.0005	0.872	0.909	0.963	3.76	2.00	1.76	2.650	1.686	0.964	0.880	0.461
		0.005	0.870	0.908	0.962	3.57	1.82	1.75	2.498	1.514	0.984	0.864	0.504
		0.05	0.867	0.916	0.960	2.85	1.00	1.85	1.802	0.764	1.038	0.893	0.425
		0.5	0.785	0.862	0.935	3.51	1.12	2.38	1.968	0.840	1.128	0.814	0.589

Table 5: Relative Frame ablation of our method on the Roads dataset

	α	Dice	clDice	Acc.	μ^{err}	μ_0^{err}	μ_1^{err}	β^{err}	β_0^{err}	β_1^{err}	ARI	VOI
relative	0.0005	0.670	0.706	0.974	107.92	79.79	28.13	103.917	79.375	24.542	0.643	0.847
	0.005	0.670	0.708	0.974	102.50	74.54	27.96	97.583	74.042	23.542	0.647	0.839
	0.05	0.667	0.709	0.974	90.79	63.33	27.46	85.042	62.833	22.208	0.655	0.828
	0.5	0.663	0.713	0.972	83.00	56.29	26.71	75.083	55.792	19.292	0.690	0.791
non-relative	0.0005	0.669	0.706	0.974	109.46	81.04	28.42	104.542	80.542	24.000	0.654	0.835
	0.005	0.671	0.709	0.974	100.50	72.96	27.54	95.167	72.458	22.708	0.661	0.829
	0.005	0.669	0.712	0.973	92.21	64.54	27.67	84.708	64.042	20.667	0.675	0.818
	0.5	0.661	0.711	0.972	82.83	56.33	26.50	75.583	55.833	19.750	0.695	0.787

Table 6: Dimension 1 and dimensions 0,1 matching ablation for the Hu et al. method on the Roads dataset

	α	Dice	clDice	Acc.	μ^{err}	μ_0^{err}	μ_1^{err}	β^{err}	β_0^{err}	β_1^{err}	ARI	VOI
dim 1	0.0005	0.669	0.706	0.974	107.50	79.04	28.46	102.500	78.542	23.958	0.651	0.838
	0.005	0.674	0.712	0.974	101.00	73.04	27.96	95.833	72.542	23.292	0.660	0.836
	0.05	0.669	0.707	0.974	105.79	78.42	27.38	99.875	77.917	21.958	0.654	0.832
	0.5	0.656	0.699	0.970	117.17	89.21	27.96	105.417	88.708	16.708	0.709	0.787
dim 0,1	0.0005	0.672	0.709	0.974	108.58	80.21	28.38	104.083	79.708	24.375	0.649	0.839
	0.005	0.673	0.710	0.974	105.50	77.25	28.25	100.667	76.750	23.917	0.656	0.836
	0.05	0.668	0.708	0.974	94.00	66.29	27.71	88.083	65.792	22.292	0.649	0.832
	0.5	0.662	0.711	0.972	89.42	62.58	26.83	80.583	62.083	18.500	0.692	0.798

Table 7: Pretraining (denoted with *) vs. training from scratch (denoted without *) of ours and the Hu et al. method on the Elegans dataset.

	Training	α	Dice	clDice	Acc.	μ^{err}	μ_0^{err}	μ_1^{err}	β^{err}	β_0^{err}	β_1^{err}	ARI	VOI
CREMI	Ours	0.05	0.882	0.938	0.953	130.12	27.88	102.24	45.72	27.16	18.56	0.919	0.393
	Ours*	0.05	0.889	0.940	0.957	131.36	28.52	102.84	64.40	27.96	36.44	0.905	0.437
	Hu et al.	0.05	0.880	0.932	0.953	165.52	55.84	109.68	85.60	55.28	30.32	0.905	0.436
	Hu et al.*	0.05	0.895	0.942	0.960	132.80	35.52	97.28	75.36	34.96	40.40	0.909	0.425
Elegans	Ours	0.005	0.919	0.960	0.983	3.40	2.10	1.30	1.90	0.80	1.10	0.927	0.359
	Ours*	0.005	0.924	0.963	0.984	3.90	2.40	1.50	2.30	1.20	1.10	0.939	0.313
	Hu et al.	0.005	0.921	0.959	0.984	4.30	2.85	1.45	2.50	1.35	1.15	0.929	0.350
	Hu et al.*	0.005	0.921	0.962	0.984	4.15	2.50	1.65	2.55	1.30	1.25	0.929	0.349

K. Datasets and training splits

The full training routine with the complete trainingsets and testsets will be available with our github repository [2]. All our trainings are done on patches of 48×48 pixels. For the buildings dataset (Mnih, 2013), we downsample the images to 375×375 pixels and randomly choose 80 samples for training and 20 for testing. For each epoch, we randomly sample 8 patches from each sample. For the Colon dataset (Carpenter et al., 2006; Ljosa et al., 2012), we downsample the images to 256×256 pixels; we randomly choose 20 samples for training and 4 for testing. For each epoch, we randomly sample 12 patches from each sample. For the CREMI dataset (Funke et al., 2019), we downsample the images to 312×312 pixels; we choose 100 samples for training and 25 for testing. For each epoch, we randomly sample 4 patches from each sample. For the Elegans dataset (Ljosa et al., 2012), we crop the images to 96×96 pixels; we randomly choose 80 samples for training and 20 for testing. For each epoch, we randomly sample 1 patch from each sample. For the synMnist dataset (LeCun, 1998), we synthetically modify the MNIST dataset to an image size of 48×48 pixels; please see our GitHub repository for details; we train on 4500 full, randomly chosen images and use 1500 for testing. For the Roads dataset (Mnih, 2013), we downsample the images to 375×375 pixels; we randomly choose 100 samples for training and 24 for testing. For each epoch, we randomly sample 8 patches from each sample.

L. Network specifications

We use the following notation:

1. In($input\ channels$), Out($output\ channels$), BI($output\ channels$) present input, output, and bottleneck channel (for U-Net);
2. C($filter\ size, output\ channels$) denote a convolutional layer followed by $ReLU$ and batch-normalization;
3. U($filter\ size, output\ channels$) denote a transposed convolution followed by $ReLU$ and batch-normalization;
4. $\downarrow 2$ denotes maxpooling;
5. $\oplus$ indicates concatenation of information from an encoder block.

L.1. Unet Configuration-I

We use this configuration for CREMI, synthMNIST, Colon and Elegans dataset. This is a lightweight U-net which has sufficient expressive power for these datasets.

ConvBlock : $C_B(3, out\ size) \equiv C(3, out\ size) \rightarrow C(3, out\ size) \rightarrow \downarrow 2$

UpConvBlock: $U_B(3, out\ size) \equiv U(3, out\ size) \rightarrow \oplus \rightarrow C(3, out\ size)$

Encoder : $IN(\text{1/3 ch}) \rightarrow C_B(3, 16) \rightarrow C_B(3, 32) \rightarrow C_B(3, 64) \rightarrow C_B(3, 128) \rightarrow C_B(3, 256) \rightarrow B(256)$

Decoder : $B(256) \rightarrow U_B(3, 256) \rightarrow U_B(3, 128) \rightarrow U_B(3, 64) \rightarrow U_B(3, 32) \rightarrow U_B(3, 16) \rightarrow Out(1)$

L.2. Unet Configuration-II

We had to choose a different U-Net architecture for the road and building dataset because we realized that a larger model is needed to learn useful features for this complex task.

ConvBlock : $C_B(3, out\ size) \equiv C(3, out\ size) \rightarrow C(3, out\ size) \rightarrow \downarrow 2$

UpConvBlock: $U_B(3, out\ size) \equiv U(3, out\ size) \rightarrow \oplus \rightarrow C(3, out\ size)$

Encoder : $IN(\text{3 ch}) \rightarrow C_B(3, 64) \rightarrow C_B(3, 128) \rightarrow C_B(3, 256) \rightarrow C_B(3, 512) \rightarrow C_B(3, 1024) \rightarrow B(1024)$

Decoder : $B(1024) \rightarrow U_B(3, 1024) \rightarrow U_B(3, 512) \rightarrow U_B(3, 256) \rightarrow U_B(3, 128) \rightarrow U_B(3, 64) \rightarrow Out(1)$

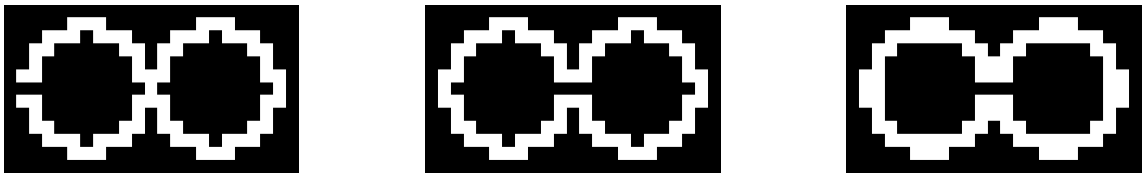
[2] https://github.com/nstucki/Betti-matching/

M. Evaluation metrics

We evaluate our experiments using a set of topological and pixel-based metrics. The metrics are computed with respect to the binarized predictions. Here, Betti matching error constitutes the most meaningful quantification, see section 3.3. We calculate the Betti matching error for dimension 0 (μ_0^{err}) and dimension 1 (μ_1^{err}) as well as their sum (μ^{err}). Furthermore, we implement the **Betti number error** for dimension 0 (β_0^{err}), dimension 1 (β_1^{err}), and their sum (β^{err}):

$$\beta^{\text{err}}(\boldsymbol{P}, \boldsymbol{G}) = \sum_{d=0}^{\infty} |\beta_d(D(\boldsymbol{P})_{0.5}) - \beta_d(D(\boldsymbol{G})_{0.5})|$$

It computes the Betti numbers of both foregrounds and sums up their absolute difference in each dimension, i.e., it compares the topological complexity of the foregrounds. It is important to consider the dimensions separately since they have different relevance on different datasets. E.g., Roads has many 1-cycles, whereas Buildings has many 0-cycles (connected components).



(a) Prediction 1
Acc $= \mathbf{0.99}$
Dice $= \mathbf{0.99}$
$\beta^{\text{err}} = 0$
$\mu^{\text{err}} = 2$

(b) Ground truth

(c) Prediction 2
Acc $= 0.97$
Dice $= 0.95$
$\beta^{\text{err}} = 0$
$\mu^{\text{err}} = \mathbf{0}$

Figure 20: (a) and (c) show two predictions for ground truth (b). Volumetric metrics, e.g., Accuracy and Dice favor (a) over (c), and even Betti number error can not differentiate between (a) and (c) while only Betti matching detects the spatial error in (a) and favors (c).

Additionally, we use the traditional Dice metric and Accuracy, which describe the in total correctly classified pixels, as well as the clDice metric from (Shit et al., 2021). Here, we calculate the clDice between the volumes and the skeleta, extracted using the *skeletonize* function of the skimage python-library. We compute all metrics on the individual test images of their respective size (without patching) and take the mean across the whole testset.

N. Basic definitions and terminology

N.1. Cubical complexes

A d-dimensional **(cubical) cell** in $\mathbb{R}^n$ is the Cartesian product $c = \prod_{j=1}^{n} I_j$ of intervals $I_j = [a_j, b_j]$ with $a_j \in \mathbb{Z}$, $b_j \in \{a_j, a_j + 1\}$ and $d \in \{0, \dots, n\}$ is the number of non-degenerate intervals among $\{I_1, \dots, I_d\}$.

If c and d are cells and $c \subseteq d$, we call c a **face** of d of **codimension** $\dim(d) - \dim(c)$. A face of codimension one is also called a **facet**.

A d-dimensional **(cubical) complex** in $\mathbb{R}^n$ is a finite set of cubical cells in $\mathbb{R}^n$ with maximal dimension d that is closed under the face relation, i.e., if $d \in K$ and c is a face of d, then $c \in K$. Furthermore we call a cubical complex $K' \subseteq K$ a **subcomplex** of K.

A **filtration** of a cubical complex K is given by a family $(K_r)_{r\in\mathbb{R}}$ of subcomplexes of K, which satisfies:

(1) $K_r \subseteq K_s$ for all $r \leq s$,

(2) $K = K_r$ for some $r \in \mathbb{R}$.

A **filtered (cubical) complex** K_* is a cubical complex K together with a nested sequence of subcomplexes, i.e., a sequence of complexes

$$\emptyset = K_0 \subseteq K_1 \dots \subseteq K_m = K.$$

A function $f\colon K \to \mathbb{R}$ on a cubical complex is said to be **order preserving** if $f(c) \leq f(d)$ for a face c of a cell d.

N.2. Homology

A **chain complex** C_* consists of a family $\{C_d\}_{d\in\mathbb{Z}}$ of vector spaces and a family of linear maps $\{\partial_d\colon C_d \to C_{d-1}\}_{d\in\mathbb{Z}}$ that satisfy $\partial_{d-1} \circ \partial_d = 0$.

For $d \in \mathbb{Z}$, we denote by K_d the set of d-dimensional cells in a cubical complex K. The $\mathbb{F}_2$-vector space $C_d(K)$ freely generated by K_d is the **chain group** of K in degree d. We can think of the elements in $C_d(K)$ as sets of d-dimensional cells and call them **chains**. These chain groups are connected by linear **boundary maps** $\partial_d\colon C_d(K) \to C_{d-1}(K)$, which map a cell to the sum of its faces of codimension 1 and are extended linearly to all of $C_d(K)$. The **cubical chain complex** $C_*(K)$ is given by the pair $(\{C_d(K)\}_{d\in\mathbb{Z}}, \{\partial_d\}_{d\in\mathbb{Z}})$. We denote by $Z_d(K) = \ker \partial_d$ the subspace of **cycles** and by $B_d(K) = \operatorname{im} \partial_{d+1}$ the subspace of **boundaries** in $C_d(K)$. Since $\partial_{d-1} \circ \partial_d = 0$, every boundary is a cycle and the **homology group** of K in degree d is defined by the quotient space $H_d(K) := Z_d(K)/B_d(K)$. In other words, $H_d(K)$ consists of equivalence classes of d-cycles and two d-cycles z_1, z_2 are equivalent (**homologous**) if their difference is a boundary. For convenience, we define $H_*(K) = \bigoplus_{d\in\mathbb{Z}} H_d(K)$. Note that the homology groups still carry the structure of a $\mathbb{F}_2$-vector space and their dimension $\beta_d(K) = \dim_{\mathbb{F}_2}(H_d(K))$ is the dth **Betti number** of K.

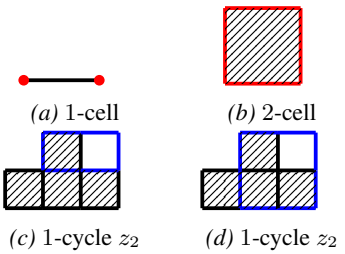


Figure 21: (a) and (b) show cells and their boundary (red). (c) and (d) visualize two homologous 1-cycles (blue) in a cubical complex.

A map $f\colon K \to K'$ between cubical complexes is said to be **cubical** if it respects the face relation, i.e., $f(c)$ must be a face of $f(d)$ in K' if c is a face of d in K.

A *cubical* map $f\colon K \to K'$ induces a linear map $C_*(f)\colon C_*(K) \to C_*(K')$, by mapping a cell $c \in K$ with $\dim(f(c)) = \dim(c)$ to $f(c)$ and extending this assignment linearly to all of $C_*(K)$. Then $C_*(f)$ descends to a linear map $H_*(f)\colon H_*(K) \to H_*(K')$ in homology since $\partial_* \circ C_*(f) = C_*(f) \circ \partial_*$.

N.3. Persistence modules

A **persistence module** M consists of a family $\{M_r\}_{r\in\mathbb{R}}$ of vector spaces, which are connected by linear **transition maps** maps $M_{r,s}\colon M_r \to M_s$ for all $r \le s$, such that

(1) $M_{r,r} = \mathrm{id}_{M_r}$ for all $r \in \mathbb{R}$,

(2) $M_{s,t} \circ M_{r,s} = M_{r,t}$ for $r \le s \le t$.

M is said to be **pointwise finite-dimensional** (p.f.d.) if M_r is finite-dimensional for every $r \in \mathbb{R}$.

A basic example of a persistence module is an **interval module** $C(\mathcal{I})$ for a given interval $\mathcal{I} \subseteq \mathbb{R}$. It consists of vector spaces

$$C(\mathcal{I})_r = \begin{cases} \mathbb{F}_2 & \text{if } r \in \mathcal{I}, \\ 0 & \text{otherwise.} \end{cases}$$

and transition maps

$$C(\mathcal{I})_{r,s} = \begin{cases} \mathrm{id}_{\mathbb{F}_2} & \text{if } r, s \in \mathcal{I}, \\ 0 & \text{otherwise.} \end{cases}$$

for $r \le s$.

A **morphism** $\Phi\colon M \to N$ between persistence modules is a family $\{\Phi_r\colon M_r \to N_r\}_{r\in\mathbb{R}}$ of linear maps, such that for all $r \le s$ the following diagram commutes:

$$\begin{array}{ccc} M_r & \xrightarrow{M_{r,s}} & M_s \\ {\scriptstyle \Phi_r}\downarrow & & \downarrow{\scriptstyle \Phi_s} \\ N_r & \xrightarrow[N_{r,s}]{} & N_s \end{array}$$

We call Φ an **isomorphism** (resp. **monomorphism**, **epimorphism**) of persistence modules if Φ_r is a isomorphism (resp. monomorphism, epimorphism) of vector spaces for all $r \in \mathbb{R}$.

For a family $\{M_i\}_{i\in I}$ of persistence modules, the **direct sum** $\bigoplus_{i\in I} M_i$ is the persistence module consisting of vector spaces $(\bigoplus_{i\in I} M_i)_r = \bigoplus_{i\in I}(M_i)_r$ for all $r \in \mathbb{R}$ and transition maps $(\bigoplus_{i\in I} M_i)_{r,s} = \bigoplus_{i\in I}(M_i)_{r,s}$ for all $r \le s \in \mathbb{R}$.

A **multiset** X consists of a set $|X|$ together with a **multiplicity function** $\mathrm{mult}_X\colon |X| \to \mathbb{N} \cup \{\infty\}$. Equivalently it can be represented by its **underlying set** $\amalg X = \bigcup_{x\in|X|} \coprod_{i=1}^{\mathrm{mult}_X(x)} \{x\}$. We say X is finite if its underlying set $\amalg X$ is finite and its cardinality $\#X$ is given by the cardinality of its underlying set.

Let K_* be a filtered cubical complex and L_* a cell-wise refinement according to the compatible ordering $c_1, \dots, c_l$ of the cells in K. The **boundary matrix** $\boldsymbol{B} \in \mathbb{F}_2^{l\times l}$ of L_* is given entry-wise by

$$\boldsymbol{B}_{i,j} = \begin{cases} 1 & \text{if } \sigma_i \text{ is a facet of } \sigma_j, \\ 0 & \text{otherwise.} \end{cases}$$

N.4. Matchings

A **map** $f\colon X \to Y$ between multisets is a map $f\colon \amalg X \to \amalg Y$ between their underlying sets.

A **matching** $\sigma\colon X \to Y$ between multisets is a bijection $\sigma\colon \amalg X' \to \amalg Y'$ for some multisets X', Y' that satisfy $\amalg X' \subseteq \amalg X$ and $\amalg Y' \subseteq \amalg Y$. We call

- $\mathrm{coim}(\sigma) = X'$ the **coimage** of σ,
- $\mathrm{im}(\sigma) = Y'$ the **image** of σ,
- $\ker(\sigma) = X \setminus X'$ the **kernel** and of σ,
- $\mathrm{coker}(\sigma) = Y \setminus Y'$ the **cokernel** of σ.

For a morphism $\Phi\colon M \to N$ of persistence modules, the **image** of Φ is the persistence module $\mathrm{im}(\Phi)$, with $\mathrm{im}(\Phi)_r = \mathrm{im}(\Phi_r)$ and transition maps $\mathrm{im}(\Phi)_{r,s} = N_{r,s}|_{\mathrm{im}(\Phi_r)}\colon \mathrm{im}(\Phi_r) \to \mathrm{im}(\Phi_s)$ for $r, s \in \mathbb{R}$.

Let M, N be persistence modules. We call M a **(persistence) submodule** of N if M_r is a subspace of N_r for every $r \in \mathbb{R}$ and the inclusions $i_r\colon M_r \hookrightarrow N_r$ assemble to a persistence map $i = (i_r)_{r\in\mathbb{R}}$. In this case we write $M \subseteq N$.

The **composition** of two matchings $X \xrightarrow{\sigma_1} Y \xrightarrow{\sigma_2} Z$ is given by the composition of the bijections

$$\sigma_1^{-1}(Y') \xrightarrow{\sigma_1} Y' \xrightarrow{\sigma_2} \sigma_2(Y'),$$

with $Y' = \amalg\,\mathrm{im}(\sigma_1) \cap \amalg\,\mathrm{coim}(\sigma_2)$.

A persistence module M is said to be **staggered** if every real number $r \in \mathbb{R}$ occurs at most once as endpoint of an interval in $\mathcal{B}(M)$.