A Convolutional Neural Network Method for Boundary Optimization Enables Few-Shot Learning for Biomedical Image Segmentation

Erica M. Rutter[1,2(✉)], John H. Lagergren[1], and Kevin B. Flores[1]

[1] Center for Research in Scientific Computation, Department of Mathematics, North Carolina State University, Raleigh, USA
{jhlagerg,kbflores}@ncsu.edu

[2] Department of Applied Mathematics, University of California, Merced, Merced, USA
erutter2@ucmerced.edu

Abstract. Obtaining large amounts of annotated biomedical data to train convolutional neural networks (CNNs) for image segmentation is expensive. We propose a method that requires only a few segmentation examples to accurately train a semi-automated segmentation algorithm. Our algorithm, a convolutional neural network method for boundary optimization (CoMBO), can be used to rapidly outline object boundaries using orders of magnitude less annotation than full segmentation masks, i.e., only a few pixels per image. We found that CoMBO is significantly more accurate than state-of-the-art machine learning methods such as Mask R-CNN. We also show how we can use CoMBO predictions, when CoMBO is trained on just 3 images, to rapidly create large amounts of accurate training data for Mask R-CNN. Our few-shot method is demonstrated on ISBI cell tracking challenge datasets.

Keywords: Biomedical image segmentation · Few shot learning · Convolutional neural network

1 Introduction

Convolutional neural networks (CNNs) have recently been used to automate the segmentation of biomedical images [3], enabling an increase in the speed and accuracy of diagnosis, histology, and cell image analysis. However, creating segmentation training data for CNNs is a time intensive process requiring expert human annotation by clinicians or scientists. Thus, there is a need for methods to reduce the annotation burden by (1) drastically decreasing the amount of data

Electronic supplementary material The online version of this chapter (https://doi.org/10.1007/978-3-030-33391-1_22) contains supplementary material, which is available to authorized users.

© Springer Nature Switzerland AG 2019
Q. Wang et al. (Eds.): DART 2019/MIL3ID 2019, LNCS 11795, pp. 190–198, 2019.
https://doi.org/10.1007/978-3-030-33391-1_22

required to accurately train CNNs for segmentation, or (2) semi-automating the segmentation process while requiring minimal expert annotation. Recently, a novel method was proposed for using CNNs to improve segmentation accuracy by optimizing the task of tracing the boundary of objects in biomedical images [9]. This work found that using CNNs for optimizing boundary tracing accuracy better ensured contiguity of segmented regions and resulted in hyper-accurate cell segmentations. A unique aspect of the boundary optimization method is that the number of training examples obtained from a training image and its mask is equal to the number of pixels on the boundary of any object in the image. This is because the input to the CNN for boundary optimization is a small patch of the training image centered around any pixel on the boundary of an object, and the output is the prediction of the relative pixel displacements of the next m pixels in the trace (see Fig. 1, right panel). Thereby, a single segmentation training example can potentially yield hundreds or thousands (depending on the image size) of training examples for the task of boundary optimization. In this work, we investigated whether this property of boundary optimization could be leveraged to create accurate CNN-based segmentation methods for tasks (1) and (2) above with using only a few training images.

Contributions:

- We show that our **Co**nvolutional neural network **m**ethod for **b**oundary **o**ptimization (CoMBO) can be used to accurately segment biomedical images using just 3 training examples. To make predictions, this method requires an extremely minimal amount of human annotation, i.e., a single pixel per object.
- Provide a comparison of CoMBO with Mask R-CNN [3] and U-net [8].
- We show that CoMBO predictions can be used to rapidly create accurate training data for Mask R-CNN. The Mask R-CNN model trained on CoMBO predictions is just as accurate as a Mask R-CNN model trained on human annotations.

Related Work (Few-Shot Learning): Our work is related to the task of training a method for image segmentation with only a few training examples, i.e., few-shot learning. Unlike the task of one or few-shot image classification, the concept of one or few-shot image segmentation is relatively new [2,5,6,10]. Importantly, many previous methods for few-shot segmentation have been developed with a large margin of error and for multiple classes, since the focus has not been on biomedical imaging. Shaban et al. [10] created the first one-shot semantic segmentation network. Many few-shot segmentation techniques rely on using pre-trained networks [2,6], which may not be as applicable to medical imaging datasets. Other few-shot techniques consider multi-class segmentation, thus leveraging the existence of multiple images (one per class). Michaelis et al. [5] created a one-shot segmentation algorithm in clutter, but their method is best suited towards an instance where there is only one target in the image to segment, while there may be many targets to segment in a biomedical image.

2 Methods

In contrast to previous few-shot segmentation learning approaches, our method does not use transfer learning or pre-training. By formulating the learning task as a boundary optimization task, we naturally create many image-label pairs upon which to train a CNN for our algorithm. Our method modifies a previously developed CNN-based algorithm for object tracing described in [9].

The input to the CNN is a 64×64 patch of an image with a previously 'traced' boundary overlaid (Fig. 1, left panel). The CNN itself consists of three repeating blocks, each of which has 3 3×3 convolutional layers followed by a max-pooling layer. The final layer is an 8×8 convolutional layer. The number of filters for each repeating block is 32, 64, and 128, while the final layer has 60 filters. The output of the CNN is the next predicted 30 pixel horizontal and vertical displacements of the boundary relative to the center of the image (Fig. 1, middle). These horizontal and vertical displacements are then overlaid on the image as the cell boundary (Fig. 1, right panel cyan). A key modification we make to the algorithm in [9] is that we use the predicted displacements to move the trace multiple steps instead of one step at a time. This has resulted in higher accuracies, since the algorithm can 'skip' over problematic areas, while also speeding up forward passes by an order of magnitude. We do this by using a Bresenham line to connect the predicted pixel locations, thus ensuring a smooth outline of a cell. The number of steps to trace at each iteration is treated as a hyper-parameter selected using the validation data.

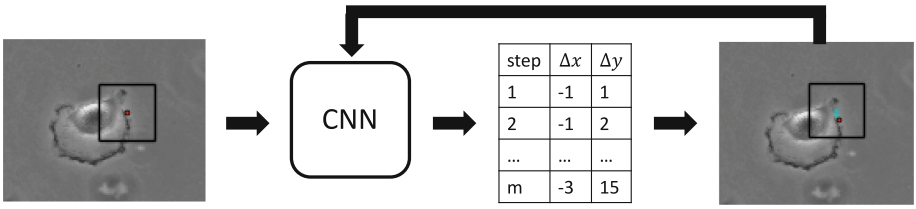

Fig. 1. Schematic of the tracing algorithm. Weak annotation shown as a red dot. The CNN takes as input the black patch, returns the next m predicted pixel locations, which are overlaid on the image in cyan. (Color figure online)

To trace an object in an image, we first choose an initial trace location (Fig. 1, left panel, red dot). In previous work [9] this initialization point was determined via output from other convolutional neural networks (U-Net). Due to the inaccuracies of such estimations in the few-shot setting, we instead utilize weak annotations provided by the user, namely a single pixel on the boundary of each object. Although annotating full segmentations for each object is laborious, clicking on an initial starting location for each object in an image is relatively quick and simple: for a dataset with approximately 25–30 objects per image, we

were able to provide starting locations for approximately 12 images in 8 min. Once the initialization is begun, we pass the 64×64 image patch around the starting location through the CNN.

The forward pass on an image consists of (i) choosing an initial location (weak annotation by the user), (ii) iteratively using the CNN to trace the outline of the object, and (iii) stopping the iteration when the trace is greater than a certain length and the final pixel predictions are within a small distance to the initial pixel location.

One natural question arising from this algorithm is what happens for non-ideal tracing patterns? There are two such possible cases: either the CNN predicts a trace in the direction it just came from, or the trace deviates far from the true object boundary. We generate training data to always train counter-clockwise to ensure that traces do not go back in the direction they came from. We use a large patch size and, more importantly, predict the next thirty pixel displacements. By predicting multiple steps ahead, we can 'skip' areas in which the tracing algorithm might go awry. By including an adequate level of image context via a large patch size, the true boundary location is usually included in the image patch being passed to the CNN and the prediction can direct the trace back toward the boundary. We have not experienced a trace going off-course, but we include Supplementary Figure S1 which shows that the CNN predicts a trajectory that recovers from an initial pixel location off the boundary.

3 Experiments

Data: We evaluated our methodology on two grayscale light microscopy image data sets from the ISBI cell tracking benchmarks [4,11]: (1) GFP-GOWT1 mouse stem cells (Fluo-N2DH-GOWT1), and (2) Glioblastoma-astrocytoma U373 cells (PhC-C2DH-U373). From each data set, we used k images for training, 1 image for validation, and tested on the remaining images. Images from these datasets were prepared by zero-padding with 32 pixels (to ensure 64×64 patches for the CNN could be generated at the edge of the image). We found that this simplifies the algorithm in [9], which used symmetric padding, by helping to keep the trace away from the padded region when it reaches the edge of an image. Only images that had corresponding masks were used for training, validation, and testing (8 images for the GFP-GOWT1 dataset and 34 images for the U373 dataset). We performed 5-fold cross-validation for all experiments. In order to reduce stochasticity associated with initial pixel locations for the traces, the results from CoMBO are reported as the mean of 10 random initial locations for each object boundary. We perform the following series of augmentations at random to produce 48 augmented images per training image: up-down flips, left-right flips, rotations between $-45°$ and $75°$, shears between $-10°$ and $30°$, Gaussian blurring, and additive Gaussian noise.

Evaluation Metrics: Several metrics are used to assess the accuracy of our segmentations, since recent work showed that altering evaluation metrics can

vastly change how algorithms are ranked [7]. To evaluate the accuracy of both the semantic segmentation and predicted cell morphology we calculated the Jaccard score, Dice Similarity Coefficient, Hausdorff distance, and mean surface distance (MSD).

Baseline: We compare the performance of our approach to Mask R-CNN, a well known method for instance segmentation [3]. Although Mask R-CNN is not formulated specifically for few-shot learning, we use it as a baseline comparison because there are not many other published few-shot segmentation algorithms. Moreover, Mask R-CNN is one of the best-performing benchmark segmentation algorithms. To make Mask R-CNN more adept at the few-shot segmentation task, we start training Mask R-CNN from weights that were pre-trained on imagenet [1]. We fine-tuned Mask R-CNN for each dataset and k-shot experiment for 400 epochs.

4 Results

We consider few-shot learning on 1, 3, and 5 training images. For each of these sets, an additional image is used for validation. We compare results with a pre-trained Mask R-CNN fine-tuned on the same number of images. Figure 2 displays the median ($\pm$ standard deviation) of the Jaccard scores and mean surface distances (MSD) for the GFP-GOWT1 dataset and the U373 dataset.

For the GFP-GOWT1 dataset, we found that CoMBO performs significantly better for all k-shot experiments in both Jaccard score and MSD. Furthermore, we observe lower standard deviations in the CoMBO model, implying that CoMBO was much less sensitive to the choice of training data. CoMBO was especially better at predicting accurate cell morphology in the few-shot setting, as reflected in the MSD and Hausdorff metrics. The evaluation metrics (Jaccard, Dice, MSD, and Hausdorff Distance) are reported for the GFP-GOWT1 dataset in Supplementary Table S1.

For the U373 data, we found that CoMBO was significantly better than Mask R-CNN using just 3 images for training (Fig. 2, right). Moreover, the CoMBO algorithm appears to reach convergence in Jaccard scores (and MSD) with only 5 images, i.e., training on more images did not appear to improve segmentation accuracy.

Since CoMBO is a semi-automated method, we investigated whether it could be used to rapidly generate data that was accurate enough to train Mask R-CNN. If CoMBO predictions are accurate enough for this purpose, then it would show that it could be used to effectively take the human out of the loop, eliminating the need for any human annotation. To test this, we used the 3-shot trained CoMBO that had the median Jaccard score for the U373 dataset to predict masks for the remaining images (approximately 30 images). We note that this would require minimal human annotation for each image, i.e., one pixel on the boundary of each object to initialize the predicted trace. We then generated masks from the CoMBO traces and used these data to train Mask R-CNN. We found that Mask R-CNN trained on data from CoMBO traces was able to achieve similar accuracy

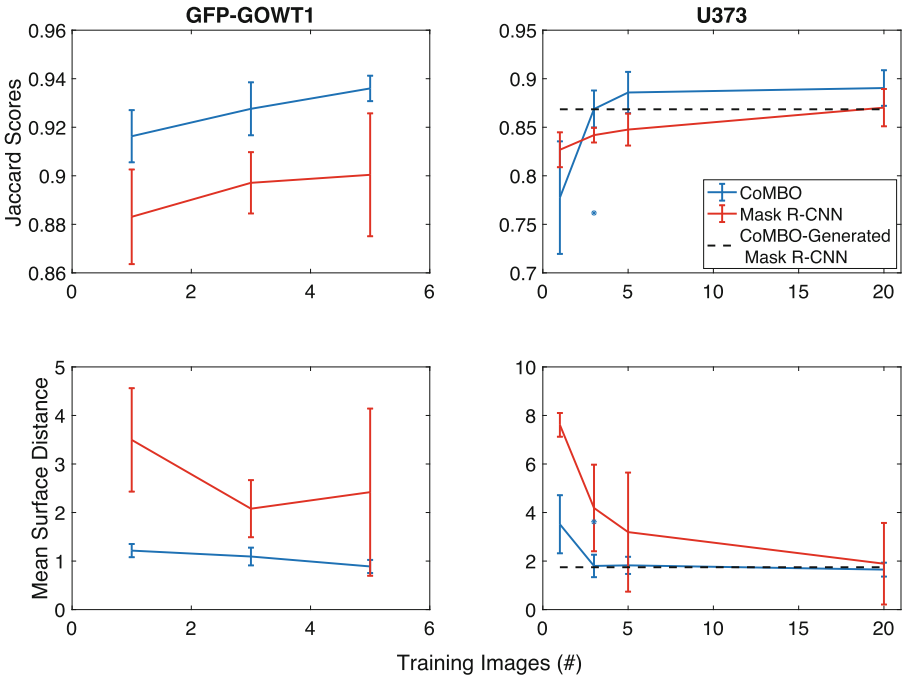

Fig. 2. Median Jaccard scores (top) and mean surface distances (bottom) for the GFP-GOWT1 dataset (left) and the U373 dataset (right). Blue=CoMBO, Red = Mask R-CNN, and the dashed black line represents Mask R-CNN retrained on images traced by CoMBO. Error bars denote standard deviation, outliers (*) were removed from standard deviation calculations. (Color figure online)

as using ground-truth masks (Fig. 2, right). These results suggest that CoMBO can be used to quickly and accurately annotate large datasets for fully automated machine learning methods. Figure 3 displays an example segmentation from the U373 testing set for 3-shot Mask R-CNN (left), 3-shot CoMBO (middle), and the Mask R-CNN trained on CoMBO-generated trainined data (right). This example shows how using the predicted masks from the 3-shot CoMBO to train Mask R-CNN is able to fix the false positives and improve both the Jaccard score and MSD.

Few-shot segmentation results for the U373 dataset are shown in Table 1 for all four accuracy metrics we considered. At all k-shot levels, CoMBO performed significantly better in the Hausdorff distance and MSD metrics. Mask R-CNN had a higher Jaccard and Dice scores for 1-shot segmentation. However, CoMBO had significantly higher Jaccard and Dice scores when trained on 3 and 5 images, and on the full dataset. Similar results for the GFP-GOWT1 dataset are reported in Supplementary Table S1.

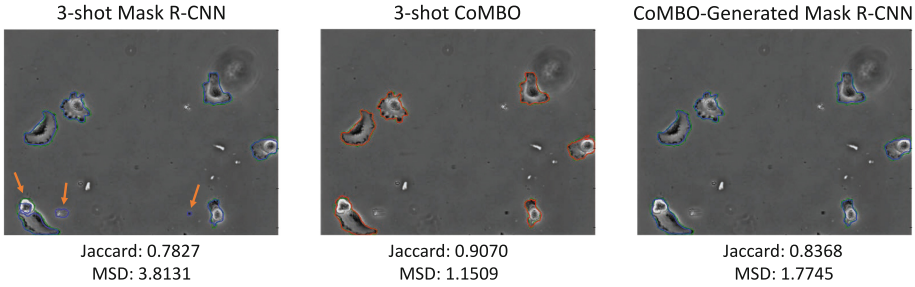

Fig. 3. An example image from the U373 test set for the 3-shot Mask R-CNN (left), 3-shot CoMBO (middle) and Mask R-CNN trained on CoMBO-generated images (right). Green=Ground Truth, Blue=Mask R-CNN, Red=CoMBO. Orange arrows highlight areas in which CoMBO is more accurate. The 3-shot Mask R-CNN had difficulty accurately segmenting cells, both by having false positive cells and inaccurately segmenting the leftmost cell. (Color figure online)

Table 1. Performance of algorithms on the testing set averaged over five train/val/test data splits for CoMBO and Mask R-CNN for the U373 dataset. Bold denotes the best score within each k-shot experiment.

k-shot	Method	Jaccard score mean (std)	Dice coefficient mean (std)	Mean surface distance mean (std)	Hausdorff distance mean (std)
1-shot	Mask R-CNN	**0.8192** (0.01796)	**0.8986** (.01780)	7.3651 (0.4886)	137.3790 (11.9108)
	CoMBO	0.7866 (.05795)	0.8754 (.03905)	**3.4096** (1.1970)	**36.0004** (12.0755)
3-shot	Mask R-CNN	0.8434 (0.007609)	0.9135 (0.01078)	5.2904 (1.7850)	97.1921 (48.9499)
	CoMBO	**0.8679** (0.01906)	**0.9276** (0.01172)	**1.9130** (0.4637)	**25.0281** (5.7986)
5-shot	Mask R-CNN	0.8451 (0.01645)	0.9140 (0.003871)	4.7925 (2.4530)	81.9223 (60.8722)
	CoMBO	**0.8745** (0.02126)	**0.9313** (0.01369)	**1.9041** (0.3538)	**24.0791** (5.7986)
Full	Mask R-CNN	0.86278 (0.01924)	0.9246 (0.01430)	2.6368 (1.6806)	30.1077 (18.8510)
	CoMBO	**0.8914** (0.01831)	**0.9416** (0.01057)	**1.5586** (0.2866)	**18.5223** (2.0231)
	Retrained Mask R-CNN	0.8685	0.9294	1.7458	14.0817

5 Discussion

We found that our CoMBO algorithm for image segmentation is able to achieve accurate segmentations with 3 or fewer training images. We speculate that CoMBO is able to achieve high accuracy with a few training images because it transforms a small training data set, i.e., a few image/segmentation pairs, into thousands of training examples for a boundary optimization CNN task. It does so at the cost of requiring minimal user input, i.e., clicking a single pixel on the boundary of each object in an image. However, we also found that the predicted segmentations from CoMBO were accurate enough to create training data for Mask R-CNN [3], a fully automated segmentation method. The accuracy of Mask R-CNN trained on CoMBO data matched the use of ground-truth data.

Future work will include extending CoMBO to multi-class segmentation and also augmenting this method to handle instance segmentation, perhaps by using Mask R-CNN predictions as an additional channel for each patch input to the CNN. Using other few-shot algorithms to determine a starting location along the cell boundary would also enable a fully-automated few-shot segmentation learning approach.

References

1. Abdulla, W.: Mask R-CNN for object detection and instance segmentation on keras and tensorflow (2017). https://github.com/matterport/Mask_RCNN
2. Caelles, S., Maninis, K.K., Pont-Tuset, J., Leal-Taixé, L., Cremers, D., Van Gool, L.: One-shot video object segmentation. In: CVPR 2017. IEEE (2017)
3. He, K., Gkioxari, G., Dollár, P., Girshick, R.: Mask R-CNN. In: Proceedings of the IEEE International Conference on Computer Vision, pp. 2961–2969 (2017)
4. Maška, M., et al.: A benchmark for comparison of cell tracking algorithms. Bioinformatics **30**(11), 1609–1617 (2014)
5. Michaelis, C., Bethge, M., Ecker, A.: One-shot segmentation in clutter. In: International Conference on Machine Learning, pp. 3546–3555 (2018)
6. Milan, A., et al.: Semantic segmentation from limited training data. In: 2018 IEEE International Conference on Robotics and Automation (ICRA), pp. 1908–1915. IEEE (2018)
7. Reinke, A.: How to exploit weaknesses in biomedical challenge design and organization. In: Frangi, A.F., Schnabel, J.A., Davatzikos, C., Alberola-López, C., Fichtinger, G. (eds.) MICCAI 2018. LNCS, vol. 11073, pp. 388–395. Springer, Cham (2018). https://doi.org/10.1007/978-3-030-00937-3_45
8. 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

9. Rutter, E.M., Lagergren, J.H., Flores, K.B.: Automated object tracing for biomedical image segmentation using a deep convolutional neural network. In: Frangi, A.F., Schnabel, J.A., Davatzikos, C., Alberola-López, C., Fichtinger, G. (eds.) MICCAI 2018. LNCS, vol. 11073, pp. 686–694. Springer, Cham (2018). https://doi.org/10.1007/978-3-030-00937-3_78
10. Shaban, A., Bansal, S., Liu, Z., Essa, I., Boots, B.: One-shot learning for semantic segmentation. arXiv preprint arXiv:1709.03410 (2017)
11. Ulman, V., et al.: An objective comparison of cell-tracking algorithms. Nat. Methods **14**, 1141 (2017). https://doi.org/10.1038/nmeth.4473