Beyond the Pixel-Wise Loss for Topology-Aware Delineation

Agata Mosinska[1*] Pablo Márquez-Neila[12] Mateusz Koziński[1†] Pascal Fua[1]
[1]Computer Vision Laboratory, École Polytechnique Fédérale de Lausanne (EPFL)
[2]ARTORG Center for Biomedical Engineering Research, University of Bern
{agata.mosinska, pablo.marquezneila, mateusz.kozinski, pascal.fua}@epfl.ch

Abstract

Delineation of curvilinear structures is an important problem in Computer Vision with multiple practical applications. With the advent of Deep Learning, many current approaches on automatic delineation have focused on finding more powerful deep architectures, but have continued using the habitual pixel-wise losses such as binary cross-entropy. In this paper we claim that pixel-wise losses alone are unsuitable for this problem because of their inability to reflect the topological impact of mistakes in the final prediction. We propose a new loss term that is aware of the higher-order topological features of linear structures. We also exploit a refinement pipeline that iteratively applies the same model over the previous delineation to refine the predictions at each step, while keeping the number of parameters and the complexity of the model constant.

When combined with the standard pixel-wise loss, both our new loss term and an iterative refinement boost the quality of the predicted delineations, in some cases almost doubling the accuracy as compared to the same classifier trained with the binary cross-entropy alone. We show that our approach outperforms state-of-the-art methods on a wide range of data, from microscopy to aerial images.

1. Introduction

Automated delineation of curvilinear structures, such as those in Fig. 1(a, b), has been investigated since the inception of the field of Computer Vision in the 1960s and 1970s. Nevertheless, despite decades of sustained effort, full automation remains elusive when the image data is noisy and the structures are complex. As in many other fields, the advent of Machine Learning techniques in general, and Deep Learning in particular, has produced substantial advances, in large part because learning features from the data makes them more robust to appearance variations [5, 17, 27, 32].

*This work was supported by Swiss National Science Foundation.
†This work was funded by the ERC FastProof grant.

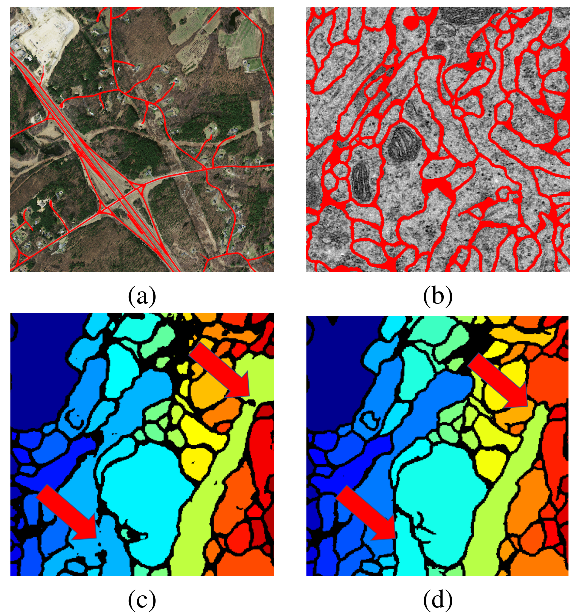


Figure 1: **Linear structures.** (a) Detected roads in an aerial image. (b) Detected cell membranes in an electron microscopy (EM) image. (c) Segmentation obtained after detecting neuronal membranes using [21] (d) Segmentation obtained after detecting membranes using our method. Our approach closes small gaps, which prevents much bigger topology mistakes.

However, all new methods focus on finding either better features to feed a classifier or more powerful deep architectures, while still using a pixel-wise loss such as binary cross-entropy for training purposes. Such loss is entirely local and does not account for the very specific and sometimes complex topology of curvilinear structures penalizing all mistakes equally regardless of their influence on geometry. As a shown in Fig. 1(c,d) this is a major problem because small localized pixel-wise mistakes can result in large topological changes.

In this paper, we show that supplementing the usual pixel-wise loss by a *topology loss* that promotes results with appropriate topological characteristics yields a substantial performance increase without having to change the network architecture. In practice, we exploit the feature maps computed by a pretrained VGG19 [26] to obtain high-level descriptions that are sensitive to linear structures. We use them to compare the topological properties of the ground truth and the network predictions and estimate our topology loss.

In addition to this we exploit iterative refinement freamework, which is inspired by the recurrent convolutional architecture of Pinheiro and Collobert [20]. We show that, unlike in the recent methods [19, 25], sharing the same architecture and parameters across all refinement steps, instead of instantiating a new network each time, results in state of the art performance and enables keeping the number of parameters constant irrespectively of the number of iterations. This is important when only a relatively small amount of training data is available, as is often the case in biomedical and other specialized applications.

Our main contribution is therefore a demonstration that properly accounting for topology in the loss used to train the network is an important step in boosting performance.

2. Related Work

2.1. Detecting Linear Structures

Delineation algorithms can rely either on hand-crafted or on learned features. Optimally Oriented Flux (OOF) [12] and Multi-Dimensional Oriented Flux (MDOF) [30], its extension to irregular structures, are successful examples of the former. Their great strength is that they do not require training data but at the cost of struggling with very irregular structures at different scales along with the great variability of appearances and artifacts.

In such challenging situations, learning-based methods have an edge and several approaches have been proposed over the years. For example, Haar wavelets [35] or spectral features [8] were used as features that were then input to the classifier. In [27], the classifier is replaced by a regressor that predicts the distance to the closest centerline, which enables estimating the width of the structures.

In more recent work, Deep Networks were successfully employed. For the purpose of road delineation, this was first done in [16], directly using image patches as input to a fully connected neural net. While the patch provided some context around the linear structures, it was still relatively small due to memory limitations. With the advent of Convolutional Neural Networks (CNNs), it became possible to use larger receptive fields. In [5], CNNs were used to extract features that could then be matched against *words* in a learned dictionary. The final prediction was made based on the votes from the nearest neighbors in the feature space. A fully-connected network was replaced by a CNN in [15] for road detection. In [14] a differentiable Intersection-over-Union loss was introduced to obtain a road segmentation, which is then used to extract graph of the road network. In the task of edge detection, nested, multiscale CNN features were utilized by Holistically-Nested Edge Detector [34] to directly produce an edge map of entire image.

In the biomedical field, the VGG network [26] pre-trained on real images has been fine-tuned and augmented by specialized layers to extract blood vessels [13]. Similarly the U-Net [21], has been shown to give excellent results for biomedical image segmentation and is currently among the methods that yield the best results for neuron boundaries detection in the ISBI'12 challenge [1].

While effective, all these approaches rely on a standard cross entropy loss for training purposes. Since they operate on individual pixels as though they were independent of each other, they ignore higher-level statistics while scoring the output. We will see in Section 4 that this is detrimental even when using an architecture designed to produce a structured output, such as the U-Net.

Of course, topological knowledge can be imposed in the output of these linear structure detectors. For example, in [32], this is done by introducing a CRF formulation whose priors are computed on higher-order cliques of connected superpixels likely to be part of road-like structures. Unfortunately, due to the huge number of potential cliques, it requires sampling and hand-designed features. Another approach to model higher-level statistics is to represent linear structures as a sequence of short linear segments, which can be accomplished using a Marked Point Process [28, 2, 24]. The inference involves Reversible Jump Markov Chain Monte Carlo and relies on a complex objective function. More recently, it has been shown that the delineation problem could be formulated in terms of finding an optimal subgraph in a graph of potential linear structures by solving an Integer Program [31, 18]. However, this requires a complex pipeline whose first step is finding points on the centerline of candidate linear structures.

Instead of encoding the topology knowledge explicitly, we propose to use higher-level features extracted using a pre-trained VGG network to score the predictions. Such feature statistics were used for image generation [4] and style transfer [6, 10], both tasks for which matching output statistics is a necessity because no ground truth annotations are available. However, delineation belongs in a different category because precise per-pixel ground truth is available and strict per-pixel supervision could be considered to be the most efficient approach. We show that this is not the case and that augmenting the pixel-oriented loss with a coarser, less localized, but semantically richer loss boosts performance. Moreover, to the best of our knowledge, it is the first successful attempt of using VGG features of *binary segmentation* images rather than natural ones, even though

VGG was pretrained on ImageNet [22].

2.2. Recursive Refinement

Recursive refinement of a segmentation has been extensively investigated. It is usually implemented as a procedure of iterative predictions [16, 20, 29], sometimes at different resolutions [23]. Such methods use the prediction from a previous iteration (and sometimes the image itself) as the input to a classifier that produces the next prediction. This enables the classifier to better consider the context surrounding a pixel when trying to assign a label to it and has been successfully used for delineation purposes [27].

In more recent works, the preferred approach to refinement with Deep Learning is to stack several deep modules and train them in an end-to-end fashion. For example, the pose estimation network of [19] is made of eight consecutive *hourglass* modules and supervision is applied on the output of each one during training, which takes several days. In [25] a similar idea is used to detect neuronal membranes in electron microscopy images, but due to memory size constraints the network is limited to 3 modules. In other words, even though such end-to-end solutions are convenient, the growing number of network parameters they require can become an issue when time, memory, and available amounts of training data are limited. This problem is tackled in [9] by using a single network that moves its attention field within the volume to be segmented. It predicts the output for the current field of view and fills in the prediction map.

Similarly, we also use the same network to refine its prediction. In terms of network architecture, our approach is most closely related to the recurrent network for image segmentation [20], with the notable difference that, while in the existing work each recursion/refinement step is instantiated for a different scale, we instantiate our refinement modules at a fixed scale and predict jointly the probability map for the whole patch. Compared to a typical Recurrent Neural Network [7], our architecture does not have memory. Moreover, in training, we use a loss function that is a weighted sum of losses computed after each processing step. This enables us to accumulate the gradients and requires neither seeds for initialization nor processing the intermediate output contrary to [9].

3. Method

We use the fully convolutional U-Net [21] as our trainable model, as it is currently among the best and most widely used architectures for delineation and segmentation in both natural and biomedical images. The U-Net is usually trained to predict the probability of each pixel of being a linear structure using a standard pixel-wise loss. As we have already pointed out, this loss relies on local measures and does not account for the overall geometry of curvilinear structures, which is what we want to remedy.

In the remainder of this section, we first describe our topology-aware loss function, and we then introduce iterative procedure to recursively refine our predictions.

3.1. Notation

In the following discussion, let $\mathbf{x} \in \mathbb{R}^{H \cdot W}$ be the $W \times H$ input image, and let $\mathbf{y} \in \{0, 1\}^{H \cdot W}$ be the corresponding ground-truth labeling, with 1 indicating pixels in the curvilinear structure and 0 indicating background pixels.

Let f be our U-Net parameterized by weights $\mathbf{w}$. The output of the network is an image $\hat{\mathbf{y}} = f(\mathbf{x}, \mathbf{w}) \in [0, 1]^{H \cdot W}$.[1] Every element of $\hat{\mathbf{y}}$ is interpreted as the probability of pixel i having label 1: $\hat{\mathbf{y}}_i \equiv p(Y_i = 1 \mid \mathbf{x}, \mathbf{w})$, where Y_i is a random Bernoulli variable $Y_i \sim \text{Ber}(\hat{\mathbf{y}}_i)$.

3.2. Topology-aware loss

In ordinary image segmentation problems, the loss function used to train the network is usually the standard pixel-wise binary cross-entropy (BCE):

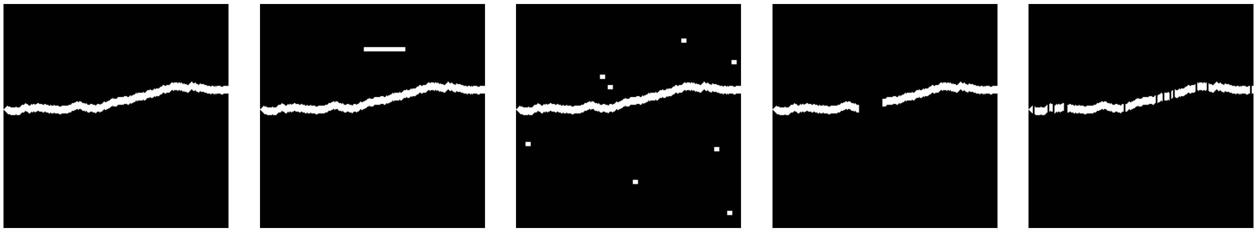
$$\mathcal{L}_{bce}(\mathbf{x}, \mathbf{y}, \mathbf{w}) = -\sum_i \left[(1 - \mathbf{y}_i) \cdot \log(1 - f_i(\mathbf{x}, \mathbf{w})) + \mathbf{y}_i \cdot \log f_i(\mathbf{x}, \mathbf{w})\right]. \quad (1)$$

Even though the U-Net computes a structured output and considers large neighborhoods, this loss function treats every pixel independently. It does not capture the characteristics of the topology, such as the number of connected components or number of holes. This is especially important in the delineation of thin structures: as we have seen in Fig. 1(c, d), the misclassification of a few pixels might have a low cost in terms of the pixel-wise BCE loss, but have a large impact on the topology of the predicted results.

Therefore, we aim to introduce a penalty term in our loss function to account for this higher-order information. Instead of relying on a hand-designed metric, which is difficult to model and hard to generalize, we leverage the knowledge that a pretrained network contains about the structures of real-world images. In particular, we use the feature maps at several layers of a VGG19 network [26] pretrained on the ImageNet dataset as a description of the higher-level features of the delineations. Our new penalty term tries to minimize the differences between the VGG19 descriptors of the ground-truth images and the corresponding predicted delineations:

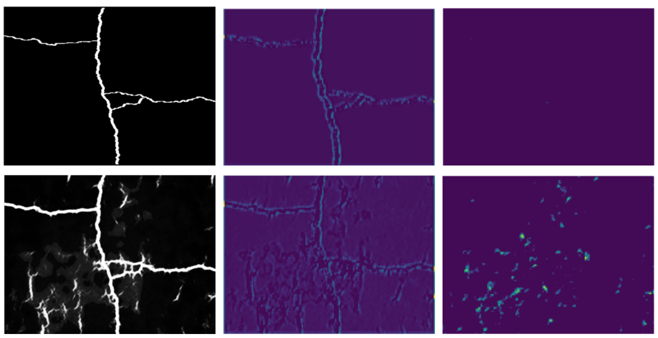
$$\mathcal{L}_{top}(\mathbf{x}, \mathbf{y}, \mathbf{w}) = \sum_{n=1}^{N} \frac{1}{M_n W_n H_n} \sum_{m=1}^{M_n} \left\| l_n^m(\mathbf{y}) - l_n^m(f(\mathbf{x}, \mathbf{w})) \right\|_2^2 , \quad (2)$$

[1] For simplicity and without loss of generality, we assume that $\mathbf{x}$ and $\hat{\mathbf{y}}$ have the same size. This is not the case in practice, and usually $\hat{\mathbf{y}}$ corresponds to the predictions of a cropped area of $\mathbf{x}$ (see [21] for details).



ground truth (a) | $\mathcal{L}_{top} = 0.2279$ (b) | $\mathcal{L}_{top} = 0.7795$ (c) | $\mathcal{L}_{top} = 0.2858$ (d) | $\mathcal{L}_{top} = 0.9977$ (e)

Figure 2: **The effect of mistakes on topology loss.** (a) Ground truth (b)-(e) we flip 240 pixels in each prediction, so that $\mathcal{L}_{bce}$ is the same for all of them, but as we see $\mathcal{L}_{top}$ penalizes more the cases with more small mistakes, which considerably change the structure of the prediction.



(a) (b) (c)

Figure 3: **Examples of activations in VGG layers.** (a) Ground-truth (top) and corresponding prediction with errors (bottom). (b) Responses of a VGG19 channel specialized in elongated structures. (c) Responses of a VGG19 channel specialized in small connected components. $\mathcal{L}_{top}$ strongly encourages responses in the former and penalizes responses in the latter.

where l_n^m denotes the m-th feature map in the n-th layer of the pretrained VGG19 network, N is the number of convolutional layers considered and M_n is the number of channels in the n-th layer, each of size $W_n \times H_n$. $\mathcal{L}_{top}$ can be understood as a measurement of the difference between the higher-level visual features of the linear structures in the ground-truth and those in predicted image. These higher-level features include concepts such as connectivity or holes that are ignored by the simpler pixel-wise BCE loss. Fig. 2 shows examples where the pixel-wise loss is too weak to penalize properly a variety of errors that occur in the predicted delineations, while our loss $\mathcal{L}_{top}$ correctly measures the topological importance of the errors in all cases: it penalizes more the mistakes that considerably change the structure of the image and those that do not resemble linear structures.

The reason behind the good performance of the VGG19 in this task can be seen in Fig. 3. Certain channels of the VGG19 layers are activated by the type of elongated structures we are interested in, while others respond strongly to small connected components. Thus, minimizing $\mathcal{L}_{top}$ strongly penalizes generating small false positives, which do not exist in the ground-truth, and promotes the generation of elongated structures. On the other hand, the shape of the predictions is ignored by $\mathcal{L}_{bce}$.

In the end, we minimize

$$\mathcal{L}(\mathbf{x}, \mathbf{y}, \mathbf{w}) = \mathcal{L}_{bce}(\mathbf{x}, \mathbf{y}, \mathbf{w}) + \mu \mathcal{L}_{top}(\mathbf{x}, \mathbf{y}, \mathbf{w}) \quad (3)$$

with respect to $\mathbf{w}$. μ is a scalar weighing the relative influence of both terms. We set it so that the order of magnitude of both terms is comparable. Fig. 4(a) illustrates the proposed approach.

3.3. Iterative refinement

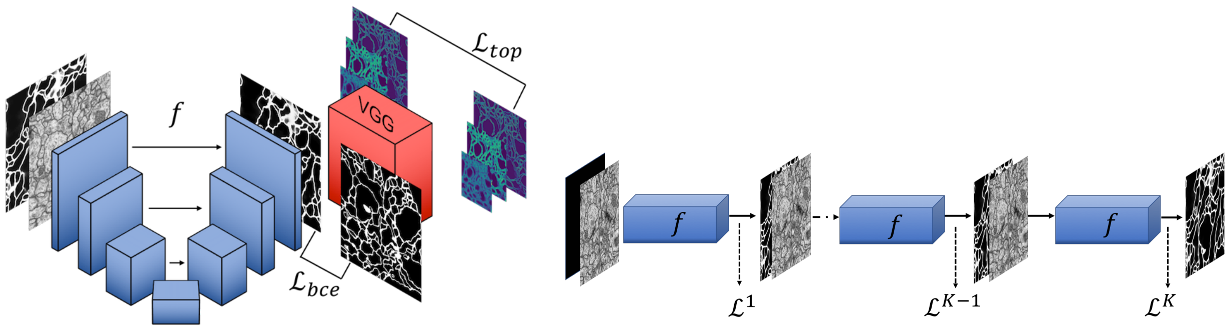
The topology loss term of Eq. 2 improves the quality of the predictions. However, as we will see in Section 4, some mistakes still remain. They typically show up in the form of small gaps in lines that should be uninterrupted. We iteratively refine the predictions to eliminate such problems. At each iteration, the network takes both the input image and the prediction of the previous iteration to successively provide better predictions.

In earlier works that advocate a similarly iterative approach [19, 25], a different module f^k is trained for each iteration k, thus increasing the number of parameters of the model and making training more demanding in terms of the amount of required labeled data. An interesting property of this iterative approach is that the correct delineation $\mathbf{y}$ should be the fixed point of each module f^k, that is, feeding the correct delineation should return the input

$$\mathbf{y} = f^k(\mathbf{x} \oplus \mathbf{y}), \quad (4)$$

where $\oplus$ denotes channel concatenation and we omitted the weights of f^k for simplicity. Assuming that every module f^k is Lipschitz-continuous on $\mathbf{y}$,[2] we know that the

[2] Lipschitz continuity is a direct consequence of the assumption that every f^k will always improve the prediction of the previous iteration.



(a) (b)

Figure 4: **Network architecture.** (a) We use the U-Net for delineation purposes. During training, both its output and the ground-truth image serve as input to a pretrained VGG network. The loss $\mathcal{L}_{top}$ is computed from the VGG responses. The loss $\mathcal{L}_{bce}$ is computed pixel-wise between the prediction and the ground-truth. (b) Our model iteratively applies the same U-Net f to produce progressive refinements of the predicted delineation. The final loss is a weighted sum of partial losses $\mathcal{L}^k$ computed at the end of each step.

fixed-point iteration

$$f^k(\mathbf{x} \oplus f^k(\mathbf{x} \oplus f^k(\ldots))) \tag{5}$$

converges to $\mathbf{y}$. We leverage this fixed-point property to remove the necessity of training a different module at each iteration. Instead, we use the same single network f at each step of the refinement pipeline, as depicted in Fig. 4(b). This makes our model much simpler and less demanding of labeled data for training.

This approach has also been applied before in [20] for image segmentation. Their application of the recurrent module was oriented towards increasing the spatial context. Rather than doing that, we keep the scale of the input to the modules fixed, in order to exploit the capacity of the network to correct its own errors. We show that it helps the network to learn a contraction map that successively improves the estimations. Our predictive model can therefore be expressed as

$$\hat{\mathbf{y}}^{k+1} = f(\mathbf{x} \oplus \hat{\mathbf{y}}^k, \mathbf{w}), \quad k = 0, \ldots, K-1\,, \tag{6}$$

where K is the total number of iterations and $\hat{\mathbf{y}}^K$ the final prediction. We initialize the model with an empty prediction $\hat{\mathbf{y}}^0 = \mathbf{0}$.

Instead of minimizing only the loss for the final network output, we minimize a weighted sum of partial losses. The k-th partial model, with $k \leq K$, is the model obtained from iterating Eq. 6 k times. The k-th partial loss $\mathcal{L}^k$ is the loss from Eq. 3 evaluated for the k-th partial model. Using this notation, we define our *refinement* loss as a weighted sum of the partial losses

$$\mathcal{L}_{ref}(\mathbf{x}, \mathbf{y}, \mathbf{w}) = \frac{1}{Z} \sum_{k=1}^{K} k\, \mathcal{L}^k(\mathbf{x}, \mathbf{y}, \mathbf{w})\,, \tag{7}$$

with the normalization factor $Z = \sum_{k=1}^{K} k = \frac{1}{2}K(K+1)$. We weigh more the losses associated with the final iterations to boost the accuracy of the final result. However, accounting for the earlier losses enables the network to learn from all the mistakes it can make along the way and increases numerical stability. It also avoids having to preprocess the predictions before re-injecting them into the computation, as in [9].

In practice, we first train a single module network, that is, for $K = 1$. We then increment K, retrain, and iterate. We limit K to 3 during training and testing as the results do not change significantly for larger K values. We will show that this successfully fills in small gaps.

4. Results

Data. We evaluate our approach on three datasets featuring very different kinds of linear structures:

1. **Cracks:** Images of cracks in road [36]. It consists of 104 training and 20 test images. As can be seen in Fig. 5, the multiple shadows and cluttered background makes their detection a challenging task. Applications include quality inspection and material characterization.
2. **Roads:** The Massachusetts Roads Dataset [15] is one of the largest publicly available collections of aerial road images, containing both urban and rural neighbourhoods, with many different kinds of roads ranging from small paths to highways. The set is split into 1108 training and 49 test images, 2 of which are shown in Fig. 6.
3. **EM:** We detect neuronal boundaries in Electron Microscopy images from the ISBI'12 challenge [1] (Fig. 7). There are 30 training images, with ground truth annotations, and 30 test images for which the ground-truth is withheld by the organizers. Following [23], we split the training set into 15 training and 15 test images. We report our results on this split.

Training protocol. Since the U-Net cannot handle very large images, we work with patches of 450×450 pixels for training. We perform data augmentation mirroring and rotating the training images by 90°, 180° and 270°. Additionally, in the **EM** dataset, we also apply elastic deformations as suggested in [21] to compensate for the small amount of training data. Ground-truth of **Cracks** dataset consists of centerlines, so we dilate it by margin of 4 pixels to perform segmentation. We use batch normalization for faster convergence and use current batch statistics also at the test time as suggested in [3]. We chose Adam [11] with a learning rate of 10^{-4} as our optimization method.

Pixel-wise metrics. Our algorithm outputs a probabilty map, which lends itself to evaluation in terms of precision- and recall-based metrics, such as the F1 score [23] and the precision-recall break-even point [15]. They are well suited for benchmarking binary segmentations, but their local character is a drawback in the presence of thin structures. Shifting a prediction even by a small distance in a direction perpendicular to the structure yields zero precision and recall, while still reasonably representing the data. We therefore evaluate the results in terms of *correctness*, *completeness*, and *quality* [33]. They are metrics designed specifically for linear structures, which measure the similarity between predicted skeletons and ground truth-ones. They are more sensitive to precise locations or small width changes of the underlying structures. Potential shifts in centerline positions are handled by relaxing the notion of a true positive from being a precise coincidence of points to not exceeding a distance threshold. Correctness corresponds to relaxed precision, completeness to relaxed recall, and quality to intersection-over-union. We give precise definitions in appendix. In our experiments we use a threshold of 2 pixels for roads and cracks, and 1 for the neuronal membranes.

Topology-based metrics. The pixel-wise metrics are oblivious of topological differences between the predicted and ground-truth networks. A more topology-oriented set of measures was proposed in [32]. It involves finding the shortest path between two randomly picked connected points in the predicted network and the equivalent path in the ground-truth network or vice-versa. If no equivalent path exists, the former is classified as *infeasible*. It is classified as *too-long/-short* if the length of the paths differ by more than 10%, and as *correct* otherwise. In practice, we sample 200 paths per image, which is enough for the proportion of correct, infeasible, and too-long/-short paths to stabilize.

The organizers of the EM challenge use a performance metric called *foreground-restricted random score*, oriented at evaluating the preservation of separation between different cells. It measures the probability that two pixels belonging to the same cell in reality and in the predicted output. As shown in Fig. 1(c,d), this kind of metric is far more sensitive to topological perturbations than to pixel-wise errors.

VGG layers	Quality	Number of iterations	Quality
None	0.3924	**OURS-NoRef**	0.5580
layer 1	0.6408	**OURS** 1 iteration	0.5621
layer 2	0.6427	**OURS** 2 iterations	0.5709
layer 3	0.6974	**OURS** 3 iterations	0.5722
layers 1,2,3	0.7446	**OURS** 4 iterations	0.5727

Table 1: Testing different configurations. (Left) Quality scores for **OURS-NoRef** method when using different VGG layers to compute the topology loss of Eq. 2 on the **Cracks** dataset. (Right) Quality scores for **OURS** method on the **EM** dataset as a function of the number of refinement iterations. **OURS-NoRef** included for comparison.

Baselines and variants of the proposed method. We compare the results of our method to the following baselines:

- **CrackTree** [36] a crack detection method based on segmentation and subsequent graph construction
- **MNIH** [15], a neural network for road segmentation in 64×64 image patches,
- **HED** [34], a nested, multi-scale approach for edge detection,
- **U-Net** [21], pixel labeling using the U-Net architecture with BCE loss,
- **CHM-LDNN** [23], a multi-resolution recursive approach to delineating neuronal boundaries,
- **Reg-AC** [27], a regression-based approach to finding centerlines and refining the results using autocontext.

We reproduce the results for **HED**, **MNIH**, **U-Net** and **Reg-AC**, and report the results published in the original work for **CHM-LDNN**. We also perform an ablation study to isolate the individual contribution of the two main components of our approach. To this end, we compare two variants of it:

- **OURS-NoRef**, our approach with the topological loss of Eq. 3 but no refinement steps. To extract global features we use the channels from the VGG layers **relu(conv1_2)**, **relu(conv2_2)** and **relu(conv3_4)**, and set μ to 0.1 in Eq. 3.
- **OURS**, our complete, iterative method including the topological term and $K = 3$ refinement steps. It is trained using the refinement loss of Eq. 7 as explained in Section 3.3.

4.1. Quantitative Results

We start by identifying the best-performing configuration for our method. As can be seen in Table 1(left), using all three first layers of the VGG network to compute the topology loss yields the best results. Similarly, we evaluated the impact of the number of improvement iterations on the resulting performance on the **EM** dataset, which we

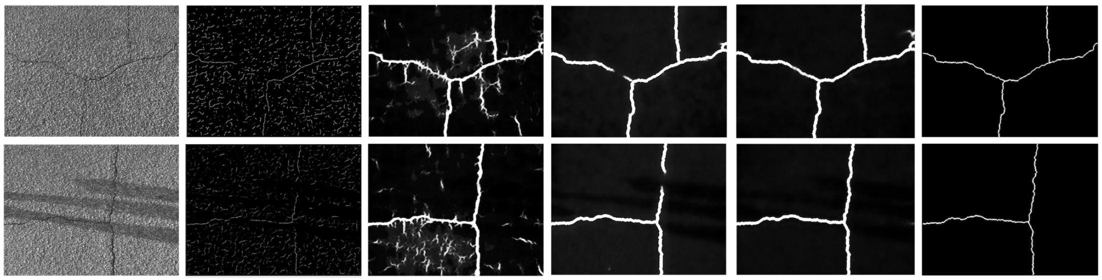


Figure 5: **Cracks.** From left to right: image, **Reg-AC**, **U-Net**, **OURS-NoRef** and **OURS** prediction, ground-truth.

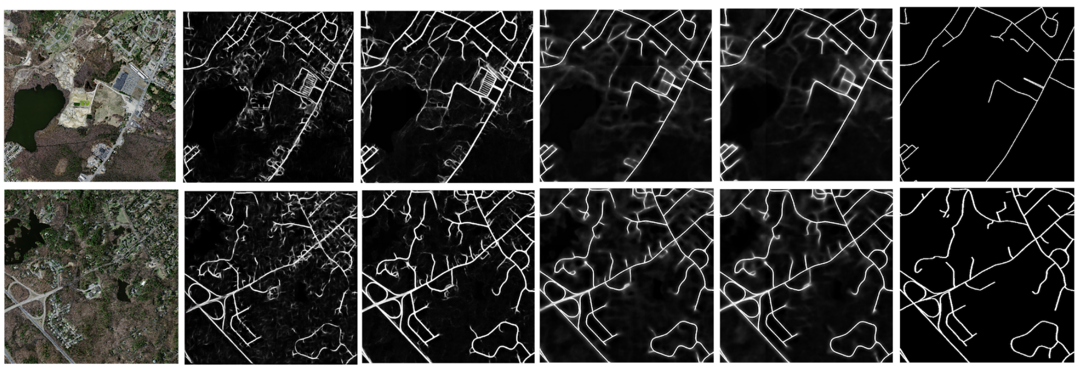


Figure 6: **Roads**. From left to right: image, **MNIH**, **U-Net**, **OURS-NoRef** and **OURS** prediction, ground-truth.

Method	P/R	Method	F1
MNIH [15]	0.6822	**CHM-LDNN** [23]	0.8072
HED [34]	0.7107	**HED** [34]	0.7850
U-Net [21]	0.7460	**U-Net** [21]	0.7952
OURS-NoRef	0.7610	**OURS-NoRef**	0.8140
OURS	**0.7782**	**OURS**	**0.8230**

Table 2: Experimental results on the **Roads** and **EM** datasets. (Left) Precision-recall break-even point (P/R) for the **Roads** dataset. Note the results are expressed in terms of the standard precision and recall, as opposed to the relaxed measures reported in [15]. (Right) F1 scores for the **EM** dataset.

Dataset	Method	Correct.	Complet.	Quality
Cracks	**CrackTree** [36]	0.7900	0.9200	0.7392
	Reg-AC [27]	0.1070	0.9283	0.1061
	U-Net [21]	0.4114	0.8936	0.3924
	OURS-NoRef	0.7955	0.9208	0.7446
	OURS	**0.8844**	**0.9513**	**0.8461**
Roads	**Reg-AC** [27]	0.2537	0.3478	0.1719
	MNIH [15]	0.5314	0.7517	0.4521
	U-Net [21]	0.6227	0.7506	0.5152
	OURS-NoRef	0.6782	0.7986	0.5719
	OURS	**0.7743**	**0.8057**	**0.6524**
EM	**Reg-AC** [27]	0.7110	0.6647	0.5233
	U-Net [21]	0.6911	0.7128	0.5406
	OURS-NoRef	0.7096	0.7231	0.5580
	OURS	**0.7227**	**0.7358**	**0.5722**

Table 3: Correctness, completeness and quality scores for extracted centerlines.

present in Table 1(right). The performance stabilizes after the third iteration. We therefore used three refinement iterations in all further experiments. Note that in Table 1(right) the first iteration of **OURS** yields a result that is already better than **OURS-NoRef**. This shows that iterative training not only makes it possible to refine the results by iterating at test time, but also yields a better standalone classifier.

We report results of our comparative experiments for the three datasets in Tables 2, 3, and 4. Even without refinement, our topological loss outperforms all the baselines. Refinement boosts the performance yet further. The differences are greater when using the metrics specifically designed to gauge the quality of linear structures in Table 3 and even more when using the topology-based metrics in Table 4. This confirms the hypothesis that our contributions improve the quality of the predictions mainly in its topological aspect. The improvement in per-pixel measures, presented in Table 2 suggests that the improved topology is correlated with better localisation of the predictions.

Finally, we submitted our results to the ISBI challenge server for the **EM** task. We received a foreground-restricted random score of 0.981. This puts us in first place among algorithms relying on a single classifier without additional processing. In second place is the recent method of [25], which achieves the slightly lower score of 0.978 even though it relies on a significantly more complex classifier.

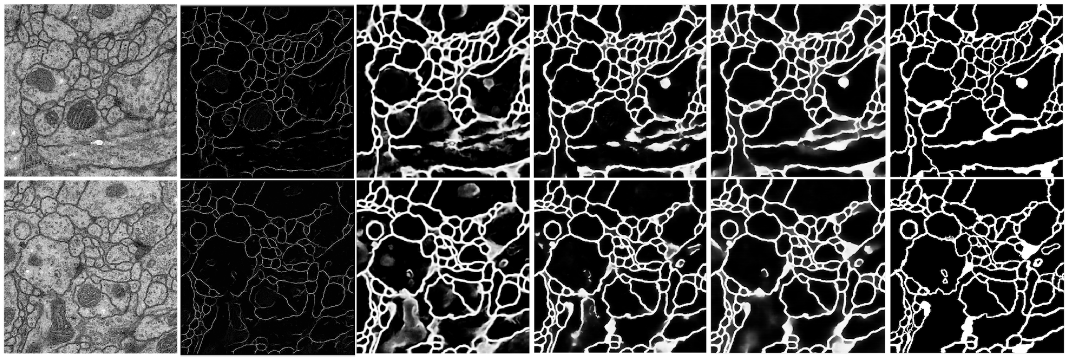


Figure 7: **EM**. From left to right: image, **Reg-AC**, **U-Net**, **OURS-NoRef** and **OURS** prediction, ground-truth.

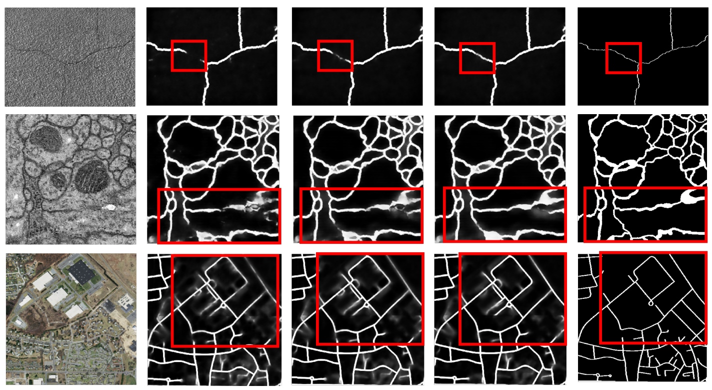


Figure 8: **Iterative Refinement.** Prediction after 1, 2 and 3 refinement iterations. The right-most image is the ground-truth. The red boxes highlight parts of the image where refinement is closing gaps.

Dataset	Method	Correct	Infeasible	2Long 2Short
Cracks	**Reg-AC** [27]	39.7	56.8	3.5
	U-Net [21]	68.4	27.4	4.2
	OURS-NoRef	90.8	6.1	3.1
	OURS	**94.3**	3.1	2.6
Roads	**Reg-AC** [27]	16.2	72.1	11.7
	MNIH [15]	45.5	49.73	4.77
	U-Net [21]	56.3	38.0	5.7
	OURS-NoRef	63.4	32.3	4.3
	OURS	**69.1**	24.2	6.7
EM	**Reg-AC** [27]	36.1	38.2	25.7
	U-Net [21]	51.5	16.0	32.5
	OURS-NoRef	63.2	16.8	20.0
	OURS	**67.0**	15.5	17.5

Table 4: The percentage of correct, infeasible and too-long/too-short paths sampled from predictions and ground truth.

4.2. Qualitative Results

Figs. 5, 6, and 7 depict typical results on the three datasets. Note that adding our topology loss term and iteratively refining the delineations makes our predictions more structured and consistently eliminates false positives in the background, without losing the curvilinear structures of interest as shown in Fig. 8. For example, in the aerial images of Fig. 6, line-like structures such as roofs and rivers are filtered out because they are not part of the training data, while the roads are not only preserved but also enhanced by closing small gaps. In the case of neuronal membranes, the additional topology term eliminates false positives corresponding to cell-like structures such as mitochondria.

5. Conclusion

We have introduced a new loss term that accounts for topology of curvilinear structures by exploiting their higher-level features. We have further improved it by applying recursive refinement that does not increase the number of parameters to be learned. Our approach is generic and can be used for detection of many types of linear structures including roads and cracks in natural images and neuronal membranes in micrograms. We have relied on the U-Net to demonstrate it but it could be used in conjunction with any other network architecture. In future work, we will explore the use of adversarial networks to adapt our measure of topological similarity and learn more discriminative features.

References

[1] I. Arganda-Carreras, S. Turaga, D. Berger, D. Ciresan, A. Giusti, L. Gambardella, J. Schmidhuber, D. Laptev, S. Dwivedi, J. Buhmann, T. Liu, M. Seyedhosseini, T. Tasdizen, L. Kamentsky, R. Burget, V. Uher, X. Tan, C. Sun, T. Pham, E. Bas, M. Uzunbas, A. Cardona, J. Schindelin, and S. Seung. Crowdsourcing the Sreation of Image Segmentation Algorithms for Connectomics. *Frontiers in Neuroanatomy*, page 142, 2015. 2, 5

[2] D. Chai, W. Forstner, and F. Lafarge. Recovering Line-Networks in Images by Junction-Point Processes. In *Conference on Computer Vision and Pattern Recognition*, 2013. 2

[3] Ö. Çiçek, A. Abdulkadir, S. Lienkamp, T. Brox, and O. Ronneberger. 3D U-Net: Learning Dense Volumetric Segmentation from Sparse Annotation. In *Conference on Medical Image Computing and Computer Assisted Intervention*, pages 424–432, 2016. 6

[4] A. Dosovitskiy and T. Brox. Generating Images with Perceptual Similarity Metrics based on Deep Networks. pages 658–666. 2

[5] Y. Ganin and V. Lempitsky. N4-Fields: Neural Network Nearest Neighbor Fields for Image Transforms. In *Asian Conference on Computer Vision*, 2014. 1, 2

[6] L. A. Gatys, A. S. Ecker, and M. Bethge. Image Style Transfer Using Convolutional Neural Networks. In *Conference on Computer Vision and Pattern Recognition*, 2016. 2

[7] S. Hochreiter and J. Schmidhuber. Long Short-Term Memory. *Neural Computation*, 9(8):1735–1780, 1997. 3

[8] X. Huang and L. Zhang. Road Centreline Extraction from High-Resolution Imagery Based on Multiscale Structural Features and Support Vector Machines. *International Journal of Remote Sensing*, 30:1977–1987, 2009. 2

[9] M. Januszewski, J. Maitin-Shepard, P. Li, J. Kornfeld, W. Denk, and V. Jain. Flood-Filling Networks. *arXiv Preprint*, 2016. 3, 5

[10] J. Johnson, A. Karpathy, and L. Fei-fei. Densecap: Fully Convolutional Localization Networks for Dense Captioning. In *Conference on Computer Vision and Pattern Recognition*, 2016. 2

[11] D. Kingma and J. Ba. Adam: A Method for Stochastic Optimisation. In *International Conference for Learning Representations*, 2015. 6

[12] M. Law and A. Chung. Three Dimensional Curvilinear Structure Detection Using Optimally Oriented Flux. In *European Conference on Computer Vision*, 2008. 2

[13] K. Maninis, J. Pont-Tuset, P. Arbeláez, and L. V. Gool. Deep Retinal Image Understanding. In *Conference on Medical Image Computing and Computer Assisted Intervention*, 2016. 2

[14] G. Mattyusand, W. L. R., and Urtasun. Deeproadmapper: Extracting Road Topology from Aerial Images. In *International Conference on Computer Vision*, 2017. 2

[15] V. Mnih. *Machine Learning for Aerial Image Labeling*. PhD thesis, University of Toronto, 2013. 2, 5, 6, 7, 8

[16] V. Mnih and G. Hinton. Learning to Detect Roads in High-Resolution Aerial Images. In *European Conference on Computer Vision*, 2010. 2, 3

[17] V. Mnih and G. Hinton. Learning to Label Aerial Images from Noisy Data. In *International Conference on Machine Learning*, 2012. 1

[18] A. Mosinska, J. Tarnawski, and P. Fua. Active Learning and Proofreading for Delineation of Curvilinear Structures. In *Conference on Medical Image Computing and Computer Assisted Intervention*, volume 10434, pages 165–173, 2017. 2

[19] A. Newell, K. Yang, and J. Deng. Stacked Hourglass Networks for Human Pose Estimation. In *European Conference on Computer Vision*, 2016. 2, 3, 4

[20] P. Pinheiro and R. Collobert. Recurrent Neural Networks for Scenel Labelling. In *International Conference on Machine Learning*, 2014. 2, 3, 5

[21] O. Ronneberger, P. Fischer, and T. Brox. U-Net: Convolutional Networks for Biomedical Image Segmentation. In *Conference on Medical Image Computing and Computer Assisted Intervention*, 2015. 1, 2, 3, 6, 7, 8

[22] O. Russakovsky, J. Deng, H. Su, J. Krause, S.Satheesh, S. Ma, Z. Huang, A. Karpathy, A. Khosla, M. Bernstein, A. Berg, and L. Fei-Fei. Imagenet Large Scale Visual Recognition Challenge. *International Journal of Computer Vision*, 115(3):211–252, 2015. 3

[23] M. Seyedhosseini, M. Sajjadi, and T. Tasdizen. Image Segmentation with Cascaded Hierarchical Models and Logistic Disjunctive Normal Networks. In *International Conference on Computer Vision*, 2013. 3, 5, 6, 7

[24] S.G. Jeong and Y. Tarabalka and J. Zerubia. Marked point process model for curvilinear structures extraction. In *Energy Minimization Methods in Computer Vision and Pattern Recognition*, pages 436–449, 2015. 2

[25] W. Shen, B. Wang, Y. Jiang, Y. Wang, and A. L. Yuille. Multi-stage Multi-recursive-input Fully Convolutional Networks for Neuronal Boundary Detection. 2017. 2, 3, 4, 7

[26] K. Simonyan and A. Zisserman. Very Deep Convolutional Networks for Large-Scale Image Recognition. In *International Conference for Learning Representations*, 2015. 2, 3

[27] A. Sironi, E. Turetken, V. Lepetit, and P. Fua. Multiscale Centerline Detection. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 38(7):1327–1341, 2016. 1, 2, 3, 6, 7, 8

[28] R. Stoica, X. Descombes, and J. Zerubia. A Gibbs Point Process for Road Extraction from Remotely Sensed Images. *International Journal of Computer Vision*, 57(2):121–136, 2004. 2

[29] Z. Tu and X. Bai. Auto-Context and Its Applications to High-Level Vision Tasks and 3D Brain Image Segmentation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2009. 3

[30] E. Turetken, C. Becker, P. Glowacki, F. Benmansour, and P. Fua. Detecting Irregular Curvilinear Structures in Gray Scale and Color Imagery Using Multi-Directional Oriented Flux. In *International Conference on Computer Vision*, December 2013. 2

[31] E. Turetken, F. Benmansour, B. Andres, P. Glowacki, H. Pfister, and P. Fua. Reconstructing Curvilinear Networks Using Path Classifiers and Integer Programming. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 38(12):2515–2530, 2016. 2

[32] J. Wegner, J. Montoya-Zegarra, and K. Schindler. A Higher-Order CRF Model for Road Network Extraction. In *Conference on Computer Vision and Pattern Recognition*, 2013. 1, 2, 6

[33] C. Wiedemann, C. Heipke, H. Mayer, and O. Jamet. Empirical Evaluation Of Automatically Extracted Road Axes. In *Empirical Evaluation Techniques in Computer Vision*, pages 172–187, 1998. 6

[34] S. Xie and Z. Tu. Holistically-Nested Edge Detection. *International Conference on Computer Vision*, pages 1395–1403, 2015. 2, 6, 7

[35] S. K. Zhou, C. Tietjen, G. Soza, A. Wimmer, C. Lu, Z. Puskas, D. Liu, and D. Wu. A Learning Based Deformable Template Matching Method for Automatic Rib Centerline Extraction and Labeling in CT Images. In *Conference on Computer Vision and Pattern Recognition*, 2012. 2

[36] Q. Zou, Y. Cao, Q. Li, Q. Mao, and S. Wang. CrackTree: Automatic crack detection from pavement images. *Pattern Recognition Letters*, 33(3):227–238, 2012. 5, 6, 7