Segmentation of Tubular Structures
Using Iterative Training with Tailored Samples

Wei Liao
Independent Researcher
liaowei.post@gmail.com

Abstract

We propose a minimal path method to simultaneously compute segmentation masks and extract centerlines of tubular structures with line-topology. Minimal path methods are commonly used for the segmentation of tubular structures in a wide variety of applications. Recent methods use features extracted by CNNs, and often outperform methods using hand-tuned features. However, for CNN-based methods, the samples used for training may be generated inappropriately, so that they can be very different from samples encountered during inference. We approach this discrepancy by introducing a novel iterative training scheme, which enables generating better training samples specifically tailored for the minimal path methods without changing existing annotations. In our method, segmentation masks and centerlines are not determined after one another by post-processing, but obtained using the same steps. Our method requires only very few annotated training images. Comparison with seven previous approaches on three public datasets, including satellite images and medical images, shows that our method achieves state-of-the-art results both for segmentation masks and centerlines.

1. Introduction

Segmentation of tubular structures is an essential task in many application areas of computer vision, including navigation, delineation of roads in satellite images, and analysis of blood vessels in medical images (see Fig. 1 for examples). Beyond segmentation masks, such applications often also require centerlines with line-topology, i.e., the centerlines should not contain interruptions, isolated pieces, or width (Fig. 2). Centerlines with line-topology are usually represented as sequences of coordinates.

While non-tubular object such as cars often have relatively high variance of appearance (e.g., color or texture) but low variance of shapes, tubular structures typically have relatively low variance of appearance but high variance of
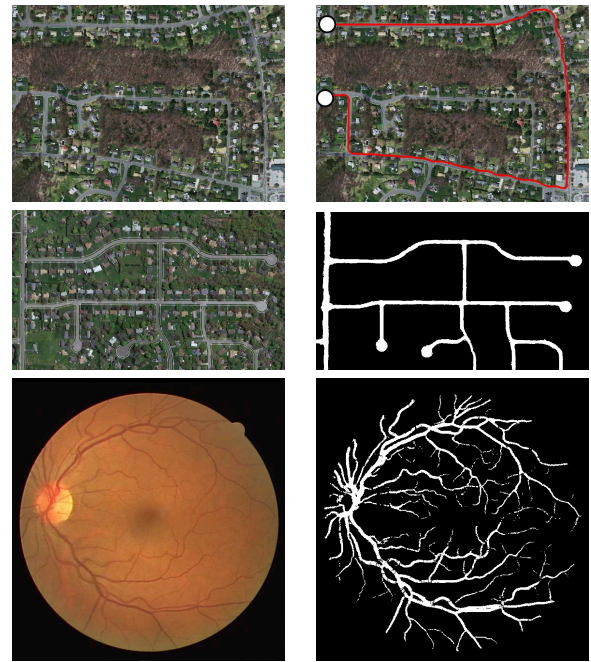
(a) Input image   (b) Our results

Figure 1. Examples of application areas of tubular structure segmentation. First row: Finding path between two points in a satellite image (navigation). Second row: Segmentation of road network in a satellite image. Third row: Segmentation of the vessel tree in a image of retinal vessels.
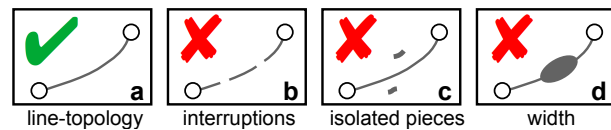

Figure 2. Illustration of line-topology and non-line-topology.

shapes. For example, blood vessels in medical images of the same imaging modality typically have similar intensity distributions, but they may contain very irregular shapes caused by pathology. General segmentation methods, such as U-Net [23], are widely used also to segment tubular structures, but these methods do not take the special properties of tubular structures into account, and cannot sat-

isfy certain topological constraints, such as connectivity of two points. To obtain centerlines with line-topology, post-processing is usually required. Also, the amount of annotations needed to train such methods is usually quite high.

In contrast, using minimal path methods, such as Dijkstra's algorithm [9] or the fast marching method [7, 26], points on the centerlines are obtained explicitly, and line-topology is guaranteed naturally. However, to obtain segmentation masks of the tubular structures, often an additional post-processing step is necessary [11]. Alternatively, a new dimension can be introduced into the parameter space of the minimal path method to represent the radius [15, 22, 6], but this leads to significantly higher computational cost. In the recent method of Path-CNN [16], a CNN is integrated into Dijkstra's algorithm to obtain centerlines and masks without post-processing. However, the masks generated using this method is relatively imprecise, since training is only performed using samples along user-specified centerlines, i.e., samples which are inside the ground truth masks of tubular structures but do not lie exactly on centerlines are not used. Furthermore, the samples used to train this CNN are often different from samples which are actually encountered during the inference.

We propose a method to determine segmentation masks and centerlines with line-topology for tubular structures in a unified way. Our method uses a similar framework as [16], i.e., we integrate a CNN classifier into the minimal path method. However, we use a novel technique for creating samples out of existing annotations (see Fig. 6 and 7 for an overview of differences). Instead of using the annotation (i.e., ground truth mask) directly and training the model only once, we introduce an *iterative training scheme*. In each iteration, we apply minimal path method with the CNN classifier trained in the previous iteration to generate new samples out of the annotations. These samples are used to re-train the CNN classifier in the current iteration. The ground truth masks do *not* change, but by iteratively re-training and re-sampling, we are able to extract *different* samples which are much more realistic and better *tailored* for the minimal path method. To our knowledge, this is the first approach to apply iterative training to minimal path method, and to use minimal path method itself to improve samples during the training phase. In this way, we better exploit the properties of the minimal path method, and achieve more precise masks and centerlines on three datasets. Also, our method needs only very few annotated images, making it especially effective for medical image analysis, for which the amounts of annotations are often limited.

2. Related Work

Topology enhancement Local features in the neighborhood of a pixel can be used to estimate the *tubularity measure*, i.e., probability that this pixel belongs to a tubular structure. Several widely used methods employ differential measures based on Hessian matrix [10, 25] or flux [31]. While most of these methods rely on hand-tuned parameters, there also exist learning-based methods, such as [28]. These methods are usually used by other methods as feature extraction step to achieve segmentation masks or centerlines. More recently methods [12, 27] use deep learning to enhance tubular structures, but post-processing is still needed to achieve line-topology of the result.

Approaches based on minimal path Minimal path methods are commonly used to extract centerlines of tubular structures with line-topology. These methods often suffer from the short cut problem, i.e., the centerlines may leak out the tubular structures. This can especially happen when the tubular structure has high curvature, or two tubular structures are very close to each other, or the tubularity measure cannot differentiate between foreground and background. The short cut problem has been approached using a variety of methods [4, 5, 13, 32, 15, 22, 6]. The recent method [16] integrates a CNN classifier into the minimal path method to avoid several types of short cut problems. However, the classifier in [16] is only trained once, and minimal path method is only used during inference. In contrast, our method is trained iteratively, and minimal path method is used both during training and inference. Furthermore, in [16] a hand-tuned tubularity feature [10] is used at initialization, but in later steps other learning-based features are used. In contrast, we use learning-based features throughout all steps, thus it is not necessary to tune parameters manually for [10].

Training scheme Instead of using annotations directly, [21] uses more realistic samples provided by a segmentation network. We use a similar idea, but in our approach, the training process is iterative, i.e., the classifier is trained using samples which are generated in the previous iteration *using the classifier itself* in conjunction with the minimal path method. The classifier in [21] is only used to classify samples created by *another* segmentation network, while our classifier is an *integral part* of the segmentation method itself. Furthermore, [21] relies on other tubularity features, which require large amounts of training data. In contrast, our method does not depend on other features and yields high-quality results even for small amounts of training data. Finally, [21] does not achieve segmentation masks.

Road segmentation There are recent methods for the segmentation of road networks [21, 2, 3], and some of them use minimal path method as a component [21]. However, our focus is on the methodology itself, and our method can also be applied to other images, such as medical images.

3. Minimal Path Method with Integrated CNN

Minimal path methods can be naturally used to segment centerlines of tubular structures. These methods are efficient since they are usually based on Dijkstra's algorithm [9], which is a greedy algorithm, or its continuous counterpart, the fast marching method [7, 26]. Our approach is based on Dijkstra's algorithm, but the same idea also applies to the fast marching method.

3.1. Centerline Extraction

Suppose a graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ is induced by image I, where $\mathcal{V}$ and $\mathcal{E}$ are the sets of *vertices* and *edges*, respectively. Each vertex v corresponds to a pixel in I, so that $\mathcal{G}$ can be imagined as a regular grid over the image. The example in Fig. 3 shows such an image grid. Neighboring pairs of vertices u and v are connected by an edge $e_{u,v}$, which has a weight $\mathbf{w}[u, v]$ depending on the underlying image feature. For the tubular structure to be segmented, such as the cyan object in Fig. 3, we assume that the start point $\mathbf{x}_s$ and end point $\mathbf{x}_e$ are given. The requirement for given start and end points is not a severe limitation, since in many applications these points can be determined using other methods automatically, such as the method in [21]. Finding the centerline of this tubular structure is equivalent to determining a path γ, i.e., a sequence of vertices $\{v_0, v_1, \ldots, v_{|\gamma|}\}$, from $\mathbf{x}_s$ to $\mathbf{x}_e$ in such a way that the total weight $\sum_{i=1}^{|\gamma|} \mathbf{w}[e_{v_{i-1},v_i}]$ of γ is minimized among all possible paths between $\mathbf{x}_s$ and $\mathbf{x}_e$. This path with minimum total weight is referred to as the *minimal path*.

The minimal path is detected by computing the minimum total weight $d(u)$ of paths between $\mathbf{x}_s$ and any other vertex u in the graph. In Fig. 3, an orange curve partitions $\mathcal{V}$ into two regions: In the region $\mathcal{S}$, which contains $\mathbf{x}_s$, $d(u)$ is already determined for all vertices, while in the region $\mathcal{Q}$, $d(u)$ is not yet determined. For every vertex u inside $\mathcal{S}$, its *predecessor* is recorded using the function $\pi(u)$. By repeatedly looking up the predecessor, one can always back-trace to $\mathbf{x}_s$. Paths to $\mathbf{x}_s$ are illustrated as blue lines exemplarily for several vertices (black dots) in Fig. 3a. In each iteration of the minimal path method, one vertex is added to $\mathcal{S}$, so that $\mathcal{S}$ always grows towards $\mathbf{x}_e$, while $\mathcal{Q}$ always shrinks. Once $\mathcal{S}$ reaches $\mathbf{x}_e$, the minimal path can be found by tracing from $\mathbf{x}_e$ back to $\mathbf{x}_s$ using π (red line in Fig. 3b). For details, we refer to standard texts such as [8, 14].

3.2. Segmentation Mask with Path-CNN

In the Path-CNN method [16], an additional step is introduced into Dijkstra's algorithm. Each time when a vertex v_0 is added to $\mathcal{S}$, a short path γ_L (magenta line in Fig. 3a) is back-traced using π. γ_L has only a fixed length and usually does not end in $\mathbf{x}_s$. Then an image patch P (red box) is cropped around γ_L. If γ_L is curved, then P is also curved.
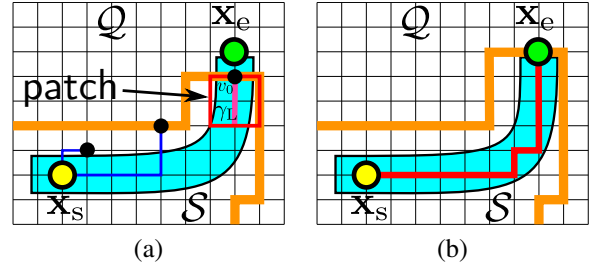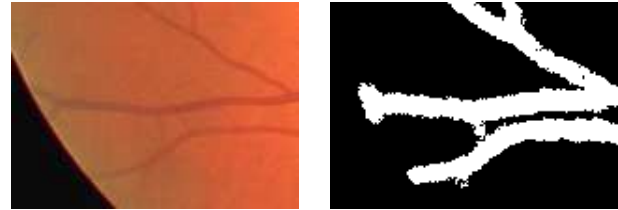

(a)　　　　　　(b)

Figure 3. Segmentation of a tubular structure using Dijkstra's algorithm. (a) An intermediate step. (b) Path extraction: The set $\mathcal{S}$ has reached the end point $\mathbf{x}_e$, and the minimal path can be extracted using the predecessor function π.

(a) Input image　　　　(b) Segmentation mask

Figure 4. Result of Path-CNN on an image of retinal vessels from DRIVE. (a) Input image. (b) The segmentation mask is too large.

P is referred to as *sample*. Then P is rectified, i.e., warped into a rectangular patch P_w so that γ_L becomes a *straight vertical line* in the middle of P_w. The patch in Fig. 3a is the same as its rectified version. After that, P_w is classified by a CNN classifier C. Further steps depend on the classification result, and more details are given in Sect. 4.2 below. Since Path-CNN not only computes π but also classifies each pixel in the image as foreground or background, consequently its result contains not only *centerlines* of tubular structures, but also a binary *segmentation mask* of them. However, since Path-CNN is trained directly using samples along ground truth centerlines, its classification performance for pixels off the centerlines is not strong, and often the boundary of the tubular structures is not delineated exactly. An example is shown in Fig. 4, in which the mask for retinal vessels generated by Path-CNN is too large.

4. Iterative Training with Tailored Samples

In this section, we first present our concept of samples in the context of minimal path method, and describe the differences between samples and annotations. After that, we show details of our schemes for inference and training.

4.1. Sample and Annotation

An important reason for the inexact boundary in the segmentation mask in Fig. 4 is the discrepancy between samples used for training and samples encountered actually during inference. Similar to the description in Sect. 3.2, a *sample* in our method is an image patch defined by its centerline

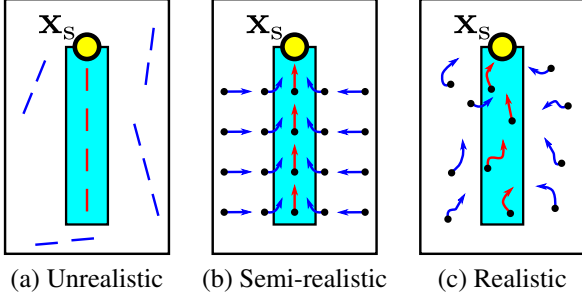| (a) Unrealistic | (b) Semi-realistic | (c) Realistic |

Figure 5. Sampling schemes. Positive and negative samples are shown as red and blue line segments. (a) In Path-CNN [16], samples are drawn along lines (annotations). Such samples are unrealistic. (b) In the initial iteration of our method, samples are drawn using the predecessor function π computed using the ground truth mask. They are not very realistic, but better than samples in (a). (c) In the main iterations of our method, samples are drawn using π, which is computed using the CNN classifier trained in the previous iteration. Samples drawn in this way are more realistic.

and a specific width. Instead of using annotations such as segmentation masks or lines directly, we propose to *derive* samples with much more semantic information. Otherwise, the samples may be generated inappropriately. For example, in the sampling scheme of Path-CNN, both positive and negative samples (image patches) are drawn along annotations (lines), i.e., annotations are used *directly*. The centerlines of such positive and negative samples are shown as red and blue line segments in Fig. 5a, respectively. The positive training samples lie on the centerlines, but during the minimal path computation (i.e., inference), real samples which should be classified as tubular structure are rarely exactly on the centerlines. To deal with this discrepancy, we introduce an *iterative* training scheme to generate training samples which are specifically *tailored* for minimal path methods, without changing existing annotations. Compared with the training samples in [16], our training samples contain more semantic information to improve the minimal path method, since such training samples are likely to be more similar to samples encountered during inference (Fig. 5b and c). In the following, we describe the inference method first, because it is not only used during the inference phase, but also used during the training phase.

4.2. Inference

The main steps of our inference method and a comparison with previous methods is given in Fig. 6. In common minimal path methods, features are computed only once before the main loop of Dijkstra's algorithm. In Path-CNN, hand-tuned features are used to initialize the graph. But during the main loop, learning-based features are extracted using a CNN classifier to update the graph. Our method, which is similar to Path-CNN but does not use hand-tuned features, is described as Algorithm 1.
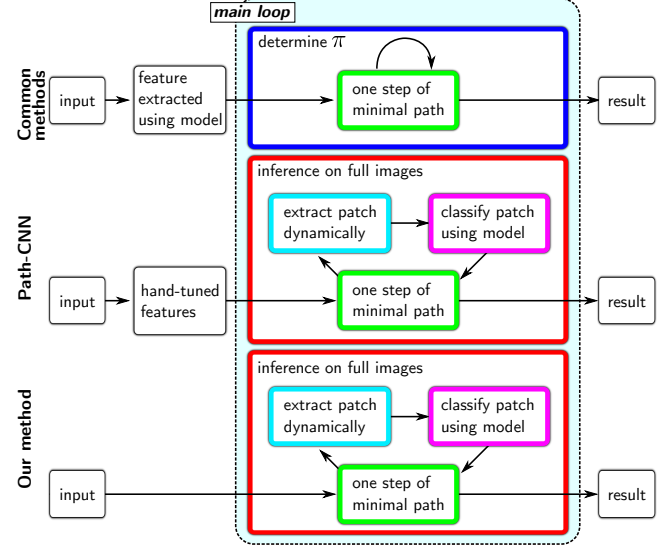

Figure 6. Comparison of the inference step in different minimal path methods. The operations (boxes with colored border) correspond to functions in Algorithm 1 with the same name.

Algorithm 1: InferenceOnFullImages: Minimal path method with dynamic weight adaptation.

Input: Image I, start point $\mathbf{x}_s$, CNN classifier C
Output: predecessor function π, segmentation mask M

1 **for** *each* $e \in \mathcal{E}$ **do** $\mathbf{w}[e] \leftarrow \epsilon$;
2 **for** *each* $\mathbf{x} \in \mathcal{V}$ **do**
3    $\pi[\mathbf{x}] \leftarrow$ **none**; $d[\mathbf{x}] \leftarrow \infty$; $M[\mathbf{x}] \leftarrow FG$;
4 $d[\mathbf{x}_s] \leftarrow 0$; $\mathcal{Q} \leftarrow \mathcal{V}$; $\mathcal{S} \leftarrow \varnothing$;
5 **while** $\mathcal{Q} \neq \varnothing$ **do** // Main loop
6    $u \leftarrow \arg\min_{\mathbf{x} \in \mathcal{Q}} d(\mathbf{x})$;
7    $\mathcal{Q} \leftarrow \mathcal{Q} - \{u\}$; $\mathcal{S} \leftarrow \mathcal{S} \cup \{u\}$;
8    $P \leftarrow$ ExtractPatchDynamically(u, π, I);
9    $l \leftarrow$ ClassifyPatchUsingModel(C, P);
10    $M[u] = l$;
11   **for** *each* $v \in \mathcal{N}(u) \cap \mathcal{Q}$ **do**
12   **if** $l = BG$ **then**
13        $\mathbf{w}[e_{u,v}] \leftarrow \mathbf{w}[e_{u,v}] + \mathbf{w}_p$;
14   **if** $d[v] > d[u] + \mathbf{w}[e_{u,v}]$ **then**
15        $d[v] \leftarrow d[u] + \mathbf{w}[e_{u,v}]$;
16        $\pi[v] \leftarrow u$;
17 **return** π, M

The input of our algorithm includes the input image, a start point, and a CNN classifier. We initialize all edge weights with a small constant ϵ (Line 1). The functions π and d, an empty segmentation mask M, as well as regions $\mathcal{S}$ and $\mathcal{Q}$ are initialized between Line 2 and 4. Line 5 to 16 are the main loop of the inference, which only stops if π is

determined for all pixels in the image, i.e., if $\mathcal{Q}$ is empty. In each iteration, the pixel u with currently minimum d value is moved from $\mathcal{Q}$ to $\mathcal{S}$ (Line 6 and 7), and a patch P starting at u is extracted *dynamically* (Line 8), i.e., this patch can only be extracted during the main loop but not before it. Patch extraction is also illustrated in Fig. 3a. The patch P is then *rectified*, i.e., transformed into a rectangular patch so that its centerline becomes straight and vertical, and classified by the CNN classifier (Line 9). The classification label is added into the mask M (Line 10). In case of background label, the weights between u and its neighbors inside $\mathcal{N}$ (the 4-neighborhood on the image grid) and $\mathcal{Q}$ are increased by a high penalty $\mathbf{w}_\mathrm{p}$ (Line 13), so that it is less possible for the minimal path to run through u. After that, functions π and d are updated as in the usual version of Dijkstra's algorithm (Line 14 to 16). Finally, π and M are returned. M is the segmentation mask of *all* the tubular structures in the image, and π can be used to extract the minimal path between start point $\mathbf{x}_\mathrm{s}$ and *any* other point in the image, especially the end point $\mathbf{x}_\mathrm{e}$. The centerlines extracted using π have line-topology, while M does not have this property.

4.3. Training

To use more realistic samples for training, we apply an iterative scheme. The main steps and comparison with other training methods are shown in Fig. 7. In particular, while common methods and Path-CNN draw samples directly from the annotations, we use minimal path method itself to get initial samples and then repeatedly refine the samples. In the N-th iteration, the minimal path method with trained classifier C_{N-1} from the $(N-1)$-th iteration is applied to the training images to extract samples. Details of our method is given in Algorithm 2.

For simplicity reasons, we assume there is only one training image and one validation image, each with ground truth mask. In contrast to Path-CNN, hand-tuned features are not used in our method. Instead, in determine_π (Line 1) we use *ground truth* mask to define initial weights for edges in the graph: If both vertices of an edge belong to foreground, then weight of the edge is set to 1. Otherwise, it is set to a large positive constant. This graph is used to obtain π *without* using any classifier. We then use π to extract *tailored samples* from the *training image* I_tr (Line 2): Starting from any pixel $\mathbf{x}$, we generate a path γ_L by back-tracing using π for a fixed number of steps, and crop a curved image patch P along γ_L. The sample P generated in this way is positive if $\mathbf{x}$ in in foreground. Otherwise, it is negative. The path γ_L is a segment on the minimal path between $\mathbf{x}$ and $\mathbf{x}_\mathrm{s,tr}$. γ_L is computed not using weights based on image features, but using weights derived from ground truth, so its shape is still quite regular (Fig. 5b), but it is more realistic than samples extracted using manually specified lines (Fig. 5a). With these samples, an initial classifier is trained (Line 5).
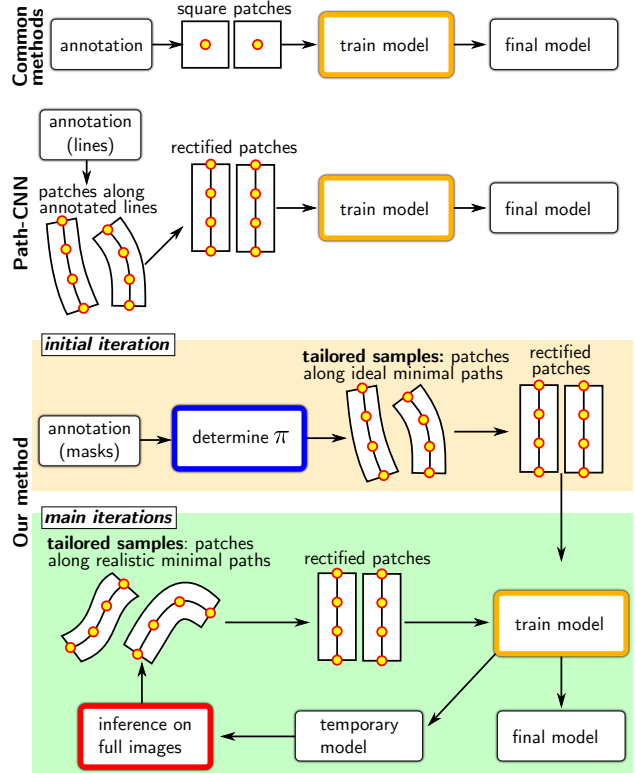

Figure 7. Comparison of the training step in different minimal path methods. Each yellow circle with red border corresponds to a pixel. The operations (boxes with colored border) correspond to functions in Algorithm 2 with the same name. The red and blue boxes are the same as the red and blue boxes in Fig. 6.

The steps in Line 6 and 7 is very similar to the steps in Line 1 and 2, but now π is computed using training image and classifier, instead of using ground truth. The samples generated in this way are getting even closer to the realistic samples, since both of them are generated by applying the same inference method, i.e., minimal path method with integrated CNN. These samples are illustrated in Fig. 5c and Fig. 7. The trained classifier is applied for inference on the validation image (Line 8) to compute a segmentation mask M. The iteration terminates if the Dice score between M and ground truth does not increase anymore (Line 9 to 13).

5. Experimental Results

We applied our method to three datasets and compared the results with seven previous methods. The accuracy of centerlines and segmentation masks is studied below, along with qualitative results.

5.1. Datasets

Three public datasets EPFL [29], MRD [20], and DRIVE [30] were used for evaluation. In most of our experiments, the training sets consisted of quite small numbers of images.

Algorithm 2: Iterative training scheme

Input: Train/val images I_{tr} and I_{v} with ground truth GT_{tr}, GT_{v}, start points $\mathbf{x}_{\mathrm{s,tr}}$, $\mathbf{x}_{\mathrm{s,v}}$, maximum number of iterations N

Output: Classifier C

         `// Initial iteration`
1   $\pi \leftarrow$ determine_$\pi(GT_{\mathrm{tr}}, \mathbf{x}_{\mathrm{s,tr}})$;
2   $\mathcal{X} \leftarrow$ CreateTailoredSamples$(\pi, I_{\mathrm{tr}}, GT_{\mathrm{tr}})$;
3   $d_{\mathrm{prev}} \leftarrow 0$;
4   **for** $i \leftarrow 1$ **to** N **do** `// Main iterations`
5       $C \leftarrow$ TrainModel$(\mathcal{X})$;
6       $\pi, M \leftarrow$ InferenceOnFullImages$(I_{\mathrm{tr}}, \mathbf{x}_{\mathrm{s,tr}}, C)$;
7       $\mathcal{X} \leftarrow$ CreateTailoredSamples$(\pi, I_{\mathrm{tr}}, GT_{\mathrm{tr}})$;
            `// Validation`
8       $\pi, M \leftarrow$ InferenceOnFullImages$(I_{\mathrm{v}}, \mathbf{x}_{\mathrm{s,v}}, C)$;
9       $d_{\mathrm{cur}} \leftarrow$ ComputeDice(M, GT_{v});
10   **if** $d_{\mathrm{cur}} > d_{\mathrm{prev}}$ **then**
11         $d_{\mathrm{prev}} \leftarrow d_{\mathrm{cur}}$;
12   **else**
13   **break**

14 **return** C

Dataset	Dijk-CNN	Prog-CNN	Path-CNN	Ours
EPFL	2.91	2.52	1.71	**1.41**
MRD	7.36	5.12	3.65	**3.54**

Table 1. Errors for centerlines measured using Eq. (1). Unit: Pixel.

the best methods to segment retinal vessels [1]). In each experiment, all methods were trained using the same images.

Dijk-CNN is the classical Dijkstra's algorithm, and edge weights in the graph are determined using a CNN before Dijkstra's algorithm is applied. Prog-CNN is the progressive minimal path method, which also uses a CNN to precompute edge weights. In contrast to Dijk-CNN, Prog-CNN extracts a short *path segment* γ in every iteration and adapts the edge weights on-the-fly, depending on the CNN-based features along γ. Path-CNN is an extension of Prog-CNN: In each iteration, not a path segment but an *image patch* is extracted (Fig. 3a) to adapt edge weights. The inference step of our method is similar to Path-CNN (Fig. 6), although the training step is very different (Fig. 7). The CNN used in Dijk-CNN, Prog-CNN, Path-CNN and our method was MobileNetV2 [24]. For Path-CNN and our method, we used a fixed patch size of 31×31 for all experiments and datasets. For Dijk-CNN and Prog-CNN, the usual data augmentation operations were used, while for Path-CNN and our method we used only horizontal and vertical flips, and a small rotation of ± 5 degrees, since further augmentations like rotation or random cropping would violate the assumption of these methods that the centerline of a rectified patch must be a *straight vertical line* in the middle of the patch (Sect. 3.2). We compared these minimal path methods in terms of the accuracy of the centerline points. To evaluate the accuracy of the segmentation masks, our method was compared with Path-CNN, U-Net, softDice, clDice, and FR-UNet.

5.3. Accuracy of the Centerlines

We used the mean error between result and ground truth to evaluate methods which extract lines as sequences of points. This error measure is defined as

$$\overline{\mathbf{e}}_{cl} = \frac{1}{N} \sum_{\gamma} \sum_{\mathbf{x}_i \in \gamma} |\mathbf{x}_i - \mathbf{x}_{\mathrm{gt}(i)}|, \tag{1}$$

where N is the total number of control points on the extracted centerlines, γ is a centerline, and $\mathbf{x}_{\mathrm{gt}(i)}$ is the ground truth point which is closest to a given point $\mathbf{x}_i$ on γ.

To compare the approaches, we selected 100 paths out of 11 test images of EPFL, and 650 paths out of 49 test images of MRD. The results are summarized in Table 1. All four methods used features obtained using the same CNN architecture, but the performance was rather different. In Dijk-CNN, the CNN is applied to each pixel individually, and the relationship between neighboring pixels is only taken into account by convolutional filters. Since tubular struc-

EPFL This dataset contains 14 satellite images of roads. Binary segmentation masks and centerlines for all roads are provided. The centerlines are presented as sequences of image coordinates. We used 3 images for training and the remaining 11 images for testing.

MRD This dataset consists of 1157 satellite images of roads, divided into a training set of 1108 images, and a test set of 49 images. Only the road centerlines are provided, but they are presented as binary masks instead of sequences of coordinates. We dilated the centerlines slightly and used the dilated region as an approximation of the foreground mask. Regions close to the foreground mask were not used for training. The regions further away from the foreground were used as background mask. We used 10 image patches for training and apply the trained model to the test set.

DRIVE In this dataset, there are 40 images of retinal vessels. The dataset is divided into 20 training images and 20 test images. Binary masks for the vessels are provided without centerlines.

5.2. Baselines

Beside minimal path methods, U-Net and its variants are also widely used to segment tubular structures. Thus, in addition to minimal path methods, i.e., Dijk-CNN [9], Prog-CNN [17], and Path-CNN [16], we also compared our method with the original U-Net [23], U-Net with softDice loss [19] and clDice loss [27], and FR-UNet [18] (one of
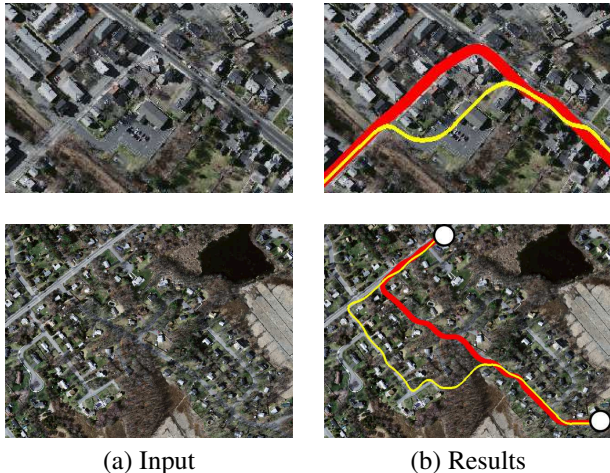
(a) Input   (b) Results

Figure 8. Determination of the path between given start and end points on two images from MRD. Result of Path-CNN is shown as yellow line, and our result is shown as red line. Path-CNN made several wrong turns, while our method found the correct paths.

tures are typically thin and long, they are difficult to capture using convolutional filters alone, leading to high errors. Prog-CNN computes features explicitly for path segments and yielded mean errors of 2.52 and 5.12 pixels for EPFL and MRD, respectively, while in both datasets, the width of most roads was over 10 pixels. By using features of rectified patches, Path-CNN and our method were able to incorporate more contextual information, and further reduced the errors to less than 2 pixels for EPFL and less than 4 pixels for MRD. Overall, our method achieved the lowest errors.

By using samples tailored for minimal path methods, our method performed better in complex environments. This is demonstrated in the qualitative comparison between Path-CNN and our approach in Fig. 8. In the upper row, Path-CNN made a wrong turn at a crossing, while in the lower row it failed to follow the correct road which had strong shadow. In contrast, our method dealt well with both cases.

5.4. Accuracy of the Segmentation Masks

Dice score is a commonly used metric to compare two masks M_1 and M_2. It is defined as

$$S_{\text{Dice}} = 2 \cdot \frac{|M_1 \cap M_2|}{|M_1| + |M_2|}, \qquad (2)$$

where $|M_1 \cap M_2|$ is the number of pixels in the overlapping region. For EPFL and DRIVE, we computed the mean Dice scores (MDS) of the test sets, i.e., the Dice scores are computed for each test image individually and then averaged.

The results are summarized Table 2. For EPFL, we achieved the highest MDS. Also, the boundary of tubular structures was delineated more precisely using our method, as illustrated in Fig. 9. The masks obtained using Path-CNN

Dataset	Path-CNN	U-Net	softDice	clDice	FR-UNet	Ours
EPFL	0.6322	0.9076	0.8803	0.9052	0.8847	**0.9125**
DRIVE	0.5927	0.7815	0.7644	0.7888	**0.8208**	0.8065

Table 2. Dice score for segmentation measured using Eq. (2).

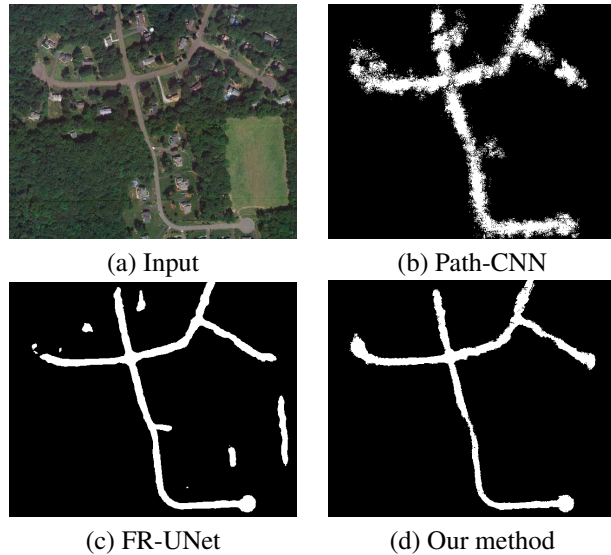(a) Input   (b) Path-CNN

(c) FR-UNet   (d) Our method

Figure 9. Results for a test image from EPFL. All three models are trained with only 3 images. Path-CNN could not determine the boundary precisely, and FR-UNet found false positive regions.

have quite noisy boundary, while in the result of FR-UNet, several background regions are segmented as foreground. In contrast, our method successfully avoided false positive regions, and achieved smoother boundary than Path-CNN. For DRIVE, our MDS was slightly below FR-UNet but higher than all other methods. In addition, we ran a further experiment using only 3 training images and all test images on DRIVE. Our MDS was still as high as 0.7578, while the MDS of U-Net and Path-CNN dropped to 0.5217 and 0.5470, respectively. A comparison is shown in Fig. 10. Our method not only achieved more precise boundary, but also made the segmentation of tubular structures more robust in regions with low image quality, especially when the training data was scarce. In the example in Fig. 10a, a large part on the vessel tree is missing in the results of Path-CNN and U-Net, especially in the right half of the image, where the image contrast is relatively low. In contrast, our method correctly segmented most vessels, even the small ones. These results were achieved using only 3 training images, which is much less than the amount of annotations needed by most recent methods. It shows that our method is especially interesting for medical image analysis, where it is often difficult to obtain enough annotated images.

5.5. Effects of Iterative Training

The Dice scores on the validation set after each training iteration are plotted in Fig. 11. For all three datasets,

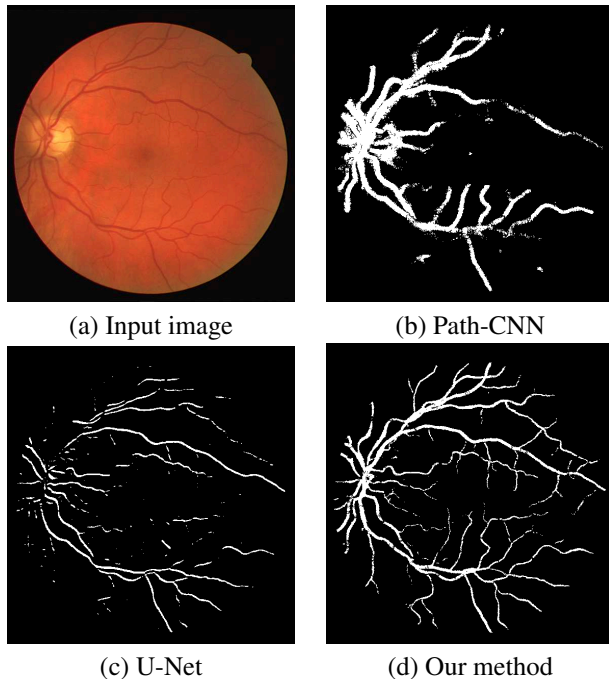(a) Input image   (b) Path-CNN

(c) U-Net   (d) Our method

Figure 10. Results for a test image from DRIVE. All three models were trained with only 3 images. Path-CNN and U-Net both failed to segment vessels in image regions with low contrast, while our method successfully segmented most vessels.
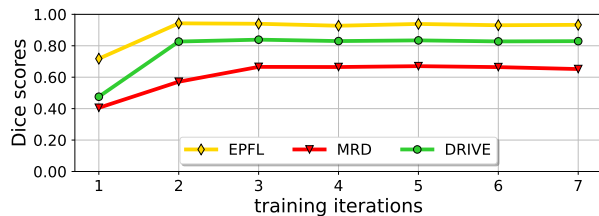

Figure 11. Dice scores on the validation set after each iteration.

the Dice score increases fast in the first iterations, and after three iterations there is no significant change anymore. Note that there are no ground truth masks for MRD, so we used dilated centerlines as pseudo-masks to train our method (Sect. 5.1). Therefore MRD has lower Dice score than the other datasets in Fig. 11. A qualitative examples in Fig. 12 shows the improvement of segmentation masks due to the iterative training scheme, using a test image in MRD. After the first iteration, the boundary of the roads is still quite noisy, and there are several false positives in the background. Also, the width of the roads is imprecise and fluctuates a lot. After the third iteration, false positives are reduced significantly, and an overlay with the original image shows that the shape of the roads is determined well, even though our method is trained only using pseudo-masks.

5.6. Implementation and Run Time

The major part of our C++ implementation runs on CPU, only the step of patch classification runs on GPU. It takes

(a)   (b)   (c)
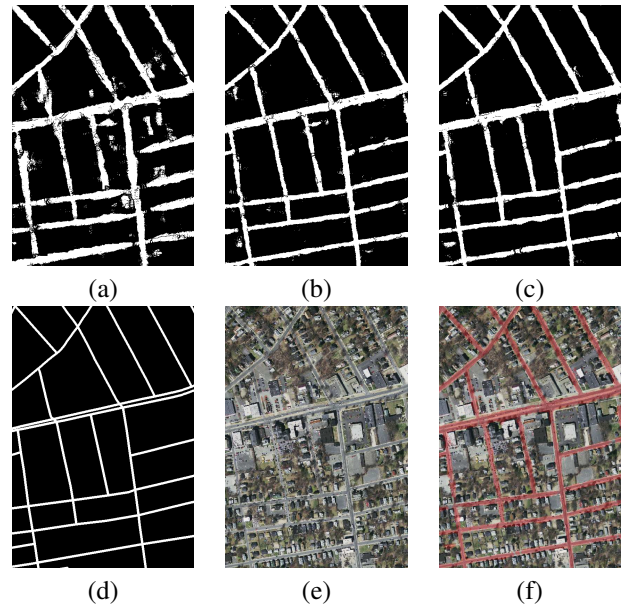
(d)   (e)   (f)

Figure 12. Improvement through the iterative training scheme for MRD. Segmentation mask after the first, second, and third iteration are shown in (a), (b), and (c) respectively. (d) Ground truth (centerline only). (e) Input image. (f) Overlay of the result after the third iteration on the input image.

about 15 minutes to segment an image of the size 600×600 using a 3.4 GHz Intel i7 CPU and NVIDIA GTX 1070 GPU.

6. Conclusion

We proposed a minimal path method to segment tubular structures and extract their centerlines. An *iterative training scheme* is introduced so that existing annotations can be used more effectively without modification. In our framework, minimal path method is not only used for segmentation and centerline extraction, but also for the generation of realistic training samples. Our minimal path method contains an integrated CNN classifier. Annotations are not directly used as samples to train this classifier. Instead, the minimal path method is used along with the classifier from the previous iteration to extract training samples out of existing annotations. Trained with these samples, the classifier in turn is used to steer the minimal path method in the next iteration to generate better samples. Training samples obtained in this way are semantically more meaningful and better *tailored* for the minimal path method, so our model is trained more effectively. Experimental results show that by using this training scheme, our approach outperforms several recent methods and yields high-quality centerlines and segmentation masks for tubular structures on several datasets in different domains, even in cases of very limited amounts of annotations. We believe that our method is especially useful for application areas in which annotations are difficult to obtain, such as medical image processing.

References

[1] Leaderboard of Retinal Vessel Segmentation on DRIVE. https://paperswithcode.com/sota/retinal-vessel-segmentation-on-drive, 2023. [Online; accessed 08. March 2023]. 6

[2] Favyen Bastani, Songtao He, Sofiane Abbar, Mohammad Alizadeh, Hari Balakrishnan, Sanjay Chawla, Sam Madden, and David DeWitt. RoadTracer: Automatic Extraction of Road Networks from Aerial Images. In *IEEE Conference on Computer Vision and Pattern Recognition*, 2018. 2

[3] Anil Batra, Suriya Singh, Guan Pang, Saikat Basu, C.V. Jawahar, and Manohar Paluri. Improved road connectivity by joint learning of orientation and segmentation. In *IEEE Conference on Computer Vision and Pattern Recognition*, 2019. 2

[4] Fethallah Benmansour and Laurent D. Cohen. Fast object segmentation by growing minimal paths from a single point on 2d or 3d images. *Journal of Mathematical Imaging and Vision*, 33(2):209 – 221, 2009. 2

[5] Fethallah Benmansour and Laurent D. Cohen. Tubular structure segmentation based on minimal path method and anisotropic enhancement. *International Journal of Computer Vision*, 92:192–210, 2011. 2

[6] Da Chen, Jiong Zhang, and Laurent D. Cohen. Minimal paths for tubular structure segmentation with coherence penalty and adaptive anisotropy. *IEEE Transactions on Image Processing*, 28(3):1271–1284, Mar. 2019. 2

[7] Laurent D. Cohen and Ron Kimmel. Global minimum for active contour models: A minimal path approach. *International Journal of Computer Vision*, 24(1):57–78, 1997. 2, 3

[8] Thomas H. Cormen, Charles E. Leiserson, Ronald L. Rivest, and Clifford Stein. *Introduction to Algorithms (Third Edition)*. MIT Press, 2009. 3

[9] E. W. Dijkstra. A Note on Two Problems in Connection with Graphs. *Numerische Mathematik*, 1:269–271, 1959. 2, 3, 6

[10] Alejandro F. Frangi, Wiro J. Niessen, Koen L. Vincken, and Max A. Viergever. Multiscale vessel enhancement filtering. In *Medical Image Computing and Computer-Assisted Intervention*, 1998. 2

[11] M. Freiman, N. Broide, M. Natanzon, L. Weizman, E. Nammer, O. Shilon, J. Frank, L. Joskowicz, and J. Sosna. Vesselscut: A graph based approach to patient-specific carotid arteries modeling. In *Proceedings of the Workshop on 3D Physiological Human*, 2009. 2

[12] Xiaoling Hu, Fuxin Li, Dimitris Samaras, and Chao Chen. Topology-preserving deep image segmentation. In *Advances in Neural Information Processing Systems*, 2019. 2

[13] Vivek Kaul, Anthony Yezzi, and Yichang Tsai. Detecting curves with unknown endpoints and arbitrary topology using minimal paths. *IEEE Transactions on Pattern Analysis and Machine Intellegence*, 34(10):1952–1965, 2012. 2

[14] Jon Kleinberg and Eva Tardos. *Algorithm Design*. Pearson Education, 2005. 3

[15] Hua Li and Anthony Yezzi. Vessels as 4-d curves: Global minimal 4-d paths to extract 3-d tubular surfaces and centerlines. *IEEE Transactions on Medical Imaging*, 26(9):1213–1223, 2007. 2

[16] Wei Liao. Progressive minimal path method with embedded cnn. In *IEEE Conference on Computer Vision and Pattern Recognition*, 2022. 2, 3, 4, 6

[17] Wei Liao, Stefan Wörz, Chang-Ki Kang, Zang-Hee Cho, and Karl Rohr. Progressive minimal path method for segmentation of 2d and 3d line structures. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 40(3):696–709, Mar. 2018. 6

[18] Wentao Liu, Huihua Yang, Tong Tian, Zhiwei Cao, Xipeng Pan, Weijin Xu, Yang Jin, and Feng Gao. Full-Resolution Network and Dual-Threshold Iteration for Retinal Vessel and Coronary Angiograph Segmentation. *IEEE Journal of Biomedical and Health Informatics*, 26(9):4623–4634, 2022. 6

[19] Fausto Milletari, Nassir Navab, and Seyed-Ahmad Ahmadi. V-net: Fully convolutional neural networks for volumetric medical image segmentation. In *International Conference on 3D Vision*, 2016. 6

[20] Volodymyr Mnih. *Machine Learning for Aerial Image Labeling*. PhD thesis, University of Toronto, 2013. 5

[21] Agata Mosinska, Mateusz Kozinski, and Pascal Fua. Joint segmentation and path classification of curvilinear structures. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2020. 2, 3

[22] Mickael Pechaud, Renaud Keriven, and Gabriel Peyre. Extraction of tubular structures over an orientation domain. In *IEEE Conference on Computer Vision and Pattern Recognition*, 2009. 2

[23] Olaf Ronneberger, Philipp Fischer, and Thomas Brox. U-Net: Convolutional Networks for Biomedical Image Segmentation. In *Medical Image Computing and Computer-Assisted Intervention*, 2015. 1, 6

[24] Mark Sandler, Andrew Howard, Menglong Zhu, Andrey Zhmoginov, and Liang-Chieh Chen. Mobilenetv2: Inverted residuals and linear bottlenecks. In *IEEE Conference on Computer Vision and Pattern Recognition*, 2018. 6

[25] Yoshinobu Sato, Shin Nakajima, Nobuyuki Shiraga, Hideki Atsumi, Shigeyuki Yoshida, Thomas Koller, Guido Gerig, and Ron Kikinis. Three-dimensional multi-scale line filter for segmentation and visualization of curvilinear structures in medical images. *Medical Image Analysis*, 2(2):143–168, 1998. 2

[26] J. A. Sethian. *Level Set Methods and Fast Marching Methods: Evolving Interfaces in Computational Geometry, Fluid Mechanics, Computer Vision, and Materials Science*. Cambridge University Press, 1999. 2, 3

[27] Suprosanna Shit, Johannes C. Paetzold, Anjany Sekuboyina, Ivan Ezhov, Alexander Unger, Andrey Zhylka, Josien P. W. Pluim, Ulrich Bauer, and Bjoern H. Menze. cldice - a novel topology-preserving loss function for tubular structure segmentation. In *IEEE Conference on Computer Vision and Pattern Recognition*, 2021. 2, 6

[28] Amos Sironi, Bugra Tekin, Roberto Rigamonti, Vincent Lepetit, and Pascal Fua. Learning separable filters. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 37(1):94–106, 2015. 2

[29] Amos Sironi, Engin Tueretken, Vincent Lepetit, and Pascal Fua. Multiscale centerline detection. *IEEE Transactions on Pattern Analysis and Machine Intellegence*, 2015. 5

[30] Joes Staal, Michael D. Abramoff, Meindert Niemeijer, Max A. Viergever, and Bram van Ginneken. Ridge-based vessel segmentation in color images of the retina. *IEEE Transactions on Medical Imaging*, 23(4):501–509, 2004. 5

[31] Engin Tueretken, Carlos Becker, Przemyslaw Glowacki, Fethallah Benmansour, and Pascal Fua. Detecting irregular curvilinear structures in gray scale and color imagery using multi-directional oriented flux. In *IEEE International Conference on Computer Vision*, 2013. 2

[32] Johannes Ulen, Petter Strandmark, and Fredrik Kahl. Shortest paths with higher-order regularization. *IEEE Transactions on Pattern Analysis and Machine Intellegence*, 37(12):2588–2600, 2015. 2