KIDENY TUMOR SEGMENTATION WITH 3D-CONVOLUTIONAL NEURAL NETWORKS

Olexander Chepurnoi*
Ukrainian Catholic University
chepurny@ucu.edu.ua

Yaroslava Lochman*
Ukrainian Catholic University
lochman@ucu.edu.ua

April 2019

ABSTRACT

Abstract. The exploitation of machine learning tools in the medical sphere as well as in many other fields has evolved very quickly in the last decades and shown its efficiency and utility. There are still many specific medical imaging problems that would be great to solve automatically. Our team presents an approach to 3D image segmentation of the kidney and kidney tumor on CT scans of the human abdomen. This problem is presented in KiTS 2019 competition [8]. We analyzed and prepared highly sparse volumetric data and fed it to the 3D U-Net model. Having this we could get the predictions as segmentation masks of kidney/tumor objects on provided data. The implementation includes the sliding window approach with half strides to build a reliable prediction on high-resolution 3D images. The convolutional neural network is trained end-to-end from scratch, i.e., no pre-trained network is required. We tested the performance of the solution on the prepared validation dataset resulting in Sørensen–Dice coefficient about 0.7.

***K*eywords** Medical Volumetric Image Segmentation · 3D Convolutional Neural Network · 3D Kidney Segmentation · 3D Tumor Segmantation

1 Introduction and Related Work

This work covers a medical case of 3D image segmentation with specific convolutional neural network (CNN) applied. More precisely we solved kidney and kidney tumor segmentation problem. The project is done within KiTS19 Challenge organized by the University of Minnesota and University of Melbourne [8]. It was chosen because cancer is still one of the biggest problems in healthcare and we believe that automated medical image processing may fasten disease diagnostics and treatment. There are more than 400 000 new cases of kidney cancer each year[2]. And surgery is its most common treatment [3]. Kidney cancer is the ninth most commonly occurring cancer in men and the 14th most commonly occurring cancer in women. The top countries with the highest incidence of kidney cancer in 2018 are shown on the Figure 1. Automatic segmentation of the organ and its tumor is a promising tool. It can impact decision making processes that are now complex given the wide range of treatment options that are currently available.

Some of the earliest works (2009 year) on kidney tumor analysis first of all aimed to classify renal tumors based on their size and anatomical features and predict the risk of overall complications resulting from the nephron-sparing surgery based on the score. In [6] along with tumor size were analyzed: anterior or posterior face, longitudinal, and rim tumor location; tumor relationships with renal sinus or urinary collecting system; and percentage of tumor deepening into the kidney. An evaluating algorithm was implemented with multivariate analysis tools. In [11] the scoring system was created based on the following observed variables: maximal diameter of the tumor, exophytic or endophytic characteristics, nearness to the closest portion of the renal sinus or collecting system, location relative to the polar line and whether the tumor is anterior or posterior to the renal coronal plane.

*Authors contributed equally

[2] Kidney Cancer Statistics 2018, www.wcrf.org/dietandcancer/cancer-trends/kidney-cancer-statistics

[3] Cancer Diagnosis and Treatment Statistics 2017, www.cancerresearchuk.org/health-professional/cancer-statistics/diagnosis-and-treatment

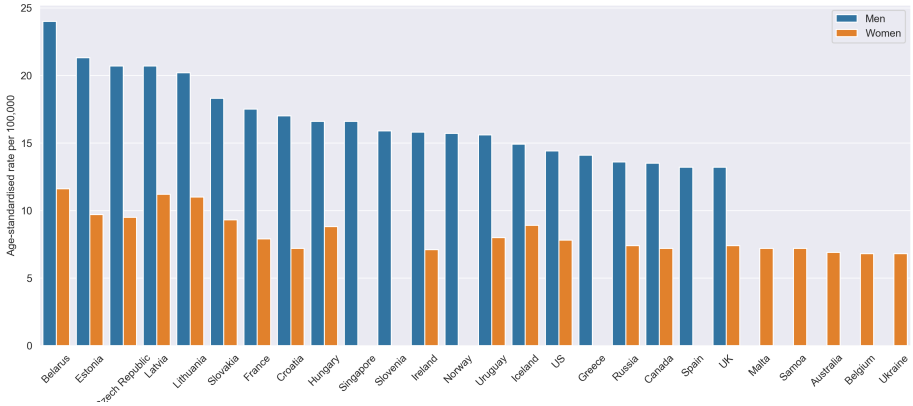


Figure 1: Kidney cancer rates in 2018. The data was drawn from www.wcrf.org/dietandcancer/cancer-trends/kidney-cancer-statistics

The 3D image segmentation problem is depicted on the Figure 2. For each input case which is a volumetric data (we will call it an input volume) the goal is to get a segmentation mask volume for kidney and a segmentation mask volume for tumor.

From the medical image segmentation side there exists a very well known fully convolutional neural network called U-Net [14] that solves the general segmentation task. Based on this architecture several models such as 3D U-Net [4], Residual Symmetric U-Net [12], V-Net [13] were created to deal with volumetric images. Authors also presented data augmentation and training pipelines to handle unbalanced data, reduce false positives and overcome a limited memory budget. There also exist ICv1 [2] and Deepem3d [16] neural networks with integrated inception and residual learning techniques and watershed algorithm applied to get accurate boundaries. The very close to our problem was effectively solved by Kid-Net [15] that handled aforementioned data and memory barriers as well.

In many CNN works the residual block introduced in ResNet [7] was widely used [13] [12] [15] [3] to facilitate training deep architecture. Alongside with 3D convolutions there exist approaches with recurrent plus 2D convolutional neural networks [1]. Feature accumulation with recurrent residual convolutional layers ensures better feature representation for segmentation tasks.

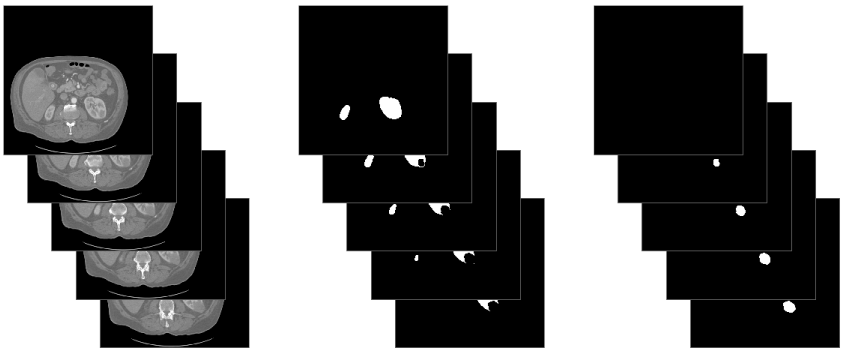


Figure 2: 3D image segmentation problem: an input is a volumetric data and can be considered as a collection of ordinary images; outputs are collections of binary masks corresponding to object classes.

2 Data

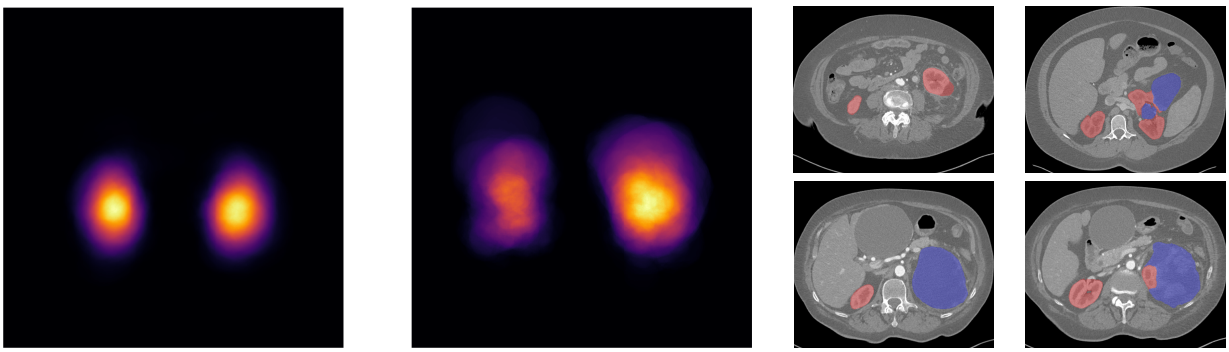
The annotated dataset consists of contrast-enhanced multi-phase abdominal Computed Tomography (CT) imaging plus kidney and kidney tumor segmentation masks for 300 patients who underwent nephrectomy for kidney tumors between 2010 and 2018 [8]. The length of the training set is 210 samples, and 90 samples will be used to compute final score.

Figure 3: Training data case example: a CT scan slice through an axial (left), sagittal (middle) and coronal (right) plane. Red annotated segments correspond to a kidney, blue – to a tumor.

2.1 Exploratory Analysis

The imaging as well as ground truth labels are provided in the anonymized NIFTI format having a shape number_of_slices $\times$ height $\times$ width. Here, number_of_slices corresponds to an axial plane view, and slices go from superior to inferior as the index increases. In all cases, the patient was supine during CT data collection, and thus height-width origins lie to the patient's left anterior. When there were multiple qualifying series for a particular case, those with the smaller slice thickness (the distance between centers of adjacent pixels in axial sections, mm) was chosen. The slice thicknesses range from 1 to 5 mm. For the training data example see Figure 3.

Raw Data and Interpolated Data There are two kinds of data provided. The first one was released earlier and corresponded to raw CT data. It consists of cases that for every patients have the same height and width 512×512 but different number of slices, slice thickness and pixel width. In fact it is unnormalized and we can't compare one case with another. The next kind is data interpolated in that way so the sizes (height, width, depth) are comparable – the units are the same for all cases, proportional to the millimeters in each dimension. Our first problem solving attempt was done with the raw data but further we switched to the interpolated by above reason.



(a) (b)

Figure 4: (a) Mean average kidney (left) and tumor (right) location in the axial plane. (b) Edge cases: location, intersecting, sizes, boundaries.

Location In every case a scan contains two kidneys, one of which has a tumor, the other doesn't. As can be seen on the Figure 4 (a) there is a certain location of concentration of kidneys which helps in sampling process of crops (see Section 2.2). However the relative locations of the left kidney to the right kidney may be different meaning that they are not mirrored that also affects the sampling process and data augmentation (we can't use horizontal flipping transformation).

For the example see top right image on the Figure 4 (b). The tumor location is not as uniform. Considering correct data collection we may conclude that it is focused more on the right side that also implies appropriate sampling.

Kidney-Tumor Boundary The kidney object and the tumor object should not overlap each other which complicates the task. It is not trivial where the kidney ends and tumor begins and there are even cases when they intersect. The Figure 4 (b) on the right illustrates some examples.

Data Balance Not only we are interested in foreground-background balance but also the ratio of kidney to tumor is important. In about 10% cases the tumors are bigger than whole kidneys (e.g. see bottom examples on the Figure 4 (b)). And as was analyzed earlier the locations should be taken into consideration.

2.2 Preprocessing

The training data was split according to the distribution of the kidney size and tumor size jointly. (See Figure 5). We report results in Section 4 on the validation set given by this split. Data leakage is avoided since we deal with full cases corresponding to different patients. At the same time the validation set is as representative as it can be having data provided by the organizers of the challenge.

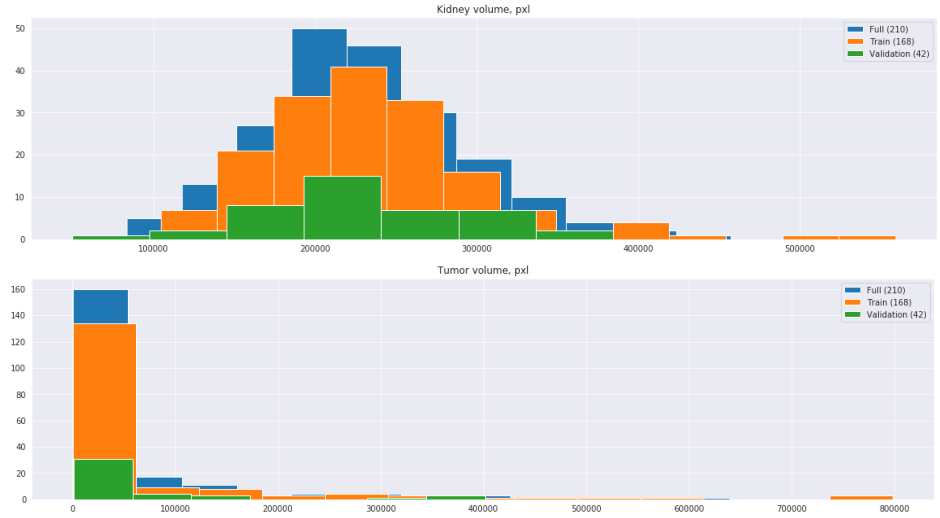


Figure 5: Distribution of kidney size (top) and tumor size (bottom) in the full 210 data samples and separated train and validation sets.

For the training subset the data preprocessing pipeline has the following steps:

- Generate all crop positions for window size $2K \times 2K \times K/2$ with stride $K \times K \times K/4$
- Calculate statistics for each segmentation crop (kidney volume, tumor volume)
- For the empty crops where kidney volume is 0, leave about 20% to avoid class imbalance
- Convert voxels of original volume data from NIFTI to the image tensor.
- Do the min-max normalization of voxels to avoid gradient explosion during the training.

For the validation subset the data preprocessing pipeline is identical except for there is no dropping of empty crops. For K we considered following values: 32, 64, 128.

2.3 Augmentation

To increase our model generalization capabilities we show more examples to the network by augmenting existed. We consider the following transformations:

- randomly rotate an image in range $[-10°, 10°]$
- randomly scale in range $[0.8, 1.2]$;

- randomly change the brightness in range $[-0.4, 0.4]$;
- we also use both simple random contrast change in range $[-0.4, 0.4]$ and contrast limited adaptive histogram equalization (CLAHE) with probability 0.5.

As one may notice the transformations are very slight, whereas adding the noise, smoothing the image, flip and affine transformations are inappropriate in biomedical imaging.

3 3D Convolutional Neural Network

3.1 Architecture overview

Although there are many possible architectures suitable for the segmentation task, most of them are developed for 2d segmentation. While we could adopt them to 3d segmentation by dealing with each slice independently, a lack of context due to the reduced amount of information from the slices, located nearly across the depth, could affect the results in a negative way. Because of that we decided to focus on architectures capable to do a segmentation of the whole 3D volumes.

Our main choice is 3DUnet [4] which is an adopted version of U-net [14] for 3D volumes. This network is based on the U-shaped architecture, which consists of an encoder part to analyze the whole tensor and compress it to the deep and dense representations, and following expanding path – decoder – to produce a full-resolution segmentation.

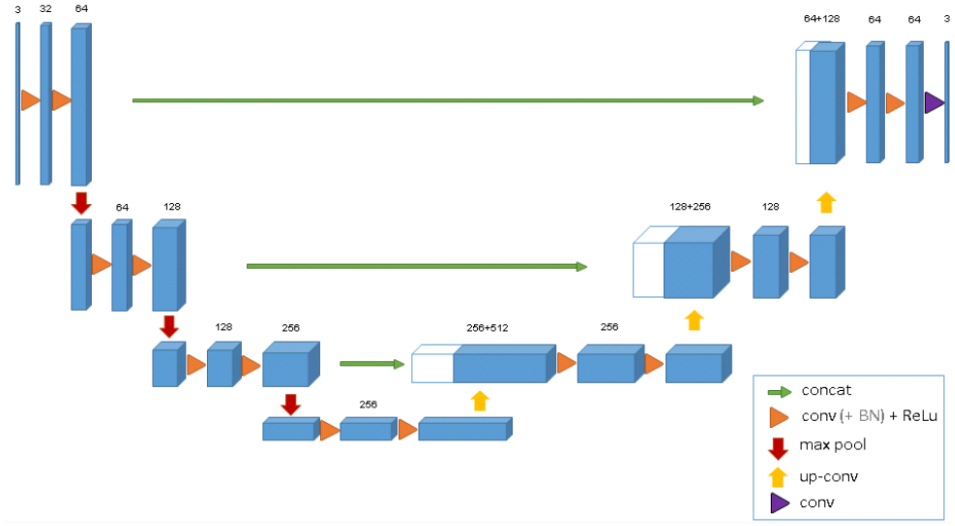


Figure 6: 3D U-Net architecture, thanks [4] for the image.

In the encoder part, each block contains two $3 \times 3 \times 3$ convolution layers each followed by a ReLU activation function, and then a $2 \times 2 \times 2$ max pooling with strides of 2 in each dimension.

In the decoder part, each block consists of an upconvolution $2 \times 2 \times 2$ (transposed convolution, for details see [5]) with strides of 2 in each dimension, followed by two $3 \times 3 \times 3$ convolutions + ReLU.

So the blocks are separated by max-pooling which is used to reduce the spatial size. Before applying a non-linearity there is a batch normalization layer. Hence we have a bunch of stacks of layers where the stack is: linearity, batch-normalization, activation. During the training in each such stack a batch is normalized with its mean and standard deviation, then the distribution is corrected with batch normalization layer parameters that are learned (for more details see [9]).

In the last layer a $1 \times 1 \times 1$ convolution performs filtering across the channels depth, reducing it to the number of labels which is 3 in our case: kidney, tumor, background.

In general our network has approximately 17 million parameters.

3.2 The learning process

The pipeline is quite hard because we are working with very big volumes of data. The main limitation is memory size of the GPU so we have to do some optimization to fit training examples in memory.

3.2.1 Training

The weights and bias in convolutions layers are initialized by sampling from uniform distribution $U(-\frac{1}{C}, \frac{1}{C})$, where C is the number of filters (output channels); for batch normalization layers the weight multiplier is initialized from $U(0, 1)$, the bias is zero.

To overcome the limitations of the memory size we train our model on the volume crops. This is possible because our neural network is fully convolutional, we can have the input of the arbitrary size (as long as we don't rescale it), and it's equivariant to translations. We have generated positions of crops with size $H \times W \times D$ and step $H_s \times W_s \times D_s$. It is very important to have overlapping to avoid issues with the prediction on the border.

As input for our training pipeline, we have a batch of $1 \times D \times H \times W$ (here 1 is channel size) crops and on the output, we have a batch of predictions of size $3 \times D \times H \times W$.

Settings for $H \times W \times D$ / $H_s \times W_s \times D_s$ we considered (ordered from the earliest to the last recent and more effective):

- $64 \times 64 \times 64$ / $32 \times 32 \times 32$
- $64 \times 64 \times 16$ / $32 \times 32 \times 8$
- $128 \times 128 \times 32$ / $64 \times 64 \times 16$
- $256 \times 256 \times 64$ / $128 \times 128 \times 32$

As a loss function, we used cross entropy function. We trained with Adam optimizer [10] for $8000+$ iterations, the mini-batch size was up to 8 depending on the crop setting. The learning rate was 0.001, and we also used learning rate scheduler that decreased it twice: first to 0.0001, then to 0.00005.

3.2.2 Validation

To evaluate the model we have to implement predicting function and calculate the score on the predictions. The scoring function is the mean of Kidney Sørensen–Dice and Tumor Sørensen–Dice and is provided by the competition organizers:

$$S = \frac{1}{N} \sum_{i=1}^{N} \frac{1}{2} \left(\frac{2\, n_{t,tp}^{(i)}}{2\, n_{t,tp}^{(i)} + n_{t,fp}^{(i)} + n_{t,fn}^{(i)}} + \frac{2\, n_{k,tp}^{(i)}}{2\, n_{k,tp}^{(i)} + n_{k,fp}^{(i)} + n_{k,fn}^{(i)}} \right)$$

where the first subscript letter indicates tumor (*t*) or kidney (*k*) class and the second indicates the prediction type (*tp* for true positive, *fp* – false positive, *fn* – false negative)

Evaluation implementation is a bit tricky due to memory limitations. We do the predictions with sliding window on the whole sample, sum up the predictions and average results in a statistically correct way. So the evaluation pipeline is following:

- Prepare validation sample crops
- Initialize an empty reference mask of an image size for storing prediction tensor
- Iterate over crops, add its prediction for each class to the corresponding place in the reference mask tensor
- Calculate entries (number of predictions per pixel)
- Get a mean of the predictions by the entries at each pixel in the mask

4 Experiments

During the project there were several preprocessing-training-evaluating iterations. First was an early attempt to train 3D U-Net on the full initial training data crops as described in Section 3.2.1.

The amended training data release was done due to mistakes in annotations in the mid-April and there were most of forces put in final training cases. One of the important advantages of the new release was the interpolated data described

in Section 2.1 that we decided to work with further. This allowed us to make a correct train-validation split and evaluate the model before testing.

We handled the unbalanced data with weighted cross-entropy loss and got the score 0.67 on validation and 0.77 on training data. For the first training of the model we used the following configuration: $64 \times 64 \times 64$ with step size equals to 32 and learning rate 0.001. We trained the network only on crops that have kidney on it and filtered out empty crops. We achieved 0.60 validation score. The training time for the neural network was approximately 12 hours on P100. The main insight of the first iteration was underfitting of the neural network. Also, training on crops with foreground data only resulted in false positive activations on images.

For the second training iteration we used crops of size $128 \times 128 \times 32$, weighted cross entropy loss with weights equals to $0.15, 1, 1$ for background, kidney, tumor classes respectively. Also, we have added a 20% of empty crops to the training data to improve the performance on volumes without kidney. Batch size of 4 used during training on P100 card for approximately 26 hours. During the training we started from learning rate 0.001. Then we lowered it to 0.0001 and for last two epochs we used 0.00005

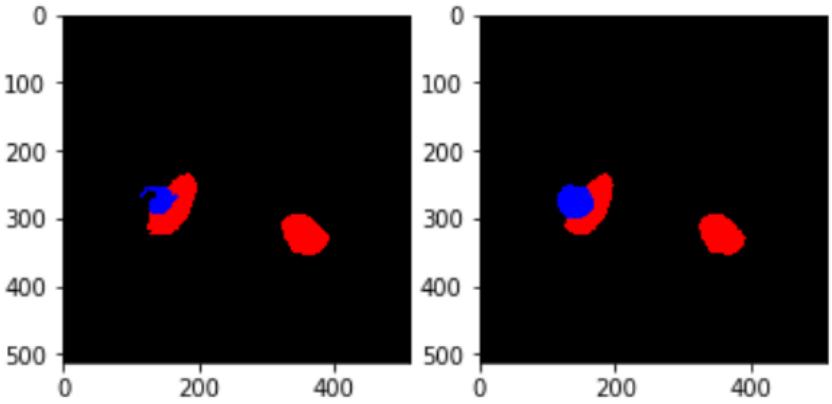


Figure 7: Predicted (left) segmentation mask vs Ground truth (right)

On the Figure 7 one can see a usual result of prediction – an example of good performance of the neural network. It recognizes the correct shape and position of the kidney and tumor. Tumor prediction is less accurate comparing to the kidney prediction.

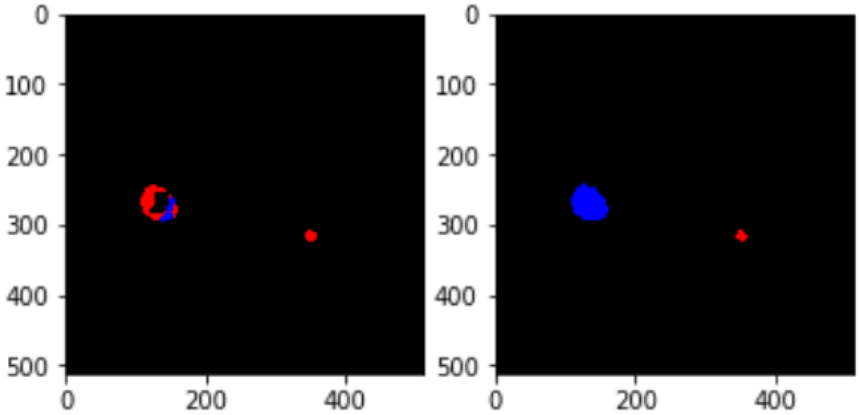


Figure 8: Predicted (left) segmentation mask vs Ground truth (right)

However, as one can see on the Figure 8, sometimes our model misclassify kidney and tumor and underrepresent tumors.

5 Conclusions

We have implemented the pipeline for kidney and tumor segmentation of highly sparse CT volumes. We achieved a 0.7 score on validation data with 3D U-Net using sliding window of size $128 \times 128 \times 32$. We are still experimenting with other settings such as $256 \times 256 \times 64$ and more up to a maximum size allowed with a minibatch 1.

We would like to highlight our intended directions to improve the network efficiency:

- Increase window size – it might improve the evaluation score by providing more information during prediction step. To implement this network size should be decreased or the GPU with more memory should be used. As was mentioned it's reasonable to take the maximum window size that could fit GPU memory with batch size equal to 1
- Integrate residual, inception blocks, try bilinear interpolation and other convolutional techniques for upsampling
- Use attention blocks
- Use other loss function, primarily Dice loss, and also Surface loss, Focal loss. This is very important for training and can significantly boost results.

We believe that combining the above methods with a properly trained neural network with learning schedule can better the model and increase a score up to $0.8 - 0.9$.

References

[1] Alom, M.Z., Hasan, M., Yakopcic, C., Taha, T.M., Asari, V.K.: Recurrent residual convolutional neural network based on u-net (r2u-net) for medical image segmentation. arXiv preprint arXiv:1802.06955 (2018)

[2] Beier, T., Pape, C., Rahaman, N., Prange, T., Berg, S., Bock, D.D., Cardona, A., Knott, G.W., Plaza, S.M., Scheffer, L.K., et al.: Multicut brings automated neurite segmentation closer to human performance. Nature Methods **14**(2), 101 (2017)

[3] Chen, H., Dou, Q., Yu, L., Qin, J., Heng, P.A.: Voxresnet: Deep voxelwise residual networks for brain segmentation from 3d mr images. NeuroImage **170**, 446–455 (2018)

[4] Çiçek, Ö., Abdulkadir, A., Lienkamp, S.S., Brox, T., Ronneberger, O.: 3d u-net: learning dense volumetric segmentation from sparse annotation. In: International conference on medical image computing and computer-assisted intervention, pp. 424–432. Springer (2016)

[5] Dumoulin, V., Visin, F.: A guide to convolution arithmetic for deep learning. arXiv preprint arXiv:1603.07285 (2016)

[6] Ficarra, V., Novara, G., Secco, S., Macchi, V., Porzionato, A., De Caro, R., Artibani, W.: Preoperative aspects and dimensions used for an anatomical (padua) classification of renal tumours in patients who are candidates for nephron-sparing surgery. European urology **56**(5), 786–793 (2009)

[7] He, K., Zhang, X., Ren, S., Sun, J.: Deep residual learning for image recognition. In: Proceedings of the IEEE conference on computer vision and pattern recognition, pp. 770–778 (2016)

[8] Heller, N., Sathianathen, N., Kalapara, A., Walczak, E., Moore, K., Kaluzniak, H., Rosenberg, J., Blake, P., Rengel, Z., Oestreich, M., et al.: The kits19 challenge data: 300 kidney tumor cases with clinical context, ct semantic segmentations, and surgical outcomes. arXiv preprint arXiv:1904.00445 (2019)

[9] Ioffe, S., Szegedy, C.: Batch normalization: Accelerating deep network training by reducing internal covariate shift. arXiv preprint arXiv:1502.03167 (2015)

[10] Kingma, D.P., Ba, J.: Adam: A method for stochastic optimization. arXiv preprint arXiv:1412.6980 (2014)

[11] Kutikov, A., Uzzo, R.G.: The renal nephrometry score: a comprehensive standardized system for quantitating renal tumor size, location and depth. The Journal of urology **182**(3), 844–853 (2009)

[12] Lee, K., Zung, J., Li, P., Jain, V., Seung, H.S.: Superhuman accuracy on the snemi3d connectomics challenge. arXiv preprint arXiv:1706.00120 (2017)

[13] Milletari, F., Navab, N., Ahmadi, S.A.: V-net: Fully convolutional neural networks for volumetric medical image segmentation. In: 2016 Fourth International Conference on 3D Vision (3DV), pp. 565–571. IEEE (2016)

[14] Ronneberger, O., Fischer, P., Brox, T.: U-net: Convolutional networks for biomedical image segmentation. In: International Conference on Medical image computing and computer-assisted intervention, pp. 234–241. Springer (2015)

[15] Taha, A., Lo, P., Li, J., Zhao, T.: Kid-net: convolution networks for kidney vessels segmentation from ct-volumes. In: International Conference on Medical Image Computing and Computer-Assisted Intervention, pp. 463–471. Springer (2018)

[16] Zeng, T., Wu, B., Ji, S.: Deepem3d: approaching human-level performance on 3d anisotropic em image segmentation. Bioinformatics **33**(16), 2555–2562 (2017)