Scientific Research Publishing

International Journal of Medical Physics, Clinical Engineering and Radiation Oncology, 2021, 10, 81-93
https://www.scirp.org/journal/ijmpcero
ISSN Online: 2168-5444
ISSN Print: 2168-5436

Auto-Segmentation on Liver with U-Net and Pixel De-Convolutional Network

Huan Yao[1*], Jenghwa Chang[1,2,3]

[1]Department of Radiation Medicine, Center for Advanced Medicine, Northwell Health, NY, USA
[2]Department of Radiation Medicine, Donald and Barbara Zucker School of Medicine at Hofstra/Northwell, NY, USA
[3]Department of Physics and Astronomy, Hofstra University, NY, USA
Email: *hyao1@northwell.edu, jchang24@northwell.edu

How to cite this paper: Yao, H. and Chang, J. (2021) Auto-Segmentation on Liver with U-Net and Pixel De-Convolutional Network. *International Journal of Medical Physics, Clinical Engineering and Radiation Oncology*, **10**, 81-93.
https://doi.org/10.4236/ijmpcero.2021.102008

Received: March 23, 2021
Accepted: May 25, 2021
Published: May 28, 2021

Abstract

Purpose: To improve the liver auto-segmentation performance of three-dimensional (3D) U-net by replacing the conventional up-sampling convolution layers with the Pixel De-convolutional Network (PDN) that considers spatial features. **Methods**: The U-net was originally developed to segment neuronal structure with outstanding performance but suffered serious artifacts from indirectly unrelated adjacent pixels in its up-sampling layers. The hypothesis of this study was that the segmentation quality of the liver could be improved with PDN in which the up-sampling layer was replaced by a pixel de-convolution layer (PDL). Seventy-eight plans of abdominal cancer patients were anonymized and exported. Sixty-two were chosen for training two networks: 1) 3D U-Net, and 2) 3D PDN, by minimizing the Dice loss function. The other sixteen plans were used to test the performance. The similarity Dice and Average Hausdorff Distance (AHD) were calculated and compared between these two networks. **Results**: The computation time for 62 training cases and 200 training epochs was about 30 minutes for both networks. The segmentation performance was evaluated using the remaining 16 cases. For the Dice score, the mean ± standard deviation were 0.857 ± 0.011 and 0.858 ± 0.015 for the PDN and U-Net, respectively. For the AHD, the mean ± standard deviation were 1.575 ± 0.373 and 1.675 ± 0.769, respectively, corresponding to an improvement of 6.0% and 51.5% of mean and standard deviation for the PDN. **Conclusion**: The PDN has outperformed the U-Net on liver auto-segmentation. The predicted contours of PDN are more conformal and smoother when compared with the U-Net.

Keywords

Liver Auto-Segmentation, Deep-Learning, U-Net, Pixel-Deconvolutional Network

1. Introduction

Liver cancer is the fifth most common cancer in men and the ninth in women in the world. It was estimated that 841,100 new liver cancer cases were diagnosed in 2018 [1], and men were twice more likely than women to develop liver cancer. Worldwide liver cancer was respectively the second- and sixth-leading causes of cancer death in men and women, with an estimate of 781,600 deaths in 2018. In the United States, age-adjusted incidence rates of liver cancer were more than tripled between 1975 and 2014 [2].

Treatments of liver cancer include surgical resection, transplantation, and radiotherapy. Accurate identification and delineation of the liver from computed tomography (CT) images is the key to the success of these procedures. Although manual delineation of the liver boundaries by experienced radiologists gives accurate contours of the liver, it is very time-consuming (about 25 minutes) given the liver is the largest organ in the human body [3]. In addition, human contouring tends to be subjective and is strongly dependent on the CT image quality. As a result, the intra- and inter-observer variability is usually high. Therefore, semiautomatic or automatic liver segmentation is very desirable and meaningful in the clinical management of liver cancers.

However, there are several challenges in achieving accurate computer-aided liver segmentation. First, the contrast between the liver and surrounding tissues is usually low, which leads to fuzzy boundaries and makes the liver difficult to identify. In addition, liver pathologies (e.g. liver tumors) and high-intensity intrahepatic veins usually produce complicated intensity distributions and heterogeneous appearances.

In the past few decades, a variety of approaches have been proposed to segment the liver from CT images. In 2008, Heimann *et al.* [4] published a comprehensive review of the advantages and disadvantages for different techniques presented in MICCAI 3D Liver Tumor Segmentation Challenge, including statistical shape models, atlas registration, level-sets, graph-cuts, and rule-based algorithm. Later, deep neural networks (DNNs) gained increasing attention in the computer vision community because of their ability to learn features automatically from the data. More recently, the Liver Tumor Segmentation (LiTS) challenge was organized. All top-scoring automatic methods submitted to the two rounds organized in 2017 used DNNs. The winner of both rounds used the U-Net [5], a "fully convolutional network" based on Convolutional Neural Network (CNN [6]). CNN takes images as input and learns various features from these images to differentiate one from each other. U-Net gets its name because it further modifies and extends the architecture of CNN, making it a U-shaped architecture. U-Net can work with very few training images and still yields accurate segmentations.

In the U-Net architecture, the images were first down-sampled through a series of convolution layers, processed, then up-sampled to the original resolution. This up-sampling (or deconvolutional) operation is carried out by a series of

transposed convolution layer (TCL) [7]. Each TCL can produce a higher resolution output from the lower resolution input so the results can be concatenated (or added) to that of the same resolution layer from the down-sampling side. This concatenation or addition usually suffers from checkerboard artifacts [8], which greatly limit the capabilities of the network in generating smooth outputs on semantic segmentation. To overcome this problem, the Pixel De-convolutional layer (PDL) [9] was proposed to replace the up-sampling layers. The U-Net with the original up-sampling layers replaced with the PDLs is called Pixel-Deconvolutional-Net (PDN).

In this study, we evaluated the performance of the bare U-Net and PDN for liver segmentation. Seventy-eight radiotherapy plans of abdominal cancer patients were anonymized and exported for this study. The liver contours were extracted and used to train these two networks. The segmentation quality of these two networks was then evaluated and compared using various indexes including similarity Dice and average Hausdorff distance (AHD). Finally, the pros and cons of the U-Net and PDN are discussed.

2. Methods and Materials

2.1. The U-Net

Figure 1(a) and **Figure 1(b)** show the conceptual block diagrams of the two networks used in this study, in which the data are input from the "Input" node on the left, flowed toward the bottom and then toward the top, processed at each node and output to the "Output" node on the right side. The basic U-net [5] that employs fully convolutional network architecture is shown in **Figure 1(a)**. In **Figure 1(a)**, the input data after the batch-normalization [10] first go through a series of down-sampling operations, *i.e.*, the "Down_*n*" nodes on the left side. Each "Down_*n*" node has the same internal structures, including a convolution layer for feature extraction, followed by a dropout layer (drop rate = 0.5) to prevent the network from over-fitting, a Rectified Linear Unit (ReLU) [11] activation layer which is simply a piecewise function to keep only the positive value of input function to overcome the vanishing gradient problem, and a max pooling layer for down-sampling. The data then enter the "Bottom" node that has the same convolution operation but without the down-sampling layer.

The opposite operations of the convolutions in U-Net are a series of up-sampling (or deconvolutional) operations, *i.e.*, the "Up_*n*" nodes on the right side of **Figure 1(a)**. Similar to the "Down_*n*" nodes, each "Up_*n*" node has the same internal structures, particularly for up-sampling. Among these up-sampling nodes, the transposed convolution layer (TCL) [7] of the "Up_*n*" node is named as "Deconv" in **Figure 1(a)**.

TCL is mainly the opposite of convolution by applying a de-convolution matrix on the input image, plus up-sampling by dilation that produces a higher resolution output from the lower-resolution input. The resulting layer are then added to the output of the corresponding resolution layer on the left side (*i.e.*,

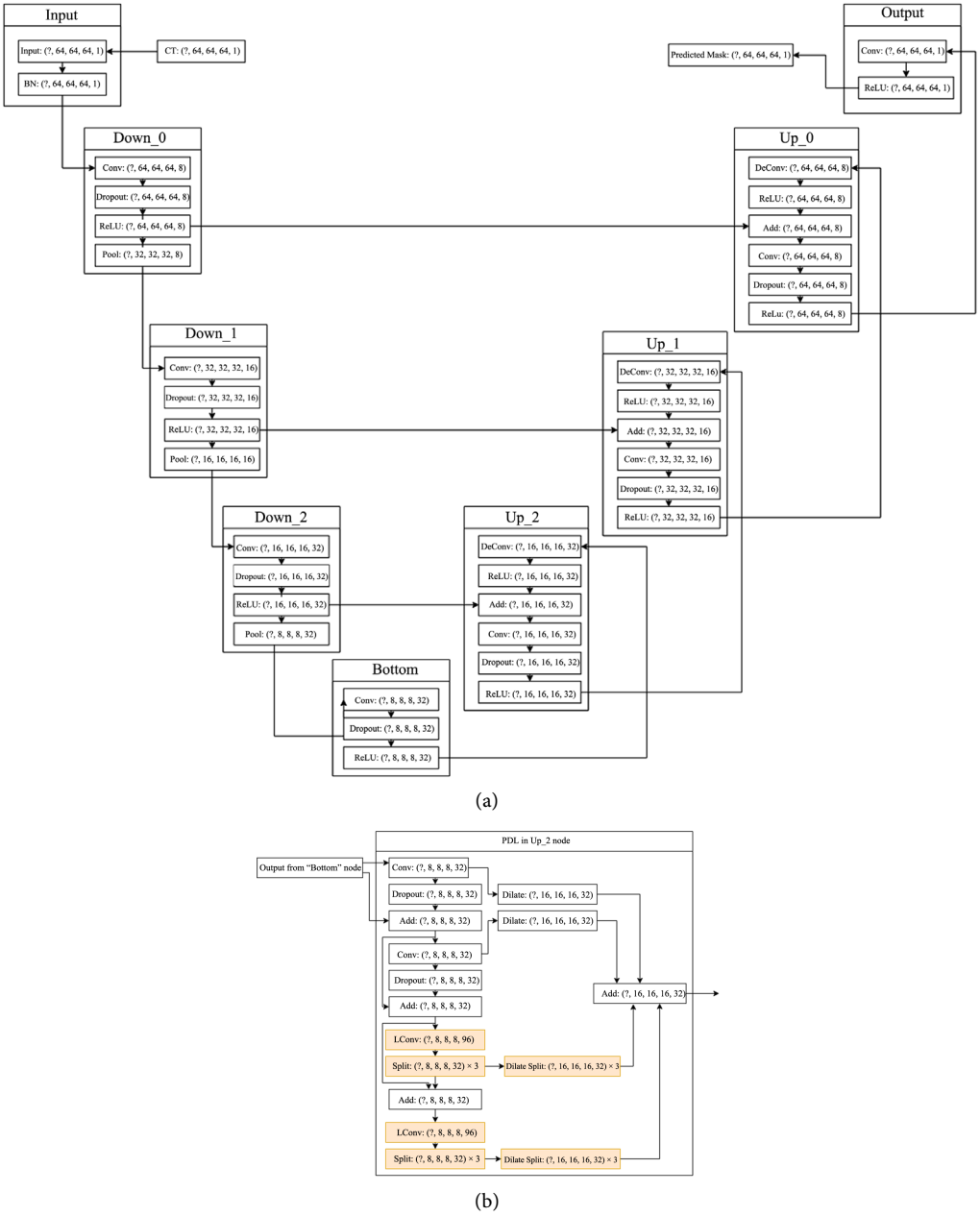

Figure 1. (a): Conceptual block diagrams of the basic U-Net. All the "Down_*n*" nodes have the same structures and so do all the "Up_*n*" nodes. Each large block with a name, for example, "Down_0" is a named node in the code. Within each node are multiple layers of codes, marked by various rectangles with layer names, one for each layer. "BN": batch-normalization layer. "Conv": convolution layer. "Dropout": dropout layer. "Add": sum of all input layers. "DeConv": de-convolution layer. "ReLu": ReLu activation layer. The "?" in the parenthesis after the layer name represents the number of cases/patients per step, e.g., 4 patients per steps in training or 1 patient per step in testing. The following three numbers are the output dimensions after the input matrix is processed through that layer. The last number represents the number of features or kernels used except that in the "CT" block it is the color channel. The arrow means the output from previous layer will be input into the next layer and the dimensions are indicated by the numbers in the parenthesis from the previous layer; (b): Conceptual block diagrams of the "PDL" layer in the "Up_2" node of the Pixel-Deconvolutional-Net (PDN), which replaces the "DeConv" layer in the "Up_2" node of the basic U-Net in (a). "LConv": a convolution layer with large number of features. "Dilate": dilation layer. "Split": split layer that splits the input to three equal size matrices. "Dilate Shift": a dilation operation followed by a one voxel shift operation along one of three axis (x/y/z). "Mask Convolution Layer": a combination of "LConv", "Split" and "Dilate Shift" highlighted in orange. The same "PDL" layer replaces the "Deconv" layer in all "Up_*n*" nodes in (a).

the "Dow_*n*" operations). This addition, usually called skip connection, is shown as the line from the "Down_*n*" layer to the corresponding "Up_*n*" layer in the U-Net of **Figure 1(a)**.

However, the outputs from these up-sampling nodes suffer from checkerboard artifacts that greatly limit the capabilities of the network model in generating smooth outputs on semantic segmentation. Detailed explanation of this artifact can be found in Odena, A., *et al.* [8]. In brief, as pointed out in Figure 2 and Figure 3 of Gao, H., *et al.* [9], standard deconvolutional operation can be decomposed into several convolutional operations, and these convoluted intermediate feature layers are shuffled and combined to produce the final layer. As a result, there is no direct relationship among these intermediate feature layers since they are generated by independent convolutional kernels. This independence usually causes uncorrelated values of adjacent pixels, leading to checkerboard artifacts [8], which greatly limit the capabilities of the network in generating smooth outputs on semantic segmentation.

2.2. The Pixel-Deconvolutional-Network (PDN)

To overcome the checkerboard problem, the Pixel De-convolutional layer (PDL) [9] was proposed to replace the TCL up-sampling layer in the up-sampling nodes so that the intermediate feature layers are generated sequentially from the previously generated ones. That is, the intermediate feature layers in a later stage are forced to depend on previous ones. This operation creates direct relationship between each intermediate feature layers. The U-Net with the original up-sampling layer replaced by the PDL is called the Pixel-Deconvolutional-Net (PDN) [9]. With PDN, the direct relationships between adjacent pixels on the output feature layers are established as explained in the following.

The up-sampling procedure of PDN is detailed using the "Up-2" up-sampling node in **Figure 1(a)** as an example by replacing the "Deconv" layer with the "PDL" layer in **Figure 1(b)**. As shown in **Figure 1(b)** the output of the "Bottom" layer was fed into the up-sampling PDL. The PDL applies two sequential convolution operations on the input layer [9]. These two convolution layers were then dilated to higher dimensions and input to the final "Add" layer. The dilation operation is done by adding zeros between two adjacent voxels in the input matrix so the output matrix dimensions are doubled, for example, from 8 × 8 × 8 to 16 ×16 × 16. The output of the second convolution layer is further processed by the two "Mask Convolution Layers". As shown in **Figure 1(b)**, the "Mask Convolution Layer" (orange rectangular plates) is a combination of "LConv", "Split" and "Dilate Shift". The "LConv" is a convolution that results in 3 times (*i.e.*, "×3" in the figure) the number of features of the input matrix (note that the number of features is indicated as the last number in the parenthesis after each layer name in **Figure 1(a)** and **Figure 1(b)**). For example, the input dimensions for "LConv" in **Figure 1(b)** is (?, 8, 8, 8, 32). After "LConv", the output dimensions will become (?, 8, 8, 8, 96). The ×3 is just because there are three Cartesian coordinate (x/y/z) for the matrix. The following "Split" operation splits this output into three

separate equal-sized matrix. The 3rd step, the "Dilate Shift" operation is a dilation followed by a one-voxel shift on the output matrix along one of x/y/z axis. As in the previous example, the (?, 8, 8, 8, 96) matrix will be split into three (?, 8, 8, 8, 32) matrices. Each matrix is dilated to size (?, 16, 16, 16, 32) and shifted one voxel along the x, y or z axis. So one "Mask Convolution Layer" generates three shifted dilated convoluted layers. Finally the two dilated layers and two shifted dilated convoluted layers are combined to form the intermediate output layer. The conceptual graph of the PDL is also shown in Figure 6 of Gao, H., *et al.* [9].

All codes were written in Python with TensorFlow 2.1 and running on a laptop with an Intel i7-7700HQ CPU, 16 GB memory and an NVIDIA GTX 1050 Ti GPU with 4 GB video memory. All the parameters in the network were chosen considering the achievable performance and endurability under these hardware limitations.

2.3. Dataset and Data Preprocessing

Seventy-eight liver cases were identified in the Eclipse treatment planning system of our facility and used to test the PDN. All patients received radiation treatments in the abdominal area and had a full contour of the liver. All contours in the data set were contoured by either a medical dosimetrist or a radiation oncologist and were reviewed and approved by the attending radiation oncologists of our facility during the daily smart rounds. The summary of the dataset was listed in **Table 1**.

For each case, the liver CT and structure sets were anonymized in the Eclipse treatment planning system [12] and exported to the local disk in DICOM RT format. With the help of SimpleITK [13] and Plastimatch [14], the CT DICOM

Table 1. Summary of the dataset consisting of the seventy-eight liver cases.

Variable	Cases	
	Number	%
Sex		
Male	33	42
Female	45	58
Treatment Site		
Abdomen	3	4
Gastric Bed	20	26
Liver	12	15
Pancreas	40	51
Para-aortic	3	4
Age		
<59	19	24
59 - 80	40	52
>80	19	24

files were transformed to 3D images and the liver contour in the DICOM RT files were converted to mask images. Both image sets were represented as 3D array. For the CT images, each voxel used two bytes to represent its gray scale, while for the mask image only one bit (1 or 0) was necessary because its function was to indicate if a voxel was inside or outside the liver contour. In compliance with the 3D image format of Tensorflow [15], one color dimension was also added to the 3D array to form a 4D array. Because the CT and mask images were both grayscale, the added color dimension was one.

The next step was to unify the dimensions of these seventy-eight image sets. Although the axial dimensions were the same 512 (Height) × 512 (Width), the number of slices for each scan was different, ranging from 100 to 250 with a mean of ~167. This inconsistence in slice number caused problems in programming the Tensorflow. In addition, our hardware computation power was not sufficient to hand the 512 × 512 axial dimensions. As a result, the original CT and mask images were converted to lower dimensions. This was achieved by resizing each image to low dimension levels, that is, 64 (Depth/Slice) × 64 (Height) × 64 (Width) × 1 (Color). B-spline interpolation was applied in the resizing operation to minimize the loss of image information due to dimension reduction. The TFRecord format, a simple format for storing a sequence of binary records, was used as the pipeline storage during the training or testing of the networks.

2.4. Network Training and Testing

Figure 2 shows the general process flow for network training and testing (or prediction) of this study. The exported data set, after going through the preprocessing as described above, was split into two groups: about 80% (62 cases) as the training set and the rest (16 cases) as the testing set. When the data of a training case were entered into the network, the 4D CT images were used as the input (*i.e.*, the input in **Figure 1(a)**) from which the network tried to segment the liver, and the liver 4D mask image as the label, *i.e.*, the ground truth of the contours, from which the network could compare and learn.

The network training was performed in a sequential manner by dividing the training process into multiple steps. That is, the training data set was first divided into subgroups with, e.g., four patients in each group. In each step, the network took one group (four patients) as input, predicted the mask image and compared the predicted mask with the label (ground truth) by calculating the Dice loss function between the two mask images. The parameters (*i.e.*, the weights/bias for segmentation) used by the networks for segmentation, were adjusted automatically by stochastic gradient descent (SGD) [16], to minimize the loss function. Once the training of the current step was done, the network took the next group of patients. The same sequence was repeated until the network finished one epoch, or one complete pass through the training set. The training process was stopped after 200 epochs, considering the balance between the length of the training time and the convergence of the loss function.

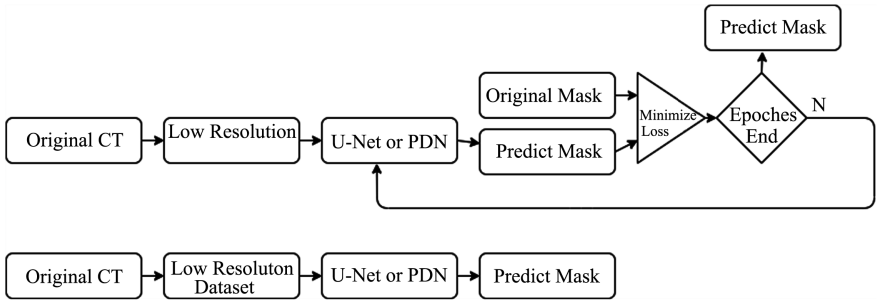

Figure 2. Process flow of this study. Top is for training and bottom is for prediction.

The prediction path was similar to the training path except for a few steps. First, when constructing the low-resolution dataset, no liver contouring was needed so only CT images were fed into the network. Second, the network parameters were fixed and not adjustable. Lastly, only one patient dataset was used for each prediction.

2.5. Performance Evaluation

The performance of the U-Net and PDN was evaluated for the low-dimension levels, *i.e.*, 64 (Depth/Slice) × 64 (Height) × 64 (Width) × 1 (Color) using the 16 cases in the testing set, after the networks were trained with the 62 cases in the training set. The similarity Dice score of segmented livers was calculated as a guideline for evaluating the segmentation quality of the predicted mask results.

In addition, the predicted mask image was resized back to the original dimensions (*i.e.* 512 × 512 × 64 × 1), which is called the "Contour" image, so that it could be compared with the original contour. For this dimension level, the Average Hausdorff Distance (AHD) was also calculated to further evaluate the segmentation results. The AHD measures the distance between the ground true set (X) (*i.e.*, label/mask images) and the predicted set (Y) (*i.e.*, predicted label from network). It involves the calculation of two directed HDs: 1) the directed HD from X to Y, given by the sum of the shortest distances between all points of X and Y divided by the number of points in X, and 2) the directed HD from Y to X, which is calculated in a similar way with the roles of X and Y switched. AHD is the mean of these two directed HDs. Obviously, the smaller the AHD is, the more similarity between the two sets. Therefore, the final metrics were the Dice score for the predicted mask images, and the Dice score and AHD for the "Contour" images.

3. Results

3.1. U-Net & PDN Comparison

The performance of these two networks was directly compared using the same training (the first 62 cases) and testing (the last 16 cases) sets, each with 200 training epochs. The mean metrics scores of the sixteen testing cases for each network are shown in Table 2. The training time of each network is also presented in the last column of Table 2.

Table 2. Mean ± standard deviation of the segmentation performance using the U-Net and PDN, with the first 62 cases being the training set, and the last 16 cases the testing set. A larger mean Dice score and a smaller mean AHD indicate better segmentation quality. The first Dice was computed for the predicted mask images directly from the output of the network with dimensions 64 × 64 × 64 × 1. The "Contour" metrics were computed after the predicted mask images were resized back to the original image dimensions (*i.e.* 512 × 512 × 64 × 1).

Network	Dice	Contour Dice	Contour AHD	Train (min)
PDN	0.865 ± 0.011	0.857 ± 0.011	1.575 ± 0.373	31.4 ± 0.08
U-Net	0.866 ± 0.016	0.858 ± 0.015	1.675 ± 0.769	27.9 ± 0.05

The similarity Dice score of predicted images and "Contour" images were almost the same for both networks. However the mean and standard deviation of "Contour AHD" for the PDN were respectively 6.0% and 51.5% better than that for the U-Net. Surprisingly, the training times for both networks were around 30 minutes for 200 epochs and only 3 more minutes for PDN although the PDL is much complicated than the TCL. The prediction time was negligible relative to the training time for both networks. Segmented livers on one slice of two testing cases for these two networks are shown in **Figure 3**.

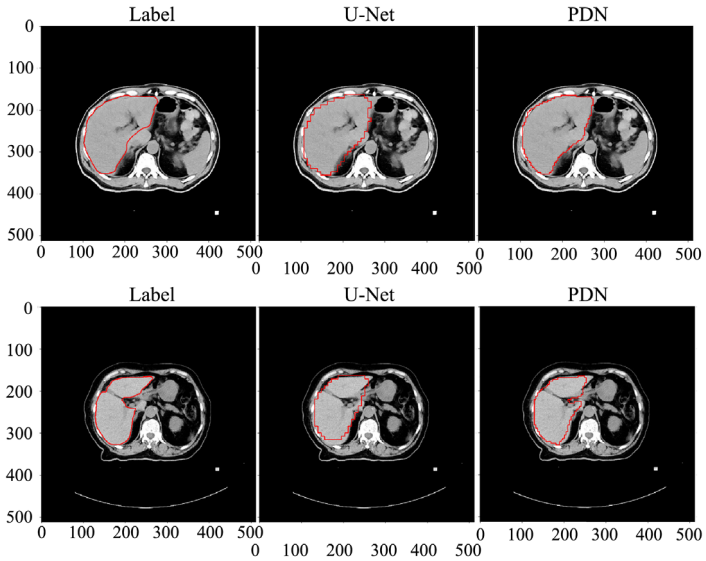

Figure 3. Segmented liver on one slice of two testing cases. From left to right were contoured by the expert (*i.e.*, Label), U-Net and PDN, respectively.

3.2. Challenging Cases

The segmentation results for two challenging cases, one with a fiducial and the other with a hole inside the contour are shown in **Figure 4**. As demonstrated in **Figure 4**, neither U-Net nor PDN was able to achieve similar segmentation of the liver containing a fiducial or a hole to that performed by human experts. However, the PDN did outperform the U-Net for the "hole" case as the PDN did not accidently include the contrast from stomach into the liver contour as seen in the lower panel of **Figure 4**.

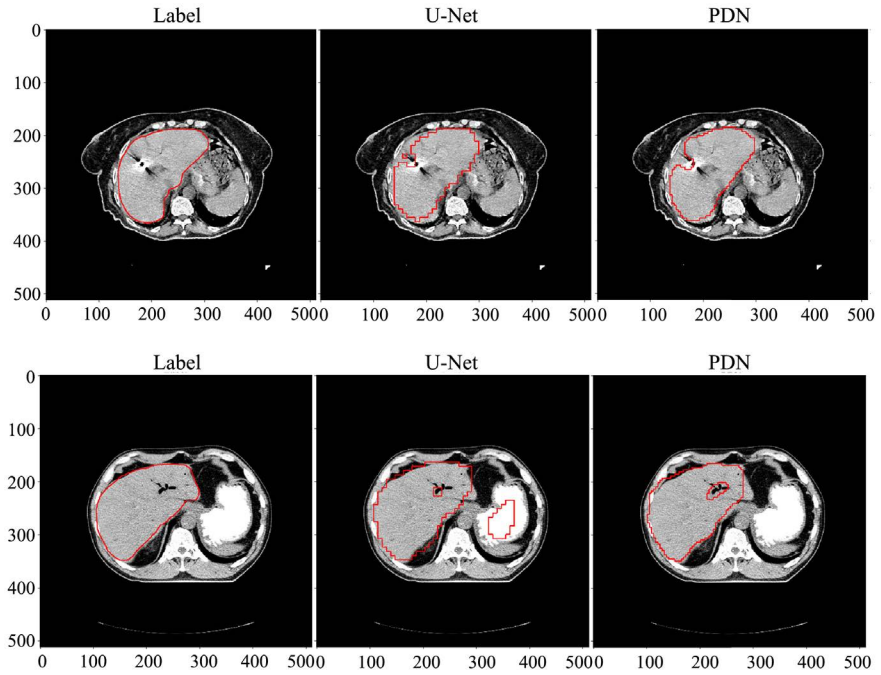

Figure 4. Segmentation results with a fiducial (upper panel) or a hole (lower panel) inside the liver. From left to right are contoured by the experts (*i.e.*, Label), U-Net and PDN, respectively.

4. Discussion

Results for the direct comparison of these two networks in Table 2, show that the PDN outperformed the U-Net for the "Contour AHD", but the Dice score was similar for these two networks. This difference could be explained by how these two image quality indexes were defined. The Dice score is related to the number of true positive, false positive and false negative of the predicted images. Given that most voxels in the ground truth and predicted images were background voxels (*i.e.*, not part of the contours and with a value of 0 for the color dimension), the number of true positive was relatively large. As a result, the Dice score only changed slightly even when the numbers of false positive and false negative decreased.

The AHD, on the other hand, is the average distance between the liver contours of the ground-truth and the predicted mask images and is independent of the background. The smaller "contour AHD" of the PDN in Table 2, therefore, indicated that the contours predicted by the PDN were closer to the ground truth than that by the U-net. This reduction of AHD was attributed to the fact that checkerboard artifacts in the predicted mask image from the U-Net were lessened by the PDN. This was evident in both Figure 3 and Figure 4. Visual inspection of the contours in both figures clearly demonstrated that the PDN could overcome the checkerboard artifacts and smooth the "zigzag" edges, and thus shorten the averaged distance between the predicted and ground truth mask images.

Better segmentation results do not mean that the PDN was a perfect learning

network. As shown in Figure 4, neither the U-Net nor the PDN was able to identify the fiducial and hole between liver lobes. We believe this is due to one simple fact. That is, the fiducials and holes are normally not contoured separately by human experts (in order to save time) and are included as part of the liver contours. As a result, the fiducials and holes were not contoured in the training cases of this study. During the testing, although both networks thought the fiducials and holes were not part of the liver, neither network had learned how to go around them. This problem might not be resolved soon once the fiducials and holes are not contoured by human experts. In the meantime, post image-processing can be used to the fix those incomplete contours due to the presence of fiducials and holes.

Although the liver segmentation method using the PDN described in this paper is promising, more work needs to be done to match the performance of human contouring. Moreover, a better performance metric is needed to assess the segmentation result when the volume of contours is much smaller than that of the whole CT volume so that the number of true positive won't be skewed by the presence of a large number of background voxels. At the same time, evaluations in clinical settings are required to measure its clinical efficacy for automatic liver segmentation. For future research, we plan to evaluate the performance including the accuracy and efficiency of PDN for more complex data sets by increasing the dimensions of input images. We also plan to further improve the optimization performance by including the AHD in the loss function and adding another network structure such as Recurrent Neural Networks (RNNs) that can memorize and learn from past events.

5. Conclusion

In conclusion, the PDN demonstrated superior performance for liver segmentation in comparison to the U-net. The liver contours segmented by the PDN have better quality and are more conformal. Particularly, the PDN can overcome the checkerboard artifacts and smooth the zigzag edges of the contours that are commonly seen in the contours segmented using the basic U-Net. This better performance is attributed to the PDN's establishment of direct relationships among adjacent pixels on the up-sampling path. The fiducials and holes inside the liver were not successfully identified during the auto-segmentation, which will be addressed in future studies.

Conflicts of Interest

The authors declare no conflicts of interest regarding the publication of this paper.

References

[1] Bray, F., *et al.* (2018) Global cancer statistics 2018: GLOBOCAN Estimates of Incidence and Mortality Worldwide for 36 Cancers in 185 Countries. *CA*: *A Cancer*

Journal for Clinicians, **68**, 394-424. https://doi.org/10.3322/caac.21492 https://www.ncbi.nlm.nih.gov/pubmed/30207593.

[2] Renehan, A.G., Zwahlen, M. and Egger, M. (2015) Adiposity and Cancer Risk: New Mechanistic Insights from Epidemiology. *Nature Reviews Cancer*, **15**, 484-498. https://doi.org/10.1038/nrc3967 https://www.ncbi.nlm.nih.gov/pubmed/26205341.

[3] Hermoye, L., *et al.* (2005) Liver Segmentation in Living Liver Transplant Donors: Comparison of Semiautomatic and Manual Methods. *Radiology*, **234**, 171-178. https://doi.org/10.1148/radiol.2341031801 https://www.ncbi.nlm.nih.gov/pubmed/15564393.

[4] Heimann, T., *et al.* (2009) Comparison and Evaluation of Methods for Liver Segmentation from CT Datasets. *IEEE Transactions on Medical Imaging*, **28**, 1251-1265. https://doi.org/10.1109/TMI.2009.2013851 https://www.ncbi.nlm.nih.gov/pubmed/19211338.

[5] Ronneberger, O., Fischer, P. and Brox, T. (2015) U-Net: Convolutional Networks for Biomedical Image Segmentation. arXiv E-Prints: 1505.04597. https://doi.org/10.1007/978-3-319-24574-4_28 https://ui.adsabs.harvard.edu/abs/2015arXiv150504597R

[6] Pinaya, W.H. L., *et al.* (2020) Convolutional Neural Networks. In: Mechelli, A. and Vieira, S., *Machine Learning*, Academic Press, Cambridge MA, USA, 173-191. https://doi.org/10.1016/B978-0-12-815739-8.00010-9

[7] Vedaldi, A. and Lenc, K. (2014) MatConvNet: Convolutional Neural Networks for MATLAB. arXiv E-Prints: 1412.4564. https://doi.org/10.1145/2733373.2807412 https://ui.adsabs.harvard.edu/abs/2014arXiv1412.4564V

[8] Odena, A., Dumoulin, V., and Olah, C. (2016) Deconvolution and Checkerboard Artifacts. *Distill*, **1**. https://doi.org/10.23915/distill.00003 http://distill.pub/2016/deconv-checkerboard

[9] Gao, H., *et al.* (2017) Pixel Deconvolutional Networks. arXiv E-Prints:1705.06820. https://ui.adsabs.harvard.edu/abs/2017arXiv170506820G

[10] Ioffe, S. and Szegedy, C. (2015) Batch Normalization: Accelerating Deep Network Training by Reducing Internal Covariate Shift. arXiv E-Prints: 1502.03167. https://ui.adsabs.harvard.edu/abs/2015arXiv150203167I

[11] Glorot, X., Bordes, A., and Bengio, Y. (2011) Deep Sparse Rectifier Neural Networks. *Proceedings of the Fourteenth International Conference on Artificial Intelligence and Statistics*, **15**, 315-323.

[12] (2012) Varian Eclipse Treatment Planning System. *Biomedical Safety & Standards*, **42**, 94 p. https://doi.org/10.1097/01.BMSAS.0000415578.78534.e8 https://journals.lww.com/biomedicalsafetystandards/Fulltext/2012/07010/Varian_Eclipse_Treatment_Planning_System.10.aspx.

[13] Yaniv, Z., *et al.* (2018) SimpleITK Image-Analysis Notebooks: A Collaborative Environment for Education and Reproducible Research. *Journal of Digital Imaging*, **31**, 290-303. https://doi.org/10.1007/s10278-017-0037-8 https://www.ncbi.nlm.nih.gov/pubmed/29181613.

[14] Shackleford, J.A., *et al.* (2012) Plastimatch 1.6—Current Capabilities and Future Directions. *Proceedings of the First International Workshop on Image-Guidance and Multimodal Dose Planning in Radiation Therapy*, Octorber 2012, Nice, France.

[15] Abadi, M., *et al.* (2016) TensorFlow: A System for Large-Scale Machine Learning. *Proceedings of the* 12*th USENIX Symposium on Operating Systems Design and*

Implementation, Savannah, GA, USA, 2-4 November 2016, 265-283.

[16] Kiefer, J. and Wolfowitz, J. (1952) Stochastic Estimation of the Maximum of a Regression Function. *The Annals of Mathematical Statistics*, **23**, 462-466. https://doi.org/10.1214/aoms/1177729392 https://projecteuclid.org:443/euclid.aoms/1177729392.