Iterative Interaction Training for Segmentation Editing Networks

Gustav Bredell, Christine Tanner, Ender Konukoglu

Computer Vision Laboratory, ETH Zurich, Zurich, Switzerland
gbredell@student.ethz.ch

Abstract. Automatic segmentation has great potential to facilitate morphological measurements while simultaneously increasing efficiency. Nevertheless often users want to edit the segmentation to their own needs and will need different tools for this. There has been methods developed to edit segmentations of automatic methods based on the user input, primarily for binary segmentations. Here however, we present an unique training strategy for convolutional neural networks (CNNs) trained on top of an automatic method to enable interactive segmentation editing that is not limited to binary segmentation. By utilizing a robot-user during training, we closely mimic realistic use cases to achieve optimal editing performance. In addition, we show that an increase of the iterative interactions during the training process up to ten improves the segmentation editing performance substantially. Furthermore, we compare our segmentation editing CNN (interCNN) to state-of-the-art interactive segmentation algorithms and show a superior or on par performance.

1 Introduction

Segmentation is one of the main medical image analysis tasks that when automated substantially facilitates morphological measurements and increase efficiency in treatment planning [11,17,18]. With the introduction of machine learning and especially convolutional neural networks (CNNs) the performance of automatic segmentation approaches improved greatly [7]. Recent studies showed that CNN-based approaches were able to achieve inter- and intra-expert performance in certain segmentation tasks, for example prostate segmentation in Magnetic Resonance Images (MRIs) as shown in [8,4]. Although these approaches achieve impressive performance on average, when considering an individual image, there are often parts of the segmentation users would like to change and improve to fit their needs. The need for edits and improvement is even larger when the test image differs slightly from the training dataset, for example due to scanner differences, and more errors are expected.

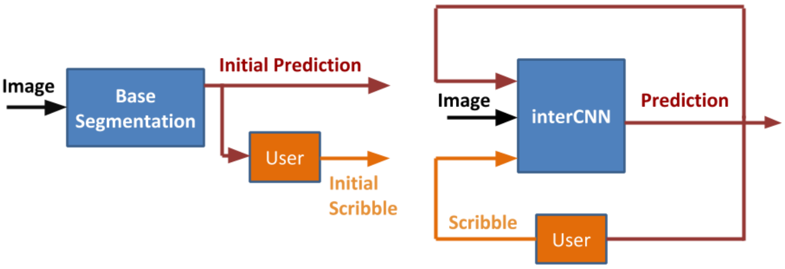
To address the need for editing, interactive segmentation algorithms have been proposed such as GrabCut, GeoS or Random Walker [14,3,5] that allow operators to modify segmentations. Even though accurate results have been shown with these methods, the interaction can be time consuming as large number of interactions might be necessary. In particular, updates aiming to correct segmentation in one region can lead to inaccuracies in another region, consequently requiring further interactions.

In recent years, studies such as [1,9,19] proposed CNNs for interactive segmentations and showed better results compared to traditional methods. These initial works focused on segmenting objects in medical images from scratch using simple user interactions, and mostly in the form of binary segmentations. More recently, authors in [20] proposed a CNN-based method for editing segmentations predicted by an automatic algorithm, one of the most important steps in translating automatic segmentations in practice, and showed the benefits for binary segmentations. In the same work, authors assumed multiple scribbles to be made at a single time and the editing network was trained to take into account all the edits, initial prediction and the image to generate an updated segmentation. This training strategy may not be ideal since it does not take into account the fact that a user may be interacting with the tool over several iterations, each time providing scribbles based on the result of the last update.

In this work, we present a different strategy for training a CNN that interactively edits segmentations. As in [20], we assume the editing CNN is an auxiliary tool that supports a base segmentation algorithm and is optimized to take into account user edits and improve segmentation accuracy. Different than [20], we investigate training in an iterative interaction fashion on simulated user inputs and we also focus on multi-label segmentation problems as well as binary ones. We assess the potential of the proposed training strategy on the prostate data of the NCI-ISBI 2013 challenge and show the value of iterative interaction training. Moreover, we empirically compare networks for editing segmentations with a state-of-the-art fully interactive segmentation algorithm that segments the image from scratch using user-made scribbles.

2 Methods

Interactive segmentation editing networks, which we refer to as *interCNN*, are trained on top of a base segmentation algorithm, specifically to interpret user inputs and make appropriate adjustments to the predictions of the base algorithm. During test time, an interCNN sees the image, initial predictions of the base algorithm and user edits in the form of scribbles, and combines all to create a new segmentation, see Figure 1. In case the new segmentation needs more edits, an interCNN can be applied in an iterative fashion until the segmentation is satisfactory by accepting additional scribbles and taking the image and its own predictions as inputs. Training of an interCNN can be done in two ways. First, as done in [20], given the segmentation of the base network, a set of scribbles are provided and the interCNN is trained to update the segmentation the best way possible by using all the scribbles, image and the base network's segmentation. Ideally, human users should provide the scribbles during the training, however, this is clearly infeasible and a robot user is often utilized to provide the scribbles and has been shown to perform well.



(a) Base segmentation and scribbles (b) Interactive improvement loop

Fig. 1. Illustration of interactive segmentation editing networks. (a) generation of initial prediction with base segmentation and first user input, (b) interactive improvement loop with proposed interCNN. Here, we use a CNN for the base segmentation algorithm for demonstration but other methods can be used. interCNN can be applied iteratively until the segmentation is satisfactory. During training, to make it feasible, the user is replaced by a robot user that places scribbles based on the discrepancy between ground truth and predicted segmentations for the training images.

Algorithm 1: training interCNN for B batch and K interaction iterations

```
Input   : images $\mathbf{I}^b$, ground-truth labels $\mathbf{L}^b$
Output: interCNN weights $\mathbf{W}_K$, predictions $\mathbf{P}^b_K$
for $b \in \{1, 2, ..., B\}$ do
    $\mathbf{P}^b_0 \leftarrow$ autoCNN($\mathbf{I}^b$)
    $\mathbf{S}^b_0 \leftarrow$ random-user($\mathbf{P}^b_0, \mathbf{L}^b$)
    for $k \in \{1, ..., K\}$ do
        $\mathbf{P}^b_k \leftarrow$ interCNN($\mathbf{I}^b, \mathbf{P}^b_{k-1}, \mathbf{S}^b_{k-1}$)
        $\mathbf{S}^b_k \leftarrow$ random-user($\mathbf{P}^b_k, \mathbf{L}^b$)
        backpropagate cross-entropy($\mathbf{P}^b_k$,$\mathbf{L}^b$) loss to update $\mathbf{W}_k$
    end
end
```

Iterative interaction training: The alternative training strategy, which we propose here, is to replicate the testing procedure and integrate iterative interactions to optimize the network. An overview of this strategy is presented as a pseudocode in Algorithm 1. Images in the training set ($\mathbf{I}^b$) are fed batch-wise into the base algorithm to create initial predictions ($\mathbf{P}^b_0$). Scribbles ($\mathbf{S}^b_k$) are produced by a robot user based on the discrepancy between $\mathbf{P}^b_k$ and the ground truth segmentations ($\mathbf{L}^b$). $\mathbf{S}^b_k$ has an image format in which the user-selected wrongly classified pixels are marked according to their correct class and all other pixels are set to max($\mathbf{L}^b$)+1. The initial scribbles $\mathbf{S}^b_0$, along with $\mathbf{P}^b_0$ and $\mathbf{I}^b$ are subsequently fed into interCNN to get an updated prediction ($\mathbf{P}^b_1$). Based on $\mathbf{P}^b_1$ new scribbles are produced by the robot user ($\mathbf{S}^b_1$) and are fed into the interCNN in the next iteration (k+1) together with $\mathbf{I}^b$ and $\mathbf{P}^b_1$. During interaction iteration k the weights of interCNN ($\mathbf{W}_k$) are updated with backpropagation based on the cross-entropy loss between $\mathbf{P}_k^b$ and $\mathbf{L}^b$. This is repeated for a fixed number of K interaction iterations before moving on to the next batch of images $\mathbf{I}^{b+1}$.

Base segmentation method: Ideally, the base segmentation algorithm is arbitrary. An interCNN can be used with any algorithm. In this work, we used a segmentation CNN as the base algorithm due to their superior performance, similar to [20], and refer to it as autoCNN.

Network architecture: We used a U-Net architecture [13] for both the autoCNN and interCNN. It has been shown that this architecture produces automatic segmentation results on medical images that is comparable to more complex architectures [16,21]. Our implementation consists of 4 down- and 4 up-convolutional layers. Each down-convolutional layer is also connected to its respective up-convolutional layer through skip-connections. The final prediction of the U-Net is obtained by a softmax layer. The input consisted of 320×320 pixel patches. Most U-Net networks take the image as the only input. interCNN, however, takes three inputs: image, prediction and scribble mask.

For the base segmentation model autoCNN, we also used the same U-Net architecture but with only the image as the input. For both interCNN and autoCNN, we used drop-out and batch normalization during training [15,6].

We note that more complex networks can also be used both for autoCNN and interCNN. Here, we use a relatively simple architecture since our focus is on the training strategy rather than the architecture.

Robot user: The robot user we utilized for training the network is based on the model introduced by Nickisch et al. [10]. Here a random-user model is used. At each iteration a scribble is produced for every class in the image by comparing the prediction to the ground truth. First, all incorrectly classified pixel are identified. Subsequently, a pixel from the incorrectly classified pixels is chosen randomly for each class separately. In a next step, a region of 9×9 pixels is placed around each randomly chosen pixel and all the pixels in this region belonging to the class the scribble is currently made for are saved as the scribble for the respective class. This process is repeated for all classes in each iteration. The scribbles from all classes are then added together to obtain the final scribble mask for the respective iteration. The randomness in choosing the scribbles prevents the interCNN from over-fitting to a specific strategy that the user may not reproduce during test time, for instance always choosing the center of gravity of the difference set.

Implementation details: We used PyTorch [12] and Python to implement our U-net and robot user, respectively. The training took place on the in-house GPU cluster mainly consisting of GeForce GTX TITAN X with 12GB memory. The Adam optimization algorithm was used for training. The batch size was fixed to 4 images, learning rate to 0.0001 and the maximal number of iterations was 140'000. The images were normalized by taking the median of the training images and dividing all images by this value. To prevent over-fitting, data augmentation was used during training. For each batch, cropping, rotation or flipping was applied to all the images within the batch with a probability of 0.5.

3 Experiments and Results

Materials: We used the prostate dataset of the NCI-ISBI 2013 challenge [2]. The dataset consists of T2-weighted MRIs of the prostate acquired with a 3.0 T scanner. In total the dataset includes 60 patient volumes, each containing 15-20 slices. Of the 60 patients only 29 had multi-class ground truth segmentations, where the central gland and the peripheral zone were labeled. We focused our experiments on these 29 subjects to present results in multi-class segmentation.

We randomly divided the patients into 4 groups G1-G4. G1 contained 15 patients and was used as training data for the base segmentation algorithm, autoCNN. G2 consisted of 8 patients and was used as validation data for autoCNN. For training interCNN, both G1 and G2 were used. Training interCNN with G2 is crucial, since often the base method already performs very well on its training data, so interCNN would not encounter large incorrect classifications if only trained with G1. One patient, G3, was used as validation data to select the best performing interCNN. G4 constituted the test data and consisted of 5 patients.

For the benchmarking against other approaches, which were all focused on binary segmentation, we kept the same groups, but transformed the multi-class labels to binary by fusing the central gland and peripheral zone.

Evaluation: We employed the random robot-user for assessing test performance for the sake of efficiency. This neglects potential user errors but it does not simulate an ideal user nor favours a particular behaviour due to the randomness.

The segmentation performance was quantified using the Dice score (DSC): $DSC = \frac{2|S_g \cap S_p|}{|S_g|+|S_p|}$ where S_g is the ground truth and S_p is the predicted segmentation, and $| \cdot |$ denotes the number of pixels. We simulated that the user was interactively editing the proposed segmentations of each test image up to 20 times. We calculated the Dice score after every simulated user interaction to see how the segmentation results are influenced by the number of user inputs.

Computation speed: The interCNN produced an updated prediction per interactive iteration with a mean time of $3.9\,\mathrm{ms} \pm 0.2\,\mathrm{ms}$, thus enabling real-time use. GrabCut needs 1.2s per update (openCV implementation). Hence a substantial increase in update speed is obtained with interCNN over GrabCut.

Iteration training parameter: As shown in Algorithm 1, the proposed training strategy is to train interCNN for a fixed K number of iterations per batch. Meaning the predictions of every batch of images are iteratively updated together with their respective scribbles and fed back into interCNN for K number of consecutive iterations before moving on to the next batch. To inspect the influence of number of iterations during training, we varied K from 1 to 15.

The results for the two prostate structures are shown in Fig. 2. It can be seen that the Dice score improvement is substantially lower if iteration parameter K is set to 1 or 5, compared to a K of 10 and higher. Even though there is an initial improvement of the Dice score with a low K, the improvement slows down at later interaction iterations. One possible explanation for this observation

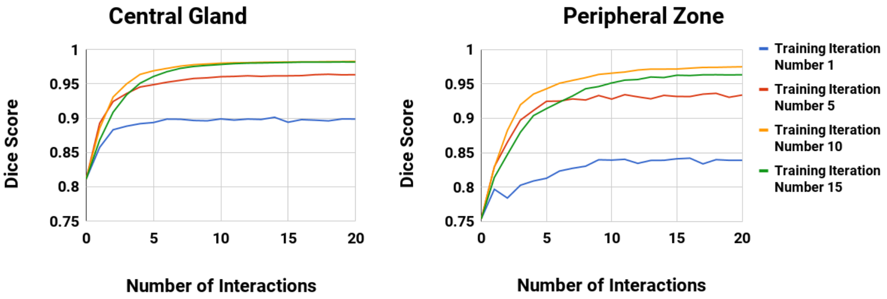


Fig. 2. Segmentation performance for interCNN trained with 1 to 15 iterations (K) for (left) central gland and (right) peripheral zone.

could be that the interCNN is mostly confronted with large incorrectly classified areas during training for low iteration parameters and learns to make large segmentation adjustments which is not required or beneficial at later stages.

Comparison to interactive segmentation from scratch: To evaluate the value of segmentation editing compared to state-of-the-art interactive segmentation from scratch, we looked at two recently proposed approaches.

UI-Net: The method is based on a CNN taking scribbles and the image as input to update its segmentation [1]. No automatic segmentation takes place, but rather initial scribbles are provided by the user. In contrast to [1], the initial scribbles were chosen randomly and not by erosion and dilation. As CNN we used the same U-Net architecture as for interCNN.

BIFSeg: This method is based on fine-tuning the last-layer of a CNN to update segmentations based on user inputs [19]. The algorithm starts by asking the user to draw a bounding-box around the object of interest. An initial segmentation is then computed and the scribbles of the user in the following iterations are used to fine-tune the last layer of the CNN that predicted the initial segmentation. We used their open-source code to benchmark against, which is claimed to also work on objects not seen during training.

In Fig. 3 the results of the comparison to interCNN with 10 training iterations can be seen. As both of the methods we compare to require user interaction, their Dice scores only start at iteration one. For BIFSeg this initial input is the bounding-box annotation. We investigated how the Dice score changed for all these methods over the course of 20 user interactions. It can be observed that interCNN, which edits existing segmentations, required substantially fewer user interactions than BIFSeg to reach a high Dice score (5 vs. 20). The performance of UI-Net, on the other hand, was very similar to the proposed method for this dataset, but it also used the full training dataset as it was trained from scratch.

The iterative improvement of the base segmentation by interCNN is illustrated on a representative test example in Fig. 4.

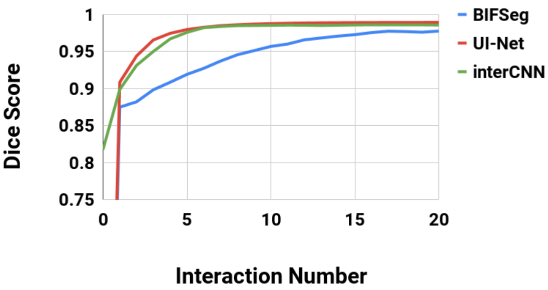


Fig. 3. Segmentation performance of proposed method (interCNN) in comparison to state-of-the-art methods for increasing number of user interactions (1-20).

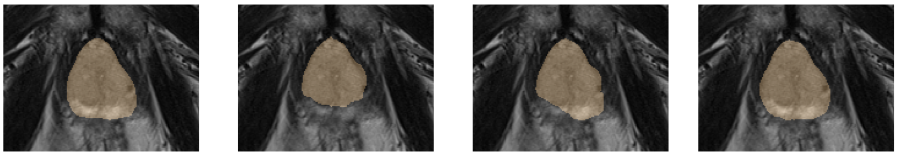


Fig. 4. Visual examples: segmentation overlays for (left→right) ground truth, autoCNN (DSC:0.84), and interCNN after interaction 1 (DSC:0.93) and 5 (DSC:0.98).

4 Conclusions

We proposed an iterative interaction training strategy for efficient segmentation editing with networks. Compared to non-iterative training, the proposed strategy yielded higher segmentation accuracy. The difference was the highest when the iteration parameter for training was at ten and higher. The proposed strategy allows the CNN to learn to correct small and large errors. Finally, we compared our method to alternatives that perform interactive segmentation from scratch. We observed that interCNN when trained with the proposed strategy yielded results on par with the state-of-the-art methods. The advantage of segmentation editing networks, such as interCNN, compared to interaction segmentation from scratch is that they do not need user interaction to initialize segmentation.

Acknowledgments: We thank the Swiss Data Science Center (project C17-04 deepMICROIA) for funding and acknowledge NVIDIA for GPU support.

References

1. Amrehn, M., Gaube, S., Unberath, M., Schebesch, F., Horz, T., Strumia, M., Steidl, S., Kowarschik, M., Maier, A.: UI-Net: Interactive artificial neural networks for iterative image segmentation based on a user model. arXiv:1709.03450 (2017)
2. Bloch, N., Madabhushi, A., Huisman, H., et al.: NCI-ISBI 2013 challenge: automated segmentation of prostate structures. The Cancer Imaging Archive (2015)
3. Criminisi, A., Sharp, T., Blake, A.: GeoS: Geodesic image segmentation. In: European Conference on Computer Vision. pp. 99–112. Springer (2008)
4. van Ginneken, B., Kerkstra, S., Litjens, G., Toth, R.: PROMISE12 challenge results. `https://promise12.grand-challenge.org/evaluation/results/` (2018)

5. Grady, L., Schiwietz, T., Aharon, S., Westermann, R.: Random walks for interactive organ segmentation in two and three dimensions: Implementation and validation. In: Medical Image Computing and Computer-Assisted Intervention. pp. 773–780. Springer (2005)
6. Ioffe, S., Szegedy, C.: Batch normalization: Accelerating deep network training by reducing internal covariate shift. arXiv:1502.03167 (2015)
7. Litjens, G., Kooi, T., Bejnordi, B.E., Setio, A.A.A., Ciompi, F., Ghafoorian, M., van der Laak, J.A., van Ginneken, B., Sánchez, C.I.: A survey on deep learning in medical image analysis. Medical Image Analysis 42, 60–88 (2017)
8. Litjens, G., Toth, R., van de Ven, W., Hoeks, C., Kerkstra, S., van Ginneken, B., Vincent, G., Guillard, G., Birbeck, N., Zhang, J., et al.: Evaluation of prostate segmentation algorithms for MRI: the PROMISE12 challenge. Medical Image Analysis 18(2), 359–373 (2014)
9. Mahadevan, S., Voigtlaender, P., Leibe, B.: Iteratively trained interactive segmentation. arXiv:1805.04398 (2018)
10. Nickisch, H., Rother, C., Kohli, P., Rhemann, C.: Learning an interactive segmentation system. In: Indian Conference on Computer Vision, Graphics and Image Processing. pp. 274–281. ACM (2010)
11. Pasquier, D., Lacornerie, T., Vermandel, M., Rousseau, J., Lartigau, E., Betrouni, N.: Automatic segmentation of pelvic structures from magnetic resonance images for prostate cancer radiotherapy. Int J Radiat Oncol 68(2), 592–600 (2007)
12. Paszke, A., Gross, S., Chintala, S., Chanan, G., Yang, E., DeVito, Z., Lin, Z., Desmaison, A., Antiga, L., Lerer, A.: Automatic differentiation in pytorch. In: NIPS-W (2017)
13. Ronneberger, O., Fischer, P., Brox, T.: U-net: Convolutional networks for biomedical image segmentation. In: Medical Image Computing and Computer-Assisted Intervention. pp. 234–241. Springer (2015)
14. Rother, C., Kolmogorov, V., Blake, A.: GrabCut: Interactive foreground extraction using iterated graph cuts. In: ACM Transactions on Graphics (TOG). vol. 23, pp. 309–314. ACM (2004)
15. Srivastava, N., Hinton, G., Krizhevsky, A., Sutskever, I., Salakhutdinov, R.: Dropout: A simple way to prevent neural networks from overfitting. The Journal of Machine Learning Research 15(1), 1929–1958 (2014)
16. Tian, Z., Liu, L., Zhang, Z., Fei, B.: PSNet: prostate segmentation on MRI based on a convolutional neural network. Journal of Medical Imaging 5(2), 021208 (2018)
17. Toth, R., Bloch, B.N., Genega, E.M., Rofsky, N.M., Lenkinski, R.E., Rosen, M.A., Kalyanpur, A., Pungavkar, S., Madabhushi, A.: Accurate prostate volume estimation using multifeature active shape models on T2-weighted MRI. Academic Radiology 18(6), 745–754 (2011)
18. Vos, P., Barentsz, J., Karssemeijer, N., Huisman, H.: Automatic computer-aided detection of prostate cancer based on multiparametric magnetic resonance image analysis. Physics in Medicine & Biology 57(6), 1527 (2012)
19. Wang, G., Li, W., Zuluaga, M.A., Pratt, R., Patel, P.A., Aertsen, M., et al.: Interactive medical image segmentation using deep learning with image-specific finetuning. IEEE Trans Med Imag (2018)
20. Wang, G., Zuluaga, M.A., Li, W., Pratt, R., Patel, P.A., Aertsen, M., Doel, T., Divid, A.L., Deprest, J., Ourselin, S., et al.: DeepIGeoS: a deep interactive geodesic framework for medical image segmentation. IEEE Tran Pattern Anal (2018)
21. Zhu, Q., Du, B., Turkbey, B., Choyke, P.L., Yan, P.: Deeply-supervised CNN for prostate segmentation. In: Int. Joint Conference on Neural Networks. pp. 178–184. IEEE (2017)