# 3D U-Net: Learning Dense Volumetric Segmentation from Sparse Annotation

Özgün Çiçek[1,2], Ahmed Abdulkadir[1,4], Soeren S. Lienkamp[2,3], Thomas Brox[1,2], and Olaf Ronneberger[1,2,5]

[1] Computer Science Department, University of Freiburg, Germany
[2] BIOSS Centre for Biological Signalling Studies, Freiburg, Germany
[3] University Hospital Freiburg, Renal Division, Faculty of Medicine, University of Freiburg, Germany
[4] Department of Psychiatry and Psychotherapy, University Medical Center Freiburg, Germany
[5] Google DeepMind, London, UK
cicek@cs.uni-freiburg.de

**Abstract.** This paper introduces a network for volumetric segmentation that learns from sparsely annotated volumetric images. We outline two attractive use cases of this method: (1) In a semi-automated setup, the user annotates some slices in the volume to be segmented. The network learns from these sparse annotations and provides a dense 3D segmentation. (2) In a fully-automated setup, we assume that a representative, sparsely annotated training set exists. Trained on this data set, the network densely segments new volumetric images. The proposed network extends the previous u-net architecture from Ronneberger et al. by replacing all 2D operations with their 3D counterparts. The implementation performs on-the-fly elastic deformations for efficient data augmentation during training. It is trained end-to-end from scratch, i.e., no pre-trained network is required. We test the performance of the proposed method on a complex, highly variable 3D structure, the *Xenopus* kidney, and achieve good results for both use cases.

**Keywords:** Convolutional Neural Networks, 3D, Biomedical Volumetric Image Segmentation, Xenopus Kidney, Semi-automated, Fully-automated, Sparse Annotation

## 1 Introduction

Volumetric data is abundant in biomedical data analysis. Annotation of such data with segmentation labels causes difficulties, since only 2D slices can be shown on a computer screen. Thus, annotation of large volumes in a slice-by-slice manner is very tedious. It is inefficient, too, since neighboring slices show almost the same information. Especially for learning based approaches that require a significant amount of annotated data, full annotation of 3D volumes is not an effective way to create large and rich training data sets that would generalize well.

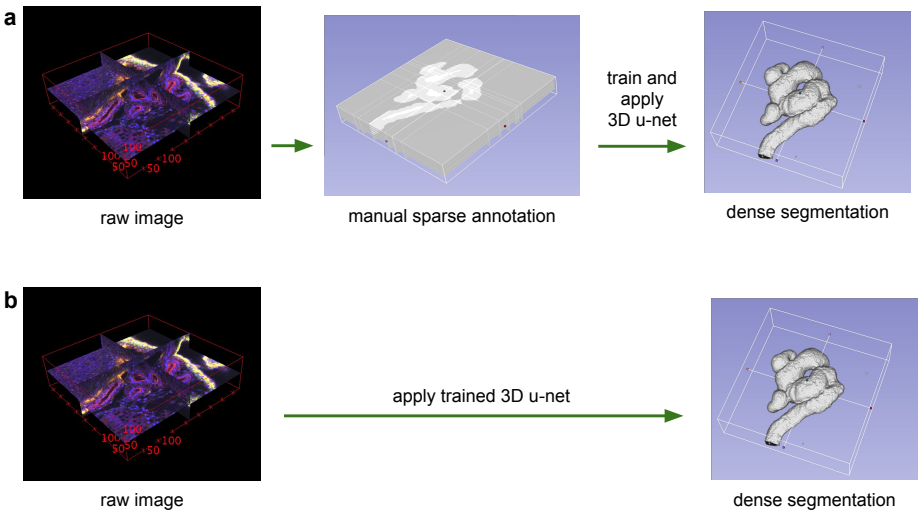



Fig. 1: Application scenarios for volumetric segmentation with the 3D u-net. (a) Semi-automated segmentation: the user annotates some slices of each volume to be segmented. The network predicts the dense segmentation. (b) Fully-automated segmentation: the network is trained with annotated slices from a representative training set and can be run on non-annotated volumes.

In this paper, we suggest a deep network that learns to generate dense volumetric segmentations, but only requires some annotated 2D slices for training. This network can be used in two different ways as depicted in Fig. 1: the first application case just aims on densification of a sparsely annotated data set; the second learns from multiple sparsely annotated data sets to generalize to new data. Both cases are highly relevant.

The network is based on the previous u-net architecture, which consists of a contracting encoder part to analyze the whole image and a successive expanding decoder part to produce a full-resolution segmentation [11]. While the u-net is an entirely 2D architecture, the network proposed in this paper takes 3D volumes as input and processes them with corresponding 3D operations, in particular, 3D convolutions, 3D max pooling, and 3D up-convolutional layers. Moreover, we avoid bottlenecks in the network architecture [13] and use batch normalization [4] for faster convergence.

In many biomedical applications, only very few images are required to train a network that generalizes reasonably well. This is because each image already comprises repetitive structures with corresponding variation. In volumetric images, this effect is further pronounced, such that we can train a network on just two volumetric images in order to generalize to a third one. A weighted loss function and special data augmentation enable us to train the network with only few manually annotated slices, i.e., from sparsely annotated training data.

We show the successful application of the proposed method on difficult confocal microscopic data set of the *Xenopus* kidney. During its development, the *Xenopus* kidney forms a complex structure [7] which limits the applicability of pre-defined parametric models. First we provide qualitative results to demonstrate the quality of the densification from few annotated slices. These results are supported by quantitative evaluations. We also provide experiments which shows the effect of the number of annotated slices on the performance of our network. The Caffe[5] based network implementation is provided as OpenSource[1].

### 1.1 Related Work

Challenging biomedical 2D images can be segmented with an accuracy close to human performance by CNNs today [11,12,3]. Due to this success, several attempts have been made to apply 3D CNNs on biomedical volumetric data. Milletari et al. [9] present a CNN combined with a Hough voting approach for 3D segmentation. However, their method is not end-to-end and only works for compact blob-like structures. The approach of Kleesiek et al. [6] is one of few end-to-end 3D CNN approaches for 3D segmentation. However, their network is not deep and has only one max-pooling after the first convolutions; therefore, it is unable to analyze structures at multiple scales. Our work is based on the 2D u-net [11] which won several international segmentation and tracking competitions in 2015. The architecture and the data augmentation of the u-net allows learning models with very good generalization performance from only few annotated samples. It exploits the fact that properly applied rigid transformations and slight elastic deformations still yield biologically plausible images. Up-convolutional architectures like the fully convolutional networks for semantic segmentation [8] and the u-net are still not wide spread and we know of only one attempt to generalize such an architecture to 3D [14]. In this work by Tran et al., the architecture is applied to videos and full annotation is available for training. The highlight of the present paper is that it can be trained from scratch on sparsely annotated volumes and can work on arbitrarily large volumes due to its seamless tiling strategy.

## 2 Network Architecture

Figure 2 illustrates the network architecture. Like the standard u-net, it has an analysis and a synthesis path each with four resolution steps. In the analysis path, each layer contains two $3 \times 3 \times 3$ convolutions each followed by a rectified linear unit (ReLu), and then a $2 \times 2 \times 2$ max pooling with strides of two in each dimension. In the synthesis path, each layer consists of an upconvolution of $2 \times 2 \times 2$ by strides of two in each dimension, followed by two $3 \times 3 \times 3$ convolutions each followed by a ReLu. Shortcut connections from layers of equal resolution in the analysis path provide the essential high-resolution features to

[1] `http://lmb.informatik.uni-freiburg.de/resources/opensource/unet.en.html`

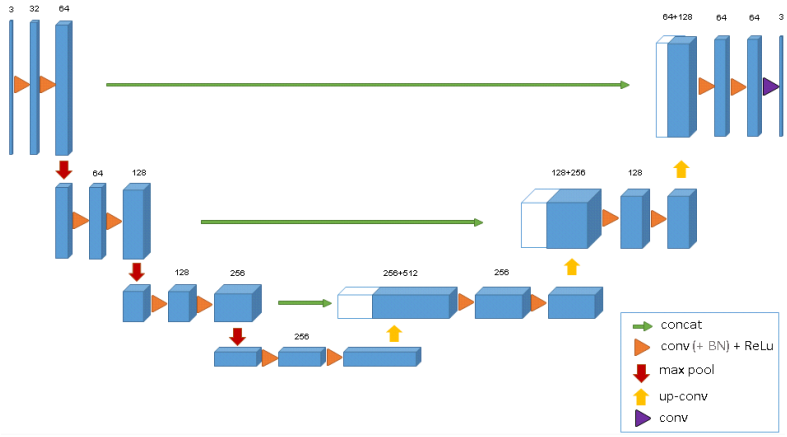



Fig. 2: The 3D u-net architecture. Blue boxes represent feature maps. The number of channels is denoted above each feature map.

the synthesis path. In the last layer a $1 \times 1 \times 1$ convolution reduces the number of output channels to the number of labels which is 3 in our case. The architecture has 19069955 parameters in total. Like suggested in [13] we avoid bottlenecks by doubling the number of channels already before max pooling. We also adopt this scheme in the synthesis path.

The input to the network is a $132 \times 132 \times 116$ voxel tile of the image with 3 channels. Our output in the final layer is $44 \times 44 \times 28$ voxels in x, y, and z directions respectively. With a voxel size of $1.76 \times 1.76 \times 2.04 \mu\text{m}^3$, the approximate receptive field becomes $155 \times 155 \times 180 \mu\text{m}^3$ for each voxel in the predicted segmentation. Thus, each output voxel has access to enough context to learn efficiently.

We also introduce batch normalization ("BN") before each ReLU. In [4], each batch is normalized during training with its mean and standard deviation and global statistics are updated using these values. This is followed by a layer to learn scale and bias explicitly. At test time, normalization is done via these computed global statistics and the learned scale and bias. However, we have a batch size of one and few samples. In such applications, using the current statistics also at test time works the best.

The important part of the architecture, which allows us to train on sparse annotations, is the weighted softmax loss function. Setting the weights of unlabeled pixels to zero makes it possible to learn from only the labelled ones and, hence, to generalize to the whole volume.

## 3 Implementation Details

### 3.1 Data

We have three samples of *Xenopus* kidney embryos at Nieuwkoop-Faber stage 36-37 [10]. One of them is shown in Fig. 1 (left). 3D Data have been recorded in

four tiles with three channels at a voxel size of $0.88 \times 0.88 \times 1.02 \mu m^3$ using a Zeiss LSM 510 DUO inverted confocal microscope equipped with a Plan-Apochromat 40x/1.3 oil immersion objective lens. We stitched the tiles to large volumes using XuvTools [1]. The first channel shows Tomato-Lectin coupled to Fluorescein at 488nm excitation wavelength. The second channel shows DAPI stained cell nuclei at 405 nm excitation. The third channel shows Beta-Catenin using a secondary antibody labelled with Cy3 at 564nm excitation marking the cell membranes. We manually annotated some orthogonal xy, xz, and yz slices in each volume using Slicer3D[2] [2]. The annotation positions were selected according to good data representation i.e. annotation slices were sampled as uniformly as possible in all 3 dimensions. Different structures were given the labels 0: "inside the tubule"; 1: "tubule"; 2: "background", and 3: "unlabeled". All voxels in the unlabelled slices also get the label 3 ("unlabeled"). We ran all our experiments on down-sampled versions of the original resolution by factor of two in each dimension. Therefore, the data sizes used in the experiments are $248 \times 244 \times 64$, $245 \times 244 \times 56$ and $246 \times 244 \times 59$ in $x \times y \times z$ dimensions for our sample 1, 2, and 3, respectively. The number of manually annotated slices in orthogonal (yz, xz, xy) slices are (7, 5, 21), (6, 7, 12), and (4, 5, 10) for sample 1, 2, and 3, respectively.

### 3.2 Training

Besides rotation, scaling and gray value augmentation, we apply a smooth dense deformation field on both data and ground truth labels. For this, we sample random vectors from a normal distribution with standard deviation of 4 in a grid with a spacing of 32 voxels in each direction and then apply a B-spline interpolation. The network output and the ground truth labels are compared using softmax with weighted cross-entropy loss, where we reduce weights for the frequently seen background and increase weights for the inner tubule to reach a balanced influence of tubule and background voxels on the loss. Voxels with label 3 ("unlabled") do not contribute to the loss computation, i.e. have a weight of 0. We use the stochastic gradient descent solver of the Caffe [5] framework for network training. To enable training of big 3D networks we used the memory efficient cuDNN[3] convolution layer implementation. Data augmentation is done on-the-fly, which results in as many different images as training iterations. We ran 70000 training iterations on an NVIDIA TitanX GPU, which took approximately 3 days.

## 4 Experiments

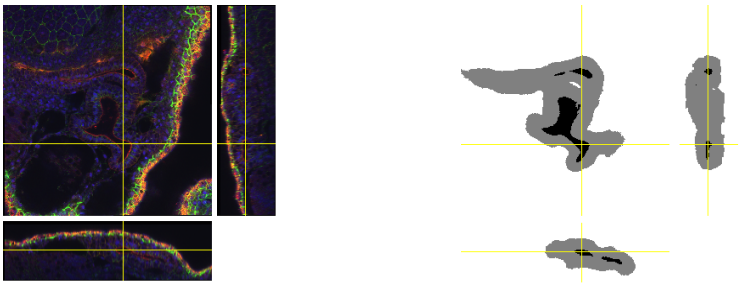
### 4.1 Semi-Automated Segmentation

For semi-automated segmentation, we assume that the user needs a full segmentation of a small number of volumetric images, and does not have prior

[2] `https://www.slicer.org`
[3] `https://developer.nvidia.com/cudnn`

segmentations. The proposed network allows the user to annotate a few slices from each volume and let the network create the dense volumetric segmentation.

For a *qualitative* assessment, we trained the network on all three sparsely annotated samples. Figure 3 shows the segmentation results for our 3rd sample. The network can find the whole 3D volume segmentation from a few annotated slices and saves experts from full volume annotation.



(a) (b)

Fig. 3: **(a)** The confocal recording of our 3rd *Xenopus* kidney. **(b)** Resulting dense segmentation from the proposed 3D u-net with batch normalization.

To assess the *quantitative* performance in the semi-automated setup, we uniformly partitioned the set of all 77 manually annotated slices from all 3 samples into three subsets and did a 3-fold cross validation both with and without batch normalization. To this end, we removed the test slices and let them stay unlabeled. This simulates an application where the user provides an even sparser annotation. To measure the gain from using the full 3D context, we compare the results to a pure 2D implementation, that treats all labelled slices as independent images. We present the results of our experiment in Table 1. Intersection over Union (IoU) is used as accuracy measure to compare dropped out ground truth slices to the predicted 3D volume. The IoU is defined as *true positives/(true positives + false negatives + false positives)*. The results show that our approach is able to already generalize from only very few annotated slices leading to a very accurate 3D segmentation with little annotation effort.

We also analyzed the effect of the number of annotated slices on the network performance. To this end, we simulated a one sample semi-automated segmentation. We started using 1 annotated slice in each orthogonal direction and increased the number of annotated slices gradually. We report the high performance gain of our network for each sample (S1, S2, and S3) with every few additional ground truth ("GT") slices in Table 2. The results are taken from networks which were trained for 10 hours with batch normalization. For testing we used the slices not used in any setup of this experiment.

Table 1: Cross validation results for semi-automated segmentation (IoU)

| test slices | 3D w/o BN | 3D with BN | 2D with BN |
|---|---|---|---|
| subset 1 | 0.822 | 0.855 | 0.785 |
| subset 2 | 0.857 | 0.871 | 0.820 |
| subset 3 | 0.846 | 0.863 | 0.782 |
| average | 0.842 | 0.863 | 0.796 |

Table 2: Effect of # of slices for semi-automated segmentation (IoU)

| GT slices | GT voxels | IoU S1 | IoU S2 | IoU S3 |
|---|---|---|---|---|
| 1,1,1 | 2.5% | 0.331 | 0.483 | 0.475 |
| 2,2,1 | 3.3% | 0.676 | 0.579 | 0.738 |
| 3,3,2 | 5.7% | 0.761 | 0.808 | 0.835 |
| 5,5,3 | 8.9% | 0.856 | 0.849 | 0.872 |

Table 3: Cross validation results for fully-automated segmentation (IoU)

| test volume | 3D w/o BN | 3D with BN | 2D with BN |
|---|---|---|---|
| 1 | 0.655 | 0.761 | 0.619 |
| 2 | 0.734 | 0.798 | 0.698 |
| 3 | 0.779 | 0.554 | 0.325 |
| average | 0.723 | 0.704 | 0.547 |

### 4.2 Fully-automated Segmentation

The fully-automated segmentation setup assumes that the user wants to segment a large number of images recorded in a comparable setting. We further assume that a representative training data set can be assembled.

To estimate the performance in this setup we trained on two (partially annotated) kidney volumes and used the trained network to segment the third volume. We report the result on all 3 possible combinations of training and test volumes. Table 3 summarizes the IoU as in the previous section over all annotated 2D slices of the left out volume. In this experiment BN also improves the result, except for the third setting, where it was counterproductive. We think that the large differences in the data sets are responsible for this effect. The typical use case for the fully-automated segmentation will work on much larger sample sizes, where the same number of sparse labels could be easily distributed over much more data sets to obtain a more representative training data set.

## 5 Conclusion

We have introduced an end-to-end learning method that semi-automatically and fully-automatically segments a 3D volume from a sparse annotation. It offers an accurate segmentation for the highly variable structures of the *Xenopus* kidney. We achieve an average IoU of 0.863 in 3-fold cross validation experiments for the semi-automated setup. In a fully-automated setup we demonstrate the performance gain of the 3D architecture to an equivalent 2D implementation. The network is trained from scratch, and it is not optimized in any way for this application. We expect that it will be applicable to many other biomedical volumetric segmentation tasks. Its implementation is provided as OpenSource.

**Acknowledgments.** We thank the DFG (EXC 294 and CRC-1140 KIDGEM Project Z02 and B07) for supporting this work. Ahmed Abdulkadir acknowledges funding by the grant KF3223201LW3 of the ZIM (Zentrales Innovationsprogramm Mittelstand). Soeren S. Lienkamp acknowledges funding from DFG (Emmy Noether-Programm). We also thank Elitsa Goykovka for the useful annotations and Alena Sammarco for the excellent technical assistance in imaging.

## References

1. Emmenlauer, M., Ronneberger, O., Ponti, A., Schwarb, P., Griffa, A., Filippi, A., Nitschke, R., Driever, W., Burkhardt, H.: Xuvtools: free, fast and reliable stitching of large 3d datasets. J Microscopy 233(1), 42–60 (2009)
2. Fedorov, A., Beichel, R., Kalpathy-Cramer, J., Finet, J., Fillion-Robin, J.C., Pujol, S., Bauer, C., Jennings, D., Fennessy, F., Sonka, M., et al.: 3D slicer as an image computing platform for the quantitative imaging network. J. Magn Reson Imaging 30(9), 1323–1341 (2012)
3. Hariharan, B., Arbeláez, P., Girshick, R., Malik, J.: Hypercolumns for object segmentation and fine-grained localization. In: Proc. CVPR. pp. 447–456 (2015)
4. Ioffe, S., Szegedy, C.: Batch normalization: Accelerating deep network training by reducing internal covariate shift. CoRR abs/1502.03167 (2015)
5. Jia, Y., Shelhamer, E., Donahue, J., Karayev, S., Long, J., Girshick, R., Guadarrama, S., Darrell, T.: Caffe: Convolutional architecture for fast feature embedding. In: Proc. ACMMM. pp. 675–678 (2014)
6. Kleesiek, J., Urban, G., Hubert, A., Schwarz, D., Maier-Hein, K., Bendszus, M., Biller, A.: Deep mri brain extraction: A 3d convolutional neural network for skull stripping. NeuroImage (2016)
7. Lienkamp, S., Ganner, A., Boehlke, C., Schmidt, T., Arnold, S.J., Schäfer, T., Romaker, D., Schuler, J., Hoff, S., Powelske, C., Eifler, A., Krönig, C., Bullerkotte, A., Nitschke, R., Kuehn, E.W., Kim, E., Burkhardt, H., Brox, T., Ronneberger, O., Gloy, J., Walz, G.: Inversin relays frizzled-8 signals to promote proximal pronephros development. PNAS 107(47), 20388–20393 (2010)
8. Long, J., Shelhamer, E., Darrell, T.: Fully convolutional networks for semantic segmentation. In: Proc. CVPR. pp. 3431–3440 (2015)
9. Milletari, F., Ahmadi, S., Kroll, C., Plate, A., Rozanski, V.E., Maiostre, J., Levin, J., Dietrich, O., Ertl-Wagner, B., Bötzel, K., Navab, N.: Hough-cnn: Deep learning for segmentation of deep brain regions in MRI and ultrasound. CoRR abs/1601.07014 (2016)
10. Nieuwkoop, P., Faber, J.: Normal Table of Xenopus laevis (Daudin)(Garland, New York) (1994)
11. Ronneberger, O., Fischer, P., Brox, T.: U-net: Convolutional networks for biomedical image segmentation. In: MICCAI. LNCS, vol. 9351, pp. 234–241. Springer (2015)
12. Seyedhosseini, M., Sajjadi, M., Tasdizen, T.: Image segmentation with cascaded hierarchical models and logistic disjunctive normal networks. In: Proc. ICCV. pp. 2168–2175 (2013)
13. Szegedy, C., Vanhoucke, V., Ioffe, S., Shlens, J., Wojna, Z.: Rethinking the inception architecture for computer vision. CoRR abs/1512.00567 (2015)
14. Tran, D., Bourdev, L.D., Fergus, R., Torresani, L., Paluri, M.: Deep end2end voxel2voxel prediction. CoRR abs/1511.06681 (2015)