COMPONENT TREE LOSS FUNCTION: DEFINITION AND OPTIMIZATION

Benjamin Perret, and Jean Cousty

LIGM, Univ Gustave Eiffel, CNRS, ESIEE Paris, F-77454 Marne-la-Vallée

ABSTRACT

In this article, we propose a method to design loss functions based on component trees which can be optimized by gradient descent algorithms and which are therefore usable in conjunction with recent machine learning approaches such as neural networks. We show how the altitudes associated to the nodes of such hierarchical image representations can be differentiated with respect to the image pixel values. This feature is used to design a generic loss function that can select or discard image maxima based on various attributes such as extinction values. The possibilities of the proposed method are demonstrated on simulated and real image filtering.

Index Terms— max-tree, connected filters, topological loss, continuous optimization, mathematical morphology

1. INTRODUCTION

Component-trees are hierarchical image representations that are classically used to perform connected image analysis and filtering [1, 2]. In such methods, an image is seen as the collection of the connected components of its level sets, thus offering a representation based on elements of higher semantic level, connected components instead of pixels, to design new image analysis methods. These approaches have provided efficient solutions in many image analysis domains such as feature detection [3, 4], segmentation [1, 2, 5, 6, 7], or object detection and proposal [8, 9].

However, those methods, based on topological decompositions, do not play well with recent machine learning approaches such as neural networks as their combinatorial nature is, at first sight, not well suited to optimization strategies based on gradient descent. In this context, some authors have recently proposed topological loss functions [10, 11, 12, 13] that enables to enforce topological constraints in continuous optimization frameworks using notions coming from the persistent homology theory. It has also been shown that hierarchies of segmentations can also be used in such context with the introduction of an ultrametric layer [14].

In this article, we propose a novel approach to use component trees, and more specifically max-trees, within continuous optimization methods. This approach is based on the observation that, in such trees, the altitude of a node (the level of the

level-set where it first appears) is directly linked to the value of some pixels of the image. Hence, we study how we can back-propagate any slight modification of the altitude of a node of the tree to a slight modification of the initial image. We then design a component tree loss function that enforces the presence of a prescribed number of maxima in the image based on maxima measures. We study how extinction values [15], maxima measures notably used in mathematical morphology to define hierarchical watersheds [16, 17], can be used to modify the behavior of the proposed loss function. Finally, the method has been implemented in Pytorch and we provide preliminary results demonstrating the use of the proposed approach on simulated and real images.

The article is organized as follows. The definition of max-trees is recalled in Sec. 2. Then, Sec. 3 presents how max-trees can be used in gradient descent algorithms and formalizes the general optimization problem which we address. In Sec. 4, we define a component tree loss function used for maxima selection in the max-trees and we introduce different maxima measures. The experiments are presented in Sec. 5. Finally, Sec. 6 concludes the work and gives some perspectives.
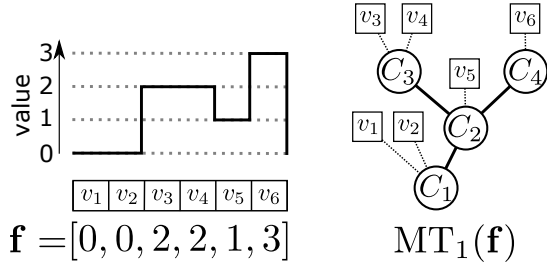
2. MAX-TREES

In this section, we recall the definition of max-trees [1, 2] which is based on the decomposition of every possible upper thresholds of an image into connected components.

In the following, the image domain is represented by a finite nonempty set $V = \{v_i\}_{i \in [\![1,n]\!]}$ of cardinality n. The elements of V are called *pixels*. Given any vector $\mathbf{v}$ of $\mathbb{R}^m$ with $m \in \mathbb{N}^+$, the i-th component of $\mathbf{v}$ is denoted $\mathbf{v}_i$. An *image* is represented by a vector $\mathbf{f} \in \mathbb{R}^n$ and, for any $i \in [\![1,n]\!]$, f_i is called *the value of the pixel* v_i. Note that any image can be represented as a vector by choosing an arbitrary ordering of the pixels (*e.g.*, a raster scan for 2d images) and that this choice does not change the results of the proposed method.

Let X be a subset of V, the set of connected components of X is denoted by $\mathcal{CC}(X)$ where connected components may be defined by any appropriate mean: *e.g.*, by path connectivity in a graph. In this article, all the examples involving 2d images are based on a classical 8-adjacency relation on a regular square grid of pixels. Let $\mathbf{f} \in \mathbb{R}^n$ be an image, the set of connected components of $\mathbf{f}$, denoted by $\mathcal{CC}(\mathbf{f})$, is defined by $\mathcal{CC}(\mathbf{f}) = \bigcup_{\lambda \in \mathbb{R}} \{\mathcal{CC}([\mathbf{f}]_\lambda)\}$ where, for any $\lambda \in \mathbb{R}$, $[\mathbf{f}]_\lambda$ is *the upper level*

set of $\mathbf{f}$ of level λ: $[\mathbf{f}]_\lambda = \{v_i \in V \mid \mathbf{f}_i \geq \lambda\}$. Note that the set $\mathcal{CC}(\mathbf{f})$ is finite and can thus be indexed by integers: $\mathcal{CC}(\mathbf{f}) = \{C_i\}_{i \in [\![1,m]\!]}$, where m is the number of connected components of $\mathbf{f}$. Let C_i in $\mathcal{CC}(\mathbf{f})$, the *altitude of* C_i is defined as the largest level λ in $\mathbb{R}$ such that C_i is a connected component of the upper level set of $\mathbf{f}$ at level λ: *i.e.*, $\max\left\{\lambda \in \mathbb{R} \mid C_i \in \mathcal{CC}([\mathbf{f}]^\lambda)\right\}$.

Let $\mathbf{f} \in \mathbb{R}^n$ be an image. The *max-tree* $\mathrm{MT}(\mathbf{f})$ *of* $\mathbf{f}$ is the pair $\left(\{C_i\}_{i \in [\![1,m]\!]}, \mathbf{a}\right)$ where $\{C_i\}$ is the set of connected components of $\mathbf{f}$ and where $\mathbf{a}$ is a vector of $\mathbb{R}^m$ such that $\mathbf{a}_i$ is equal to the altitude of C_i. The first element of the pair, denoted by $\mathrm{MT}_1(\mathbf{f})$, is called the *hierarchy of* $\mathrm{MT}(\mathbf{f})$. The second element of the pair, denoted by $\mathrm{MT}_2(\mathbf{f})$, is called the *altitude vector of* $\mathrm{MT}(\mathbf{f})$. An example is given in Fig. 1. An element of the hierarchy $\mathcal{H}$ of $\mathrm{MT}(\mathbf{f})$ is called *a node of* $\mathcal{H}$. A node C_i of $\mathcal{H}$ is called a *leaf* if there does not exists any other node included in it. There is a bijection between the leaf nodes of the hierarchy of $\mathrm{MT}(\mathbf{f})$ and the (regional) maxima of $\mathbf{f}$. The node V includes every node of $\mathcal{H}$ and is called the *root*. For any node C_i of $\mathcal{H}$, a pixel v in C_i that is not contained in any child of C_i is called *a proper pixel of* C_i. Any element v of V is a proper pixel of a unique node C_i denoted $\mathrm{par}(v)$.

$$\mathbf{f} = [0, 0, 2, 2, 1, 3] \qquad \mathrm{MT}_1(\mathbf{f})$$

Fig. 1. Max-tree example. The left figure shows a 1d image $\mathbf{f} \in \mathbb{R}^6$ defined on the domain $v_1, \ldots, v_6$. Each of the four level sets at levels 0, 1, 2, and 3, has a single connected component $C_1, \ldots, C_4$. Those components are the nodes of hierarchy $MT_1(\mathbf{f})$ (circles) shown on the right image. The plain lines represent the parent relations between nodes. The proper elements of each node are depicted by squares and the dashed lines show the parent relation between those proper elements and their respective nodes, for example we have $\mathrm{par}(v_4) = C_3$. The altitude vector $\mathbf{a} = \mathrm{MT}_2(\mathbf{f})$ of the max-tree of $\mathbf{f}$ is equal to $[0, 1, 2, 3]$, meaning for example that the altitude of the node C_2 is equal to 1. The two maxima of $\mathbf{f}$ corresponds to the leaf nodes C_3 and C_4 of the hierarchy $MT_1(\mathbf{f})$.

3. OPTIMIZATION WITH DIFFERENTIABLE MAX-TREES

In this section, we first study how the altitude vector of a max-tree can be sub-differentiated, then we state the general formulation of the optimization problem which is addressed.

Differentiable max-trees. Trees, as combinatorial structures, are generally not suited to gradient-based optimization.

However, in max-trees, the altitude of a component is mapped to the value of some pixels of the base image: its proper pixels. Then, intuitively, a small modification of the values of those proper pixels wont change the hierarchy associated to the max-tree of the image and will produce the exact same modification of the altitude of the corresponding node of the hierarchy.

Property 1. *Let* $\mathbf{f} \in \mathbb{R}^n$ *be an image. Let* $\epsilon \in \mathbb{R}^n$ *such that* $\mathrm{MT}_1(\mathbf{f}) = \mathrm{MT}_1(\mathbf{f} + \epsilon)$. *Then, for any node* C_i *of* $\mathrm{MT}_1(\mathbf{f})$, *the altitude of* C_i *in* $\mathrm{MT}(\mathbf{f} + \epsilon)$ *is equal to* $\mathbf{a}_i + \epsilon_j$ *where* $\mathbf{a}_i$ *is the altitude of* C_i *and where* v_j *is any proper pixel of* C_i.

This property indicates that the Jacobian of the function MT_2 can be written as the matrix composed of the indicator column vectors giving the index of the node associated to any pixel of V by the parent mapping (its proper elements):

$$\frac{\partial \mathrm{MT}_2(\mathbf{f})}{\partial \mathbf{f}} = \left[\mathbb{1}_{\mathrm{par}(v_1)}, \ldots, \mathbb{1}_{\mathrm{par}(v_n)}\right], \qquad (1)$$

where $\mathbb{1}_{C_k}$ is the column vector of $\mathbb{R}^m$ equals to 1 in position k, and 0 elsewhere. In a back-propagation algorithm, this means that if we have an error measure e and we have already computed $\frac{\partial e}{\partial \mathbf{a}}$, *i.e.*, how the altitude vector $\mathbf{a} = \mathrm{MT}_2(\mathbf{f})$ of the max-tree of $\mathbf{f}$ should be modified in order to minimize e, we can then back-propagate through MT with the chain rule $\frac{\partial e}{\partial \mathbf{f}} = \frac{\partial \mathbf{a}}{\partial \mathbf{f}} \frac{\partial e}{\partial \mathbf{a}}$ leading to the simple formula $\left(\frac{\partial e}{\partial \mathbf{f}}\right)_i = \left(\frac{\partial e}{\partial \mathbf{a}}\right)_{\mathrm{par}(i)}$ telling how $\mathbf{f}$ should be modified to minimize e.

For example, the transpose of the Jacobian of the altitude vector $\mathrm{MT}_2(\mathbf{f})$ of the max-tree shown in Fig. 1 is equal to

$$\begin{array}{c}
\begin{array}{cccccc} \mathbf{f}_1 & \mathbf{f}_2 & \mathbf{f}_3 & \mathbf{f}_4 & \mathbf{f}_5 & \mathbf{f}_6 \end{array} \\
\begin{array}{c} \mathbf{a}_1 \\ \mathbf{a}_2 \\ \mathbf{a}_3 \\ \mathbf{a}_4 \end{array}
\begin{pmatrix}
1 & 1 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 1 & 0 \\
0 & 0 & 1 & 1 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 1
\end{pmatrix}.
\end{array}$$

This matrix indicates, how the image $\mathbf{f}$ should be modified in order to reflect a modification of the altitude vector $\mathbf{a}$ of the nodes of the max-tree of $\mathbf{f}$. For example, in order to increase the altitude $\mathbf{a}_3$ of the component C_3 by a small value ϵ, one must increase the value of $\mathbf{f}_3$ and $\mathbf{f}_4$ by this same value ϵ.

Optimization problem. We now state a general formulation of the optimization problem that we want to solve. Let $\mathbf{y} \in \mathbb{R}^n$ be an image representing an observation. We are interested in solving the following optimization problem:

$$\underset{\mathbf{f} \in \mathbb{R}^n}{\mathrm{minimize}} \, J(\mathbf{f}; \mathbf{y}), \qquad (2)$$

where J is a differentiable cost function involving the altitude vector $\mathrm{MT}_2(\mathbf{f})$. As this altitude vector $\mathrm{MT}_2(\mathbf{f})$ is differentiable with respect to the image $\mathbf{f}$, a local optimum of the above problem can be found by gradient descent algorithm.

4. MAXIMA LOSS

In the following, we study how to define a component tree loss imposing a topological criterion, by prescribing how many

maxima should be present in the result. The proposed approach relies on two features characterizing the maxima of an image:

- *a measure of saliency*: increasing this measure should reinforce the maxima and decreasing it should make it disappear; and

- *a measure of (relative) importance*: which provides a ranking of the maxima to identify those that should be reinforced and those which should disappear.

We first introduce a generic loss function to *select* a given number of maxima and to *discard* the others according to these two measures. Then, we introduce several measures that can be used to measure the importance and the saliency of maxima.

4.1. Ranked selection loss

Assume that the hierarchy of $\mathrm{MT}(\mathbf{f})$ contains k maxima $\{M_i\}_{i=[\![1,k]\!]}$ (its leave nodes). Let $\ell \in \mathbb{N}^+$ be a target number of maxima. Let $\mathbf{sm} \in \mathbb{R}^k$ and $\mathbf{im} \in \mathbb{R}^k$ represent respectively a saliency and an importance measure on the maxima $\{M_i\}$. Then we define the ranked selection function as:

$$J_r(\mathbf{sm}, \mathbf{im}; \ell) = \sum_{i=1}^{i \leq \ell} \max(m - \mathbf{sm}_{\mathbf{r}_i}, 0) + \sum_{i=\ell+1}^{i \leq k} \mathbf{sm}_{\mathbf{r}_i}$$
$$\text{with } \mathbf{r} = \mathrm{argsort}(\mathbf{im}), \qquad (3)$$

where $m \in \mathbb{R}$ is a constant margin, whose goal is to prevent selected maxima to grow without limit, and where argsort is the function that associates to any vector $\mathbf{v}$ of $\mathbb{R}^k$, a permutation vector $\mathbf{r}$ sorting the elements of $\mathbf{v}$ in decreasing order, *i.e.*, such that for any i, j in $[\![1, n]\!]$, we have $i < j \Rightarrow v_{\mathbf{r}_i} \geq v_{\mathbf{r}_j}$.

4.2. Maxima measures

We now define maxima measures that will be used as saliency and/or importance measures in the previous loss function. Recall that the leaves of the max-tree of an image $\mathbf{f}$ corresponds to the maxima of this image and assume that the hierarchy of $\mathrm{MT}(\mathbf{f})$ contains k maxima $\{M_i\}_{i=[\![1,k]\!]}$; a measure on $\{M_i\}$ is then a vector of $\mathbb{R}^k$.

Highest altitude: A simple way to measure the importance and the saliency of a maxima is to look at its highest altitude, *i.e.*, to the value of the pixels contained in the maxima. The highest altitude of the maxima of $\mathbf{f}$ is denoted $\mathbf{alt}(\mathbf{f})$.

Extinction values: Extinction values are classical maxima measure known for their robustness [15]. Given a family of image filters $\{\sigma_k\}_k$ whose activity increases with k (for any $k_1 \leq k_2$, we have $\sigma_{k_1} \geq \sigma_{k_2}$). The extinction value of a maxima M_i of $\mathbf{f}$ is equal to the smallest k such that M_i is not contained in any maxima of $\sigma_k(\mathbf{f})$. A typical example of extinction value is the *dynamics* which is based on the filtering that removes any node of the max tree that has a height (difference between the altitude of the deepest node in the
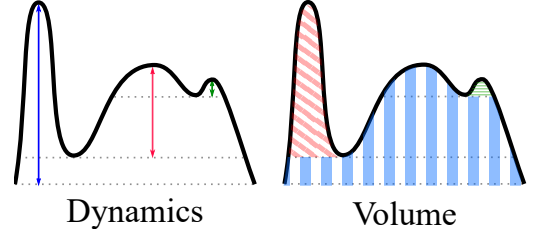
Dynamics   Volume

Fig. 2. Illustration of the dynamics and the volume associated to the maxima of a 1d function. The dynamics of a maxima is equal to the difference of altitude between the top of the maxima and the closest level that contains another maxima of greater altitude. Similarly the volume of the maxima is equal to the surface between the top of the maxima and the closest level that contains another maxima of greater volume. With the dynamics, the most important maxima is the sharp peak on the left while, with the volume, the most important maxima is the large mount in the middle.

subtree rooted in this node and the altitude of this node) smaller than a given threshold. The dynamics of the maxima of $\mathbf{f}$ will be denoted by $\mathbf{dyn}(\mathbf{f})$. Another classical extinction value is the one based on the volume filter, which removes any node of the max tree whose volume is smaller than a given threshold: this measure will be denoted by $\mathbf{vol}(\mathbf{f})$. In practice, computing the extinction value of a maxima can be done efficiently in the tree by finding the saddle node associated to this maxima, that is the closest ancestor of the maxima that contains another maxima with a greater attribute value. If no such node exists (the maxima has the largest value), then we consider that its saddle node is the root of the tree. The dynamics and volume associated to the maxima of a function are illustrated in Fig. 2.

5. EXPERIMENTS

We demonstrate the behavior of the proposed method and the various maxima measures on a simulated image and we show how it can be combined with classical loss functions to process real images. The method is implemented using the library Higra [18] for hierarchical graph analysis in combination with the continuous optimization framework Pytorch [19]. In all the experiments, an Adam optimizer [20] is used and the input image $\mathbf{y}$ is used as the initial solution. A Jupyter notebook containing the presented experiments is available online[1].

The effect of the component tree loss J_r with the proposed importance and saliency maxima measures is demonstrated on a simulated image in Fig. 3. The test image contains four maxima with different altitudes, sizes, and volumes. On the side of the importance measures, we can see that the altitude measure is not robust to noise and usually fails to select perceptually significant maxima. On the other hand, the two

---

[1]`https://www.esiee.fr/~perretb/tmp/Component_`
`Tree_Loss.ipynb`
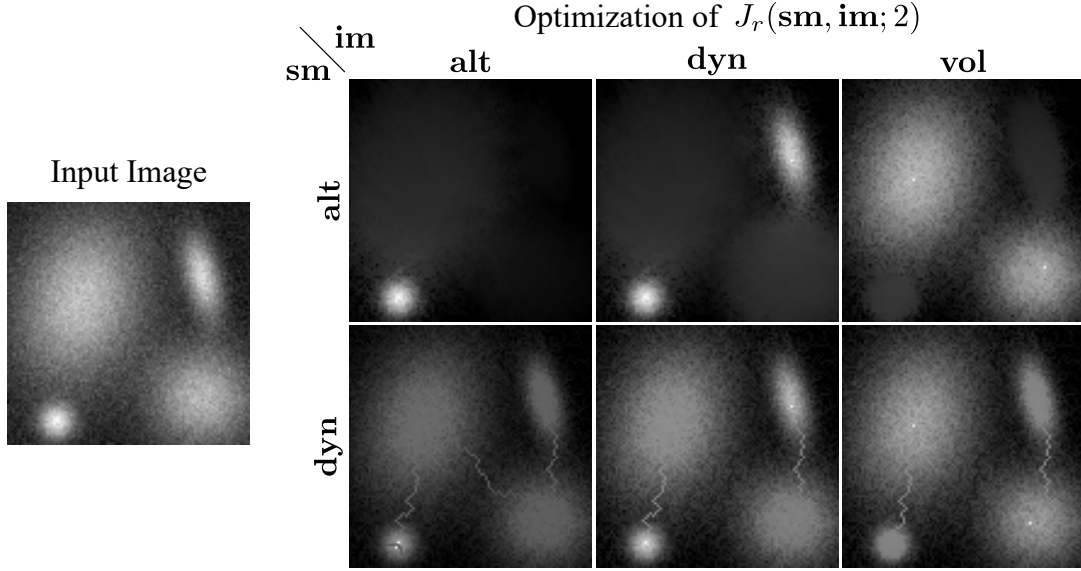
Fig. 3. Optimization of the component tree loss J_r on a simulated image with an objective of selecting 2 maxima. The figures on the right show the result for different combinations of maxima saliency measures (**sm**) and maxima importance measures (**im**).

measures based on extinction values, the dynamics and the volume, both manage to select significant maxima: with the dynamics, the two brightest maxima are selected while with the volume, the two largest maxima are selected. Regarding saliency measures, we can see that the optimization of the saliency based on maxima altitudes leads to increasing the altitudes of the top node of the selected maxima and to raising discarded maxima. The optimization of the dynamics saliency measure is more complex as increasing/decreasing the dynamics of the maxima involves increasing/decreasing the altitude of its top node and decreasing/increasing the altitudes of its saddle node: this leads to the creation of "bridges" between some maxima.

Note that optimizing J_r with the dynamics used both as the saliency and the importance measure of maxima is similar [21] to optimizing the barcode length of the connected components used in the "topological loss" based on persistent homology [10, 11, 12, 13]. However, as our approach does not require to compute the full persistence diagram associated to the image at each iteration of the optimization algorithm, we observe that it is dozens of times faster than a classical implementation of the topological loss[2].

Finally, in Fig. 4, we show how the proposed loss function can be combined with classical loss functions used in image analysis: here we optimize the term $||\mathbf{f} - \mathbf{y}||_2^2 + \lambda_1 J_r(\mathbf{dyn}(\mathbf{f}), \mathbf{dyn}(\mathbf{f}), 1) + \lambda_2 ||\nabla \mathbf{f}||_2^2$ which combines our loss based on the max-tree to enforce the presence of a single maxima with a L2 data attachment term and a total variation regularization term. We can see that we are able to successfully

---

[2] https://github.com/bruel-gabrielsson/TopologyLayer

| Image | Result |
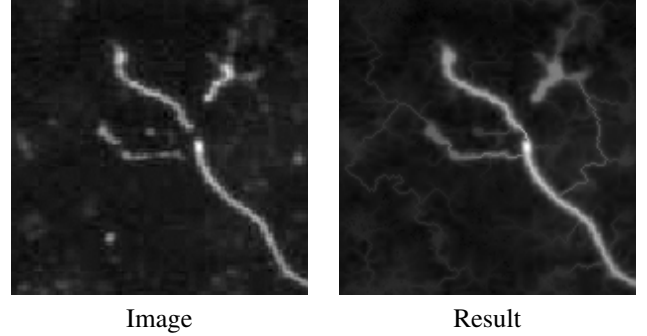
Fig. 4. Reconnection of a neurite image with a combination of the proposed loss J_r to enforce a single maximum, a L2 data attachment term and a TV2 regularization term.

reconnect the different branches of the neurite.

6. CONCLUSION

We have proposed a continuous optimization framework based on the hierarchical image representation called the max-tree. We showed how it can be used to design a component tree loss, *i.e.*, a regularization term, enabling to select or discard maxima in an image based on various measures. This approach can be generalized immediately to other hierarchical representations such as the min-tree or the tree-of-shapes [22, 23]. In future works, we plan to explore more general component tree loss functions based on such hierarchical representations and their use in supervised learning methods involving deep networks.

7. REFERENCES

[1] P. Salembier, A. Oliveras, and L. Garrido, "Anti-extensive connected operators for image and sequence processing," *IEEE TIP*, vol. 7, no. 4, pp. 555–570, 1998.

[2] R. Jones, "Connected filtering and segmentation using component trees," *CVIU*, vol. 75, no. 3, pp. 215–228, 1999.

[3] M. Donoser and H. Bischof, "Efficient maximally stable extremal region (mser) tracking," in *IEEE CVPR*, 2006, vol. 1, pp. 553–560.

[4] Y. Xu, P. Monasse, Th. Géraud, and L. Najman, "Tree-based morse regions: A topological approach to local feature detection," *IEEE TIP*, vol. 23, no. 12, pp. 5612–5625, 2014.

[5] M. Dalla Mura, J.A. Benediktsson, B. Waske, and L. Bruzzone, "Morphological attribute profiles for the analysis of very high resolution images," *IEEE TGRS*, vol. 48, no. 10, pp. 3747–3762, 2010.

[6] Y. Xu, E. Carlinet, Th. Géraud, and L. Najman, "Hierarchical segmentation using tree-based shape spaces," *IEEE TPAMI*, vol. 39, no. 3, pp. 457–469, 2016.

[7] J. Robic, B. Perret, A. Nkengne, M. Couprie, and H. Talbot, "Self-dual pattern spectra for characterising the dermal-epidermal junction in 3d reflectance confocal microscopy imaging," in *ISMM*. Springer, 2019, pp. 508–519.

[8] R. Girshick, J. Donahue, T. Darrell, and J. Malik, "Rich feature hierarchies for accurate object detection and semantic segmentation," in *IEEE CVPR*, 2014, pp. 580–587.

[9] P. Teeninga, U. Moschini, S.C. Trager, and M.H.F. Wilkinson, "Statistical attribute filtering to detect faint extended astronomical sources," *Mathematical Morphology-Theory and Applications*, vol. 1, 2016.

[10] J.R. Clough, I. Oksuz, N. Byrne, J.A. Schnabel, and A.P. King, "Explicit topological priors for deep-learning based image segmentation using persistent homology," in *IPMI*. Springer, 2019, pp. 16–28.

[11] X. Hu, F. Li, D. Samaras, and C. Chen, "Topology-preserving deep image segmentation," in *NeurIPS*, 2019, pp. 5657–5668.

[12] J. Clough, N. Byrne, I. Oksuz, V.A. Zimmer, J.A. Schnabel, and A. King, "A topological loss function for deep-learning based image segmentation using persistent homology," *IEEE TPAMI*, 2020.

[13] R.B. Gabrielsson, B.J. Nelson, A. Dwaraknath, and P. Skraba, "A topology layer for machine learning," in *AISTATS*. PMLR, 2020, pp. 1553–1563.

[14] G. Chierchia and B. Perret, "Ultrametric fitting by gradient descent," in *NeurIPS*, 2019, pp. 3181–3192.

[15] C. Vachier and F. Meyer, "Extinction value: a new measurement of persistence," in *IEEE Workshop on nonlinear signal and image processing*, 1995, vol. 1, pp. 254–257.

[16] J. Cousty and L. Najman, "Incremental algorithm for hierarchical minimum spanning forests and saliency of watershed cuts," in *ISMM*. Springer, 2011, pp. 272–283.

[17] B. Perret, J. Cousty, S.J.F. Guimaraes, and D.S. Maia, "Evaluation of hierarchical watersheds," *IEEE TIP*, vol. 27, no. 4, pp. 1676–1688, 2017.

[18] B. Perret, G. Chierchia, J. Cousty, S.J.F. Guimarães, Y. Kenmochi, and L. Najman, "Higra: Hierarchical graph analysis," *SoftwareX*, vol. 10, pp. 100335, 2019.

[19] A. Paszke, S. Gross, F. Massa, A. Lerer, J. Bradbury, G. Chanan, T. Killeen, Z. Lin, N. Gimelshein, L. Antiga, et al., "Pytorch: An imperative style, high-performance deep learning library," in *NeurIPS*, 2019, pp. 8026–8037.

[20] D.P. Kingma and J. Ba, "Adam: A method for stochastic optimization," in *ICLR*, Y. Bengio and Y. LeCun, Eds., 2015.

[21] N. Boutry, Th. Géraud, and L. Najman, "An equivalence relation between morphological dynamics and persistent homology in 1d," in *ISMM*. Springer, 2019, pp. 57–68.

[22] C. Ballester, V. Caselles, and P. Monasse, "The tree of shapes of an image," *ESAIM: Control, Optimisation and Calculus of Variations*, vol. 9, pp. 1–18, 2003.

[23] Th. Géraud, E. Carlinet, S. Crozet, and L. Najman, "A quasi-linear algorithm to compute the tree of shapes of nd images," in *ISMM*. Springer, 2013, pp. 98–110.