Pick and Trace: Instance Segmentation for Filamentous Objects with a Recurrent Neural Network

Yi Liu(✉), Su Peng, Jeffrey Caplan, and Chandra Kambhamettu

University of Delaware, Newark, DE 19716, USA
yliu@udel.edu

Abstract. Filamentous objects are ubiquitous in biomedical images, and segmenting individual filaments is fundamental for biomedical research. Unlike common objects with well-defined boundaries and centers, filaments are thin, non-rigid, varying in shape, and often densely overlapping. These properties make it extremely challenging to extract individual filaments. This paper proposes a novel approach to extract filamentous objects by transforming an instance segmentation problem into a sequence modeling problem. Our approach first identifies filaments' tip points, and we segment each instance by tracing them from each tip with a sequential encoder-decoder framework. The proposed method simulates the process of humans extracting filaments: pick a tip and trace the filament. As few datasets contain instance labels of filaments, we first generate synthetic filament datasets for training and evaluation. Then, we collected a dataset of 15 microscopic images of microtubules with instance labels for evaluation. Our proposed method can alleviate the data shortage problem since our proposed model can be trained with synthetic data and achieve state-of-art results when directly evaluated on the microtubule dataset and P. rubescens dataset. We also demonstrate our approaches' capabilities in extracting short and thick elongated objects by evaluating on the C. elegans dataset. Our method achieves a comparable result compared to the state-of-art method with faster processing time. Our code is available at https://github.com/VimsLab/DRIFT.

Keywords: Instance Segmentation · Filament Tracing · Recurrent Neural Network

1 Introduction

Filamentous objects, such as microtubules, actin filaments, and blood vessels, play a fundamental role in biological systems. For example, microtubules (See Fig. 1) form part of cell cytoskeleton structures and are involved in various cellular activities such as movement, transportation, and key signaling events. To

Supplementary Information The online version contains supplementary material available at https://doi.org/10.1007/978-3-031-43993-3_61.

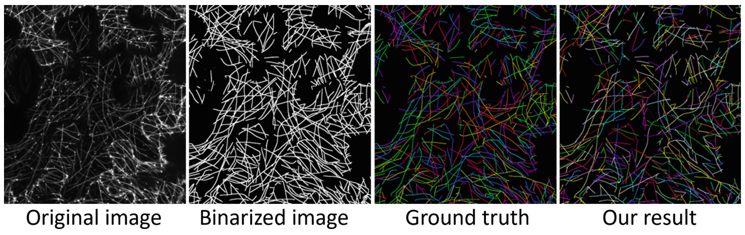

Fig. 1. Qualitative result of our method on a full resolution microscopic image of microtubules. The image has a size of 1376 × 1504 and contains 558 filaments.

understand the mechanism of filamentous objects, quantifying their properties, including quantity, length, curvature, and distribution, is fundamental for biological research. Instance segmentation is often the first step for quantitative analysis. However, as filaments are very thin, non-rigid, and usually span over the image intersecting each other, extracting individual filaments is very challenging.

Since the advent of deep learning, region-based instance segmentation methods [4,7,8,13,27], which segment instances within the detected bounding boxes, have shown remarkable performance on objects with well-defined centers and boundaries. However, filaments are non-rigid objects and span widely across the image. Each filament has a distinct shape varying in length and deformation, while segments of different filaments share a similar appearance. These properties make it extremely hard for region-based methods to detect the centers and bounding boxes for filaments and segment the target within the detected region.

Region-free methods utilize learned embedding [1,2,5,14] or affinity graph [6,19] to separate instances. These methods rely on pixel-level prediction to group instances and do not directly extract the complete shape of instances. Since filaments may be densely clustered and overlapping, these methods have difficulties in disentangling filaments in complicated scenes. Liu *et al.* [17] address the overlapping challenge by proposing an orientation-aware network to disentangle filaments at intersections, but it requires a heuristic post-processing to form instances. Hirsch *et al.* [9] predict dense patches representing instances' shape for each pixel and assemble them to form instances with affinity graph. Effectively extracting longer filaments requires predicting a larger shape patch for each pixel, leading to a longer computational time.

Lacking instance-level labels of filaments is another challenge. Most existing approaches for filaments extraction adopt traditional computer vision techniques such as morphological operations [29], template matching [28,29] and active contour [26]. These methods follow the segment-break-regroup strategy. They first obtain the binary segmentation, then break it at intersections and regroup the segments into filaments by geometric properties. The performance heavily relies on manual parameter tuning.

When a human tries to manually extract filaments in Fig. 1, directly pointing out each filament can be challenging. Instead, a human would first identify each

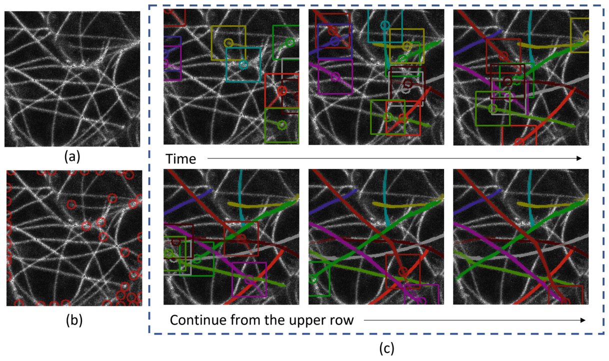

Fig. 2. Our methods mimic humans extracting filaments. (a). Image of microtubules. (b). Pick: tip points detection. (red circles are detected tips). (c). Trace: Successive stages in the process of tracing 10 instances. The boxes are the areas that the model is processing. The circles are patches' centers. (Color figure online)

filament's tip and then trace each filament. Inspired by this human behavior, we introduce Deep Recurrent Instance Filament Tracer (DRIFT) for instance segmentation on filamentous objects. Figure 2 shows an example of how DRIFT mimics a human and sequentially extracts filaments. As shown in Fig. 3, the pick module in DRIFT first detects all tip points as candidates. The recurrent neural network (RNN) based tracing module will 'look' at the patch around tip points, segment the object within the patch, and predict the next location. The trace module sequentially segments the object until a stop flag is predicted. The RNN learns where 'it' comes from, where 'it' is, and where 'it' goes next.

Our method is fundamentally different from [11,12,23,25]. Ren *et al.* [23] and Amaia [25] *et al.* use the RNN model to sequentially predict objects' bounding boxes, which are essentially region-based segmentation methods. Januszewski *et al.* [11,12] propose a flood-filling network to repeatedly perform segmentation within a set of manually defined patches to grow object masks. While the iterations in [11,12] are heuristic, our method learns to trace the filaments and sequentially segments the targets.

The major contributions of this work are as follows: (1). To our knowledge, our method is the first method that converts the instance segmentation into a sequence modeling problem. Our proposed method mimics human tracing and extracting individual filament, tackling the challenges of extracting filamentous objects. (2). We propose a synthetic filament dataset for training and evaluation. Our model trained on synthetic datasets can be applied to various real filament datasets and thus alleviate the data shortage problem. (3). We collected a dataset of 15 microscopic images of microtubules with instance labels for evaluation. (4). Our method is evaluated on four different datasets and compared with several competing approaches. Our method shows better or comparable results regarding accuracy and computational time.

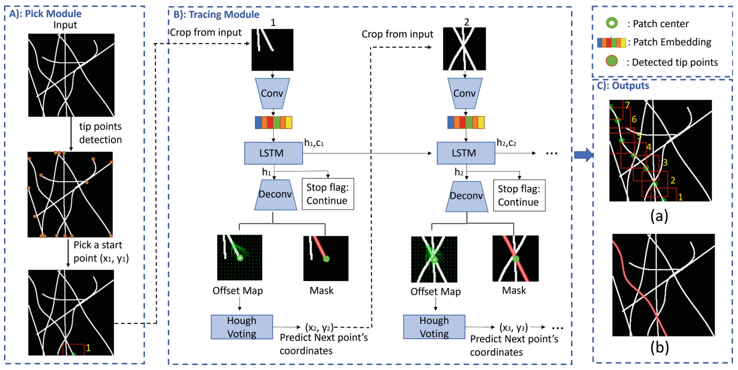

Fig. 3. Flowchart of DRIFT for instance segmentation on filaments. The example is shown with a synthetic filament image. A). Pick module for tip points detection. B). Tracing module for individual filament extraction. C-a). Red boxes are the sequence of patches processed. C-b). The final extracted filament. (Color figure online)

2 Method

Figure 3 shows the framework of our proposed method. The pick module detects tip points for all filaments. Then we crop the patches around tip points, and the tracing module will encode these patches into patch embeddings with a convolutional block. The tip point's patch embedding is used to initialize the hidden state of RNN. Then a decoder will decode the hidden state output and predict a stop flag and the object's mask within the current patch. The decoder also outputs an offset map, where each pixel predicts a vector pointing to the next center. We use the offset map to locate the next center via Hough voting. The model sequentially segment instances until the stop flag turns on.

2.1 Pick: Tip Points Detection Module

We adapt the U-shaped structure from [3,15] to regress the tip points' heatmap and use a maximum filter to acquire coordinates of tip points. Network details are included in supplementary materials.

2.2 Trace: A Recurrent Network for Filament Tracing

Network Description. After tip points are detected, the tracing module will trace and extract each instance. As shown in Fig. 4, we use patch size 64 as an example to describe our network design. Since we have converted the instance segmentation problem into a sequence learning problem, we use Long Short Term Memory (LSTM) [10] to encode the sequence of patches. We use one LSTM layer with a hidden size of 512 and an input size of 256.

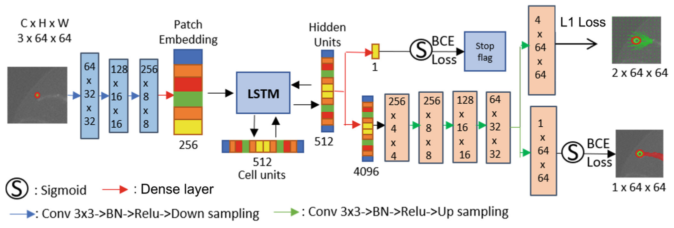

Fig. 4. Network structure for tracing module

The tracing module takes patches as input, and encode each patch into a 256 embedding vector by 3 downsampling blocks followed by a dense layer. The input of LSTM layer is the encoded embedding vector. At each step, the decoder outputs a stop flag, offset maps, and mask. We use a dense layer with a sigmoid activation function to predict the stop flag, which takes the hidden unit as input. As stop flag prediction is a classification problem, it takes an independent branch. The other branch uses a dense layer and decodes the hidden unit to a vector size of 4096, which is reshaped to $256 \times 4 \times 4$. The following layers include 3 conv3x3-bn-relu-upsampling blocks. The offset map prediction is a regression problem, and mask prediction is a binary classification problem. We split the current branch into two branches. The offset map prediction includes a conv3x3-bn-relu-upsampling block and outputs the offset map with a size of $2 \times 64 \times 64$. The offset map includes a horizontal offset channel and a vertical offset channel. The mask branch includes a conv3x3-bn-relu-upsampling-sigmoid block.

Predicting the Next Points. At each step, the decoder regresses offset maps where each pixel predicts a vector pointing to the center of the next patch. We use Hough Voting to decide the exact coordinates of the next center. Each pixel casts a vote to the next point, generating a heatmap of the number of votes for each pixel. The highest response point will be selected as the next center.

Loss Function. We use binary cross entropy (BCE) for the tip prediction. For the tracing module, we use BCE for stop flag and mask prediction and L_1 loss for offset prediction. The final loss for tracing module is

$$L_t = \sum_{t=1}^{t=T} (\lambda_1 L_{bce}(s_t, \hat{s_t}) + \lambda_2 L_{bce}(A, \hat{A}) + \lambda_3 Lreg(M, \hat{M}))$$

T is the number of steps for tracing, and s, A, M stand for stop flag, binary mask, and offset maps. $\lambda_1, \lambda_2, \lambda_3$ are the balance parameters and set as one.

Training and Inference. We use patches as input for training, and the labels are the corresponding offset map, binary mask, and stop flag. The offset map is generated by computing the distance vector between pixels in the current patch

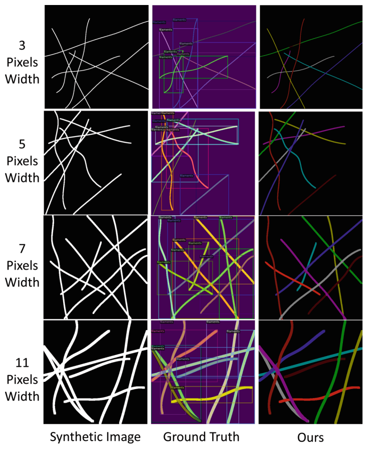

Fig. 5. Qualitative results on synthetic filaments

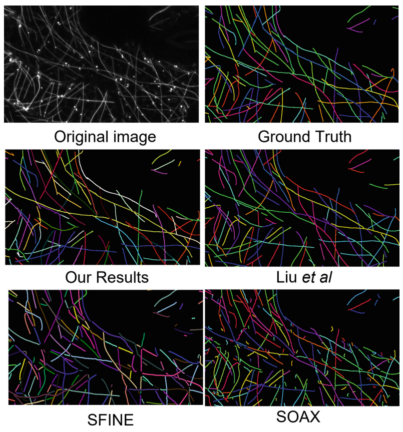

Fig. 6. Qualitative results on our microtubule dataset

to the center of the next patch. The tracing module only run Hough voting to predict the next point' coordinate and crop the next patch during inference.

3 Experiments

Our model is implemented with Pytorch [22] and trained on one RTX 2080 Ti GPU. We convert each instance into patch sequences with a step size of 30 pixels, and patch size of 64×64. We evaluate our approach on four datasets.

Table 1. Quantitative results on synthetic dataset. E, F, G, H are evaluated with our models trained on A, B, C, D respectively, which are highlighted with $\star$. AveInsNum: average number of instance per image. AveLength: average length of filaments in the dataset. AveInsNum and AveLength are reported as $Mean \pm std$.

Datasets					MRCNN [8]			Liu *et al.* [17]			Ours		
	Size	AveInsNum	Width	AveLength	AP	$AP_{0.5}$	$AP0.75$	AP	$AP_{0.5}$	$AP_{0.75}$	AP	$AP_{0.5}$	$AP_{0.75}$
A	256	6.94 ± 2.94	3	221.38 ± 40.89	0.00	0.00	0.00	0.48	0.67	0.56	**0.57**	**0.81**	**0.65**
B	256	6.61 ± 2.38	5	219.92 ± 39.95	0.01	0.02	0.01	0.45	0.71	0.51	**0.51**	**0.77**	**0.57**
C	256	6.86 ± 2.28	7	225.37 ± 38.44	0.07	0.19	0.04	0.39	0.66	**0.43**	**0.40**	**0.67**	0.42
D	256	6.76 ± 2.52	11	222.38 ± 39.54	0.19	0.46	0.14	0.30	0.41	0.31	**0.42**	**0.71**	**0.42**
E	512	9.28 ± 1.62	3	371.94 ± 106.86	0.00	0.00	0.00	0.45	0.69	0.52	**0.66**$\star$	**0.81**$\star$	**0.67**$\star$
F	512	9.16 ± 1.57	5	373.94 ± 104.29	0.01	0.02	0.01	0.47	0.68	0.54	**0.64**$\star$	**0.80**$\star$	**0.70**$\star$
G	512	9.44 ± 1.72	7	381.64 ± 103.56	0.03	0.25	0.13	0.48	0.69	0.55	**0.56**$\star$	**0.76**$\star$	**0.64**$\star$
H	512	9.75 ± 1.94	11	382.75 ± 106.81	0.17	0.44	0.10	0.42	0.60	0.44	**0.52**$\star$	**0.72**$\star$	**0.56**$\star$

Table 2. Quantitative Results

(a) Microtubule dataset

Method	AP	$AP_{0.5}$	$AP_{0.75}$	Time (min/image)
SOAX [26]	0.1860	0.3166	0.202	25
SFINE [29]	0.2222	0.4206	0.1894	5
Liu *et al.* [17]	0.3698	0.5774	0.3689	50
Ours	**0.3882**	**0.6040**	**0.3786**	**5**

(b) P. rubescens dataset

Image	Manual Count [28]	Zeder *et al.* [28]	Liu *et al.* [17]	ours
1	25.8 ± 1.5	27	25	25
2	18.5 ± 0.5	23	22	18
3	27.5 ± 1.5	27	26	27
4	44.0 ± 0	44	44	44
5	36.8 ± 0.4	40	37	36
6	21.8 ± 0.4	21	24	22
7	17.5 ± 1.1	18	18	17
Total	191.8 ± 3.6	200	196	189

(c) C. elegans dataset

Method	AP	$AP_{0.5}$	$AP_{0.75}$	Time (sec/image)
Semi-conv Ops [21]	0.569	0.885	0.661	–
M-RCNN [8]	0.559	0.865	0.641	–
Harmonic Emb. [14]	0.724	0.900	0.723	–
PatchPerPix [9]	**0.775**	**0.939**	**0.891**	13
Ours	**0**.745	0.935	0.828	**4**

3.1 Synthetic Dataset

We first create eight synthetic filament datasets. The statistics of generated datasets are shown in Table 1, and samples are shown in Fig. 5. Each dataset contains 1000 images with random filaments varying in widths. We split each dataset into 700, 200, 100 images for training, validating, and testing. We train our network on A–D (image size 256×256) for 20 epochs and evaluate our model on the their test set. We also directly evaluate the models trained with dataset A–D on dataset E–H (image size 512×512) respectively. We report the average precision (AP) in COCO evaluation criteria [16]. As shown in Table 1, our model trained with smaller size images achieves better results on the unseen datasets (E, F, G, H) with larger images and longer filaments. This is because our model learns how to trace filaments, and longer filaments do not affect the performance of our model. Also, the lower density of filament in E–H making it easier for our model to trace filaments. In addition, thicker filaments create larger overlapping areas and make it harder to separate the filaments. Therefore, the performance decreases from dataset A to D and E to H.

As shown in Table 1, MRCNN [8] achieves zero AP for filaments with a width below five, as segmentation is performed at strides of eight. Our approach outperforms Liu *et al.* [17] in all metrics except $AP_{0.75}$ of dataset C.

3.2 Microtubule Dataset

We annotated 15 microscopic images of microtubules with a size of 1376×1504. The number of filaments per image is 631 ± 167, and the length of filaments is 104 ± 96. We use a modified U-net [18,24] to obtain the binary segmentation. As the average width of microtubules is five, we directly evaluate the microtubule dataset with model trained on synthetic dataset B (see table 1). We compare our approach against SOAX [26], SFINE [29], and deep learning method in [17].

Figure 1 shows qualitative results in full size, and Fig. 6 presents qualitative comparison with detailed area. Our method and Liu *et al.* [17]'s approach can better extract long and crossing filaments. SOAX [26] and SFINE [29]'s break-regroup strategies struggle to regroup segments at intersections and create fragments. Table 2a presents the quantitative comparisons with AP. Our approach has shown a better performance than [17] regarding process time and accuracy.

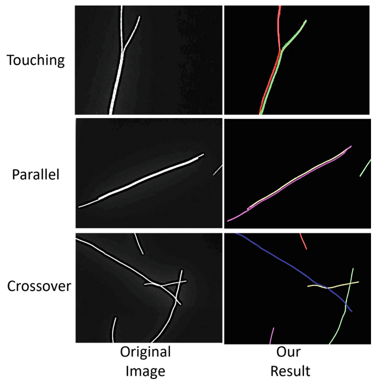

Fig. 7. Qualitative results on P. rubescens dataset.

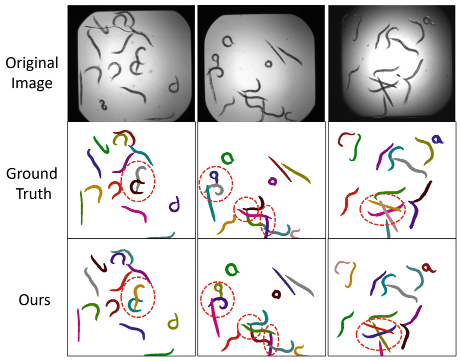

Fig. 8. Qualitative results on C.elegans dataset. Red circle highlight complicated overlapping area (Color figure online)

3.3 P. Rubescens Dataset

P. rubescens is a type of filamentous cyanobacteria, and Zeder *et al.* [28] provide a dataset (Fig. 7) of seven 5000 × 5000 microscopic images of P. rubescens. We apply our model trained with synthetic dataset A (see Table 1) to the binary predictions and follow the evaluation scheme in [28] by comparing the quantity per image. Table 2b shows our result is close to the manual count.

3.4 C. Elegans Dataset

We further investigate our model's performance on the C. elegans roundworm dataset (Fig. 8) from the Broad Bioimage Benchmark Collection [20]. The dataset contains 100 696 × 520 images with an average of 30 roundworms per image. Different from P. rubescens and microtubules, roundworms are much thicker and shorter. We convert each instance in the training set into a sequence of points with a step size of 30 and generate the corresponding 64 × 64 patches for training. The network is trained and evaluated following the set up in [14,21].

Table 2c shows the quantitative comparison between our approach and previous methods [8,9,14,21]. Our method achieves comparable $AP_{0.5}$ to SOTA, and our method runs 9 s faster than SOTA. Red circles in Fig. 8 show our approach handles complex crossover areas.

4 Conclusion

We present a novel method for filament extraction by transforming the instance segmentation problem to a sequence modeling problem. Our method comprises of a sequential encoder-decoder framework to imitate humans extracting filaments and address the challenges brought by filaments' properties, including crossover, spanning and non-rigidity. The experiments show that our method can achieve better or comparable results on filament datasets from different domains. Our method can alleviate the data shortage problem as our models trained on synthetic dataset achieve a better performance on microtubules and P. rubescens dataset. We also train and evaluate our model on C. elegans dataset, achieving comparable results with thicker and shorter filaments. Our method exhibits limitations in tracing "Y"-shaped junctions due to the limited directional information in 2D images. Future work will focus on extending the current method to 3D data.

Acknowledgement. This work was supported by the National Institute of General Medical Sciences grant (R01 GM097587) from the National Institutes of Health. Microscopy equipment was acquired with an NIH-shared instrumentation grant (S10 OD016361) and, access was supported by the NIH-NIGMS (P20 GM103446 and P20 GM139760) and the State of Delaware.

References

1. Chen, H., Qi, X., Yu, L., Heng, P.A.: DCAN: deep contour-aware networks for accurate gland segmentation. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 2487–2496 (2016)
2. Chen, L., Strauch, M., Merhof, D.: Instance segmentation of biomedical images with an object-aware embedding learned with local constraints. In: Shen, D., et al. (eds.) MICCAI 2019. LNCS, vol. 11764, pp. 451–459. Springer, Cham (2019). https://doi.org/10.1007/978-3-030-32239-7_50
3. Chen, Y., Wang, Z., Peng, Y., Zhang, Z., Yu, G., Sun, J.: Cascaded pyramid network for multi-person pose estimation. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 7103–7112 (2018)
4. Dai, J., He, K., Li, Y., Ren, S., Sun, J.: Instance-sensitive fully convolutional networks. In: Leibe, B., Matas, J., Sebe, N., Welling, M. (eds.) ECCV 2016. LNCS, vol. 9910, pp. 534–549. Springer, Cham (2016). https://doi.org/10.1007/978-3-319-46466-4_32
5. De Brabandere, B., Neven, D., Van Gool, L.: Semantic instance segmentation with a discriminative loss function. arXiv preprint arXiv:1708.02551 (2017)
6. Gao, N., et al.: SSAP: single-shot instance segmentation with affinity pyramid. In: Proceedings of the IEEE/CVF International Conference on Computer Vision, pp. 642–651 (2019)
7. Hariharan, B., Arbeláez, P., Girshick, R., Malik, J.: Simultaneous detection and segmentation. In: Fleet, D., Pajdla, T., Schiele, B., Tuytelaars, T. (eds.) ECCV 2014. LNCS, vol. 8695, pp. 297–312. Springer, Cham (2014). https://doi.org/10.1007/978-3-319-10584-0_20
8. He, K., Gkioxari, G., Dollár, P., Girshick, R.: Mask R-CNN. In: 2017 IEEE International Conference on Computer Vision (ICCV), pp. 2980–2988. IEEE (2017)
9. Hirsch, P., Mais, L., Kainmueller, D.: PatchPerPix for instance segmentation. arXiv preprint arXiv:2001.07626 (2020)
10. Hochreiter, S., Schmidhuber, J.: Long short-term memory. Neural Comput. **9**(8), 1735–1780 (1997)

11. Januszewski, M., et al.: High-precision automated reconstruction of neurons with flood-filling networks. Nat. Methods **15**(8), 605–610 (2018)
12. Januszewski, M., Maitin-Shepard, J., Li, P., Kornfeld, J., Denk, W., Jain, V.: Flood-filling networks. arXiv preprint arXiv:1611.00421 (2016)
13. Ke, L., Tai, Y.W., Tang, C.K.: Deep occlusion-aware instance segmentation with overlapping bilayers. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, pp. 4019–4028 (2021)
14. Kulikov, V., Lempitsky, V.: Instance segmentation of biological images using harmonic embeddings. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, pp. 3843–3851 (2020)
15. Lin, T.Y., Dollár, P., Girshick, R., He, K., Hariharan, B., Belongie, S.: Feature pyramid networks for object detection (2017)
16. Lin, T.-Y., et al.: Microsoft COCO: common objects in context. In: Fleet, D., Pajdla, T., Schiele, B., Tuytelaars, T. (eds.) ECCV 2014. LNCS, vol. 8693, pp. 740–755. Springer, Cham (2014). https://doi.org/10.1007/978-3-319-10602-1_48
17. Liu, Y., Kolagunda, A., Treible, W., Nedo, A., Caplan, J., Kambhamettu, C.: Intersection to overpass: instance segmentation on filamentous structures with an orientation-aware neural network and terminus pairing algorithm. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops, pp. 125–133 (2019)
18. Liu, Y., et al.: Densely connected stacked U-network for filament segmentation in microscopy images. In: Leal-Taixé, L., Roth, S. (eds.) ECCV 2018. LNCS, vol. 11134, pp. 403–411. Springer, Cham (2019). https://doi.org/10.1007/978-3-030-11024-6_30
19. Liu, Y., et al.: Affinity derivation and graph merge for instance segmentation. In: Ferrari, V., Hebert, M., Sminchisescu, C., Weiss, Y. (eds.) ECCV 2018. LNCS, vol. 11207, pp. 708–724. Springer, Cham (2018). https://doi.org/10.1007/978-3-030-01219-9_42
20. Ljosa, V., Sokolnicki, K.L., Carpenter, A.E.: Annotated high-throughput microscopy image sets for validation. Nat. Methods **9**(7), 637–637 (2012)
21. Novotny, D., Albanie, S., Larlus, D., Vedaldi, A.: Semi-convolutional operators for instance segmentation. In: Ferrari, V., Hebert, M., Sminchisescu, C., Weiss, Y. (eds.) ECCV 2018. LNCS, vol. 11205, pp. 89–105. Springer, Cham (2018). https://doi.org/10.1007/978-3-030-01246-5_6
22. Paszke, A., et al.: Automatic differentiation in PyTorch (2017)
23. Ren, M., Zemel, R.S.: End-to-end instance segmentation with recurrent attention. In: Proceedings of the 2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Honolulu, HI, USA, pp. 21–26 (2017)
24. Ronneberger, O., Fischer, P., Brox, T.: U-Net: convolutional networks for biomedical image segmentation. In: Navab, N., Hornegger, J., Wells, W.M., Frangi, A.F. (eds.) MICCAI 2015. LNCS, vol. 9351, pp. 234–241. Springer, Cham (2015). https://doi.org/10.1007/978-3-319-24574-4_28
25. Salvador, A., et al.: Recurrent neural networks for semantic instance segmentation. arXiv preprint arXiv:1712.00617 (2017)
26. Xu, T., et al.: SOAX: a software for quantification of 3D biopolymer networks. Sci. Rep. **5**, 9081 (2015)
27. Yi, J., et al.: Multi-scale cell instance segmentation with keypoint graph based bounding boxes. In: Shen, D., et al. (eds.) MICCAI 2019. LNCS, vol. 11764, pp. 369–377. Springer, Cham (2019). https://doi.org/10.1007/978-3-030-32239-7_41

28. Zeder, M., Van den Wyngaert, S., Köster, O., Felder, K.M., Pernthaler, J.: Automated quantification and sizing of unbranched filamentous cyanobacteria by model-based object-oriented image analysis. Appl. Environ. Microbiol. **76**(5), 1615–1622 (2010)
29. Zhang, Z., Nishimura, Y., Kanchanawong, P.: Extracting microtubule networks from superresolution single-molecule localization microscopy data. Mol. Biol. Cell **28**(2), 333–345 (2017)