Instance-U-Net and Watershed: Improved Segmentations for breast cancer cells

Linfeng Yang
Bioengineering Department
Stanford University
linfeng@stanford.edu

Chenyu You
Bioengineering Department
Stanford University
uniycy@stanford.edu

Abstract

The segmentation and quantification of single cells in blood and tissue samples are critical to the early diagnosis of cancers and leukemia. To assist doctors in fast and accurate diagnosis of such diseases, an accurate segmentation algorithm that could separate densely touching cells is needed. In this work, we proposed a highly efficient and effective neural network model that has accurate predictions on both cell segmentation and cell count. Specifically, to capture each single cell in cell clusters, we integrated Region Proposal Networks (RPN) to detect cells with bounding boxes. To predict the cell segmentation, we integrate Instance-U-Net with pre-computed weight matrices. To further improve the boundary predictions of densely touching cells, we integrate Watershed algorithm with cell centroids (predicted by RPN) and cell segmentations (predicted by U-Net) as inputs. Extensive quantitative and qualitative evaluations illustrated that our proposed model can produce segmentation masks with more accurate single-cell boundaries than state-of-the-art segmentation models.

1. Introduction

The rapid development of microscopic imaging in modern clinic applications enables the fast and accurate diagnosis of diseases such as leukemia and cancer. Segmenting single cells from blood/tissue sample is critical to identify diseased cells to achieve early diagnosis. On the other side, the behaviors of single cells have brought more attention to biological researchers. For example, the lineage tracking of single embryo cells is vital to understand the mechanisms in early stage embryo developments; the shape analysis of mammalian cells during Epithelial-Mesenchymal transition is key to dissect the process of how normal cells become cancer cells.

Till now, identifying cells in images (cell segmentation) has been challenging due to the high density and imbalanced illumination of the sample. Recently, convolutional neural networks (CNNs) have became the instrumental tool for computer vision tasks [4, 11, 12]. Besides the improvement of computing power, the driven force is actually the implementation of new deep-learning models such as Fully Convolutional Network (FCN) [7]. However, segmenting biomedical samples is challenging, as the amount of data is very limited. At the meantime, some regions of the images are densely compacted with touching cells that are difficult to separate. To address these issues, here we propose a new model, which integrates U-Net and Region Proposal Network to accurately predict boundaries for single cells. The **input** to our network is an **40X fluorescent gray-scale cell nucleus image of size 448X448**, we then use RPN to predict the bounding boxes and centroids for each cell nucleus. Segmentation predictions were generated by U-Net with weight matrices. Then we integrate *Watershed* algorithm to further separate touching cells where the U-Net fails to separate. The **output** of our network is **a binary mask of the same size as the input, with the regions predicted to be cell necleus as** 1**, and background as** 0. Our method avoids the memory and segmentation limitations on the state-of-the-art semantic segmentation models, with less computational time and competitive accuracy.

2. Related Work

2.1. Fully Convolutional Network

The potential of using CNNs for semantic segmentation has first been shown by applying Fully Convolution Networks (FCN) to perform end-to-end, pixel-to-pixel segmentation [7]. This work sets the base line models and metrics for FCN-based segmentation approach. However, FCNs are initially designed for multi-class segmentation, and the networks actually focus on both the classification and segmentation. Moreover, the original FCN lacks the upsampling operators, therefore the output of FCN is less accurate compared with other state-of-the-art models.

2.2. U-Net

Due to the flexibility of computation powers, more powerful semantic segmentation systems have been developed

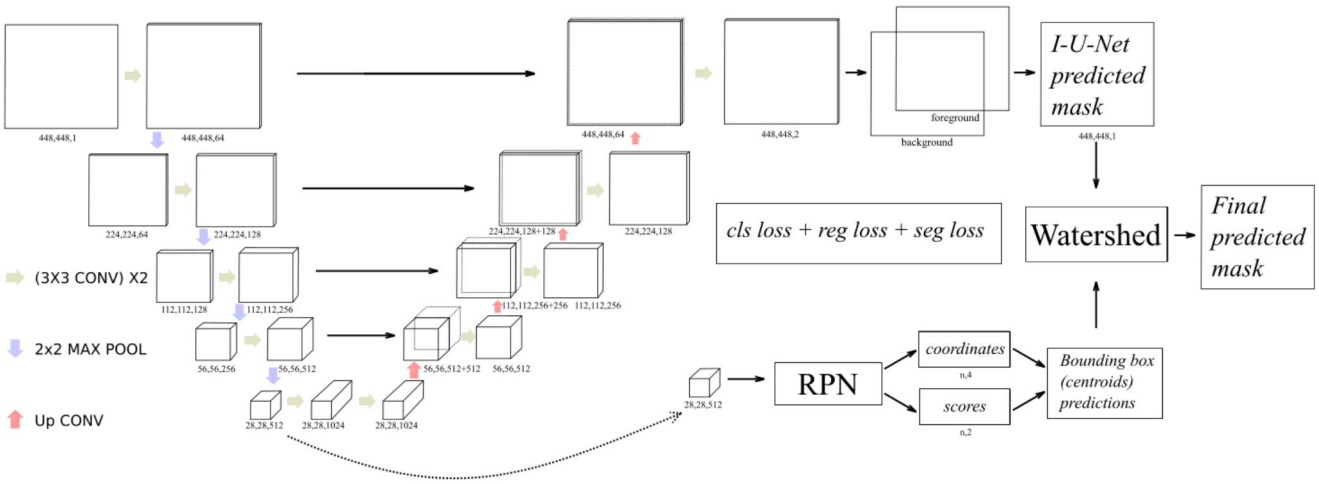

Figure 1: Network structure. The feature extraction network(which is also the down-sampling branch of Instance-U-Net) is composed of convolution layers and down-sample layers. The extracted feature map of shape $(28, 28, 512)$ is passed to two separate modules: 1.Region Proposal Network (RPN) and 2.Up-sampling branch of Instance-U-Net. RPN predicts $H * W * k$ anchors with the 4 coordinates $(x1, x2, y1, y2)$ and 2 object scores $(foreground, background)$ and generates the proposed regions after Non-max suppression (NMS). Up-sampling branch predicts segmentation masks. The RPN predicted cell centroids and Instance-U-Net predicted masks are then passed into the Watershed Layer to generate the final predicted masks. Different arrows denote different operations

based on baseline models. Among all of them, U-Net [9] is a powerful and elegant model as it yields accurate results with very few training samples (30 images). The incorporation of up-sampling operations significantly increases the resolution of final outputs. Another challenges in cell segmentation is to separate touching objects (cells). U-Net attempts to address that by introducing weight matrix to force the network to learn more on the touching pixels. However, the U-Net still often fails to segment densely touching cells as the cell boundaries are difficult to estimate in dense regions where cells have deformations, and the hyperparameters needed to compute the weight matrices may introduce segmentation errors. Most importantly, U-Net is purely the segmentation-based algorithm, it still treats each pixels nearly equally without seeing cells as objects.

2.3. Faster-R-CNN

To our knowledge, Faster-R-CNN and related models still obtain the highest accuracy in object detection. Compared with other object detection algorithms, the main contribution is the design of Region Proposal Networks (RPN). Regions of interests are proposed by placing translation-invariant anchors. Than each anchor is evaluated by RPN, which further predicts the probabilities of anchors being objects, and 4 coordination offsets. Then the model integrates Fast R-CNN [5], which uses RoIPool to extract features from bounding boxes. The features are than processed to further predict class probabilities and finer bounding boxes offsets. Since cell segmentation tasks contain only one object class, RPN is sufficient to evaluate the bounding boxes and object probabilities.

2.4. Mask-R-CNN

The emergence of idea *instance segmentation* addresses the issue of semantic segmentation: it first detects all the target instances (in this case, the touching cells), and performs segmentation within each instance. Recently, Mask-R-CNN implemented instance segmentation by fusing Region Proposal Networks (RPN) [6] with FCN. By applying FCN to segment the regions of interest proposed by RPN, Mask-R-CNN was able to perform instance segmentation with good results. Our model takes the advantage of instance segmentation by incorporating RPN object loss and bounding boxes regression loss to help the network perform object-based segmentation.

2.5. Watershed

Watershed is a classical unsupervised segmentation algorithm. The name Watershed is defined by the geological term*Watershed*, which takes a drop of water (marker) as the topographic peak. Together with the computed distance map, the Watershed algorithm simulates the process of flooding from the peak towards the steepest gradient descent [2] . In our model, we inserted a Watershed layer which takes the Instance-U-Net predicted binary masks and RPN-computed cell centroids to further compute Watershed

lines between touching cells.

3. Datasets and Features

The dataset is collected using ZEISS LSM 700 confocal microscope with 40X objective on MCF10A human breast cancer cell line. The cell nucleus was stained by DAPI and illuminated by laser with wavelength 405nm. The training set consists of 484 images with the corresponding ground-truth binary labels, and the validation and test sets contain 8 images with ground-truth labels each. The size of each image is 448X448, and each image contains about 100 cells. The ground truth labels are computed using thresholding and other segmentation methods; segmentation errors are manually corrected image-by-image. The ground truth coordinates for bounding boxes are calculated based on the ground truth labels. We apply data augmentation techniques to teach the network the desired invariance and robustness properties. Thus, in this training, we adopt random mirrors, scaling between 0.8 and 1.2, and rotation over 360 degrees for all training data to alleviate the overfitting problem. The training images are normalized by subtracting the mean value and divided by the standard deviation.

4. Methods

4.1. Feature extraction network and region proposal network

The original faster R-CNN paper [8] applied transfer learning and started from VGG-16 network. Since our task is detecting touching cells in the image, which is different than the aim of VGG models, we propose end-to-end training and build our own feature extractor: it starts from a $448X448$ grayscale cell image. Except for the first step, which increases the channel size from 1 to 64, all other steps pass the input tensor with $3X3$ convolution layer twice, followed by a $3X3$ convolution layer which also doubles the channel size. Then the tensor is max-pooled on both H and W with stride 2.

Following the same operation sequences, a feature map of shape $28X28X512$ is fed into RPN, where it applies 2 $1X1$ convolution layers with channel size $2k$ and $4k$ separately. Region proposals were generated by placing k anchors on every pixel of the feature map. In our practice $k = 9$ (3 scales$(0.5, 1, 2)$ times 3 aspect ratios$(0.5, 1, 2)$) could give optimal cell detection results. Then the bounding box coordinates $(x1, y1, x2, y2)$ of shape $(H*W*k, 4)$ and corresponding class scores $(P_{object}, P_{non-object})$ of shape $(H*W*k, 2)$ are compiled into final bounding box proposal layer after the Non-max Suppression.

The Loss function for RPN is denoted as:

$$L_(p_i, t_i) = \frac{1}{N_{cls}} \sum_i L_{cls}(p_i, p_i^*) + \frac{1}{N_{reg}} \sum_i p_i^* L_{reg}(t_i, t_i^*) \quad (1)$$

Briefly, i is the index of an anchor and p_i is the predicted probability of anchor i being an object. The ground-truth label p_i^* is 1 if the anchor is positive, and 0 if negative. t_i is a vector that represents the 4 parameterized coordinates of the predicted bounding box, and t_i is the ground-truth box associated with a positive anchor. The classification loss L_{cls} is log loss over two classes(object vs. non-object). For the regression loss, we use $L_{reg}(t_i, t_i) = R(t_i t_i)$ where R is smooth L1 function.

4.2. Segmentation by Instance-U-Net

Since RPN was originally designed to detect different objects in complex real-life environments, where each detected objects may touch each other. We reasoned that the accurate performance of RPN in detecting touching objects could be used to improve the cell segmentation performance. We bring the objectiveness insights into the original U-Net structure to build a so-called *Instance-U-Net*. To build such Network, we merged the feature-extraction network with the downsampling branch of U-Net. In this way, the U-Net downsampling branch will learn to extract optimal features that recognize each cell as **object** which is guided by L_{cls} and L_{reg}.

Besides being used by the RPN as mentioned above, the same feature map of shape (28,28,512) is shared with the Instance-U-Net, which contains another 4 conv-upsampling layers. The feature maps in the down-sampling path are concatenated with the corresponding up-sampling feature maps to preserve feature details. The final output tensor of Instance-U-Net is 2 feature map of the same shape as the input image, denoting the softmax scores of single pixels being background or foreground. Each pixel is classified based on the softmax score to produce the final mask prediction.

Two different segmentation loss formulas are implemented. The binary cross-entropy (BCE) loss function for segmentation is denoted as:

$$\sum w(x) log(p_{l(x)}(x)) \quad (2)$$

Where $w(x)$ is the weight map, $log(p_{l(x)}(x))$ is the pixel-wise cross-entropy sum over all pixels. The weight map $w(x)$ is computed for each ground truth segmentation mask to help the network learn the small separation borders between touching cells.

$$w(x) = w_c(x) + w_0 exp(-\frac{(d1(x) + d2(x))^2}{2\sigma^2}) \quad (3)$$

Where w_c is the weight map to balance the class frequencies, $d1(x)$ and $d2(x)$ denotes the distance to the border

of the nearest cell second nearest cell. In practice we set $w_0 = 10$ and $\sigma = 5$.

Since BCE is only a proxy indicator of the segmentation performance, whereas Dice coefficient is a more direct evaluation on the performance, we sought to test the Dice loss to further improve the segmentation results. The soft Dice loss is denoted as:

$$1 - 2\frac{\sum_i p_i(x) y_i(x)}{\sum_i p_i^2(x) + \sum_i y_i^2(x) + \epsilon} \tag{4}$$

Where $p_i(x)$ and $y_i(x)$ are the binary labels from predictions and ground truth labels. ϵ is a small factor to prevent division by zero, in practice we set $\epsilon = 10^{-6}$

4.3. Separating densely touching regions by Marker-based Watershed

Watershed algorithm was specifically designed to evaluate the boundaries of touching objects. Therefore we designed a built-in Tensorflow watershed layer to further improve the separation of touching cells by incorporating scikit-image's implementation[10]. As shown in Fig.2, the RPN predicted cell centroids and the Instance-U-Net computed masks are fed into Watershed layer predict the cell boundaries.

5. Experiments and Discussion

5.1. Implementation details

The models were developed using Tensorflow[1]. We adapted and incorporated [3] and luminoth faster-RCNN implementations in developing the Region Proposal Network. Experiments were performed on Tesla K80 with 12 GB Memory. Learning rate we chose was 10^{-4} to balance the converging speed and performance. We use Adam optimizer, since it provides the best converging speed and accuracy. For RPN, we use NMS threshold = 0.3 to prevent multiple bounding boxes on the same cell.

To evaluate the performance of different algorithms, we report pixel error, mean IU, RMSE, and instance error as metrics. Instance error is denoted as $\frac{\sum I_{error}}{\sum I_{all}}$, $\sum I_{error}$ is the number of instances that are not correctly separated, $\sum I_{all}$ is the total number of touching instances.

5.2. Qualitative evaluation

Representative results are shown in Fig.3 and Fig.4. From the results, we can see that the baseline U-Net model (U-Net with weighted binary cross entropy loss) has decent performance on regions where cells are touching but the total number of cells in cell clusters are small. However, U-Net still fails in many instances where regions are too dense and cell deformation happens. Instance-U-Net resolves most of the issue and consistently achieve better

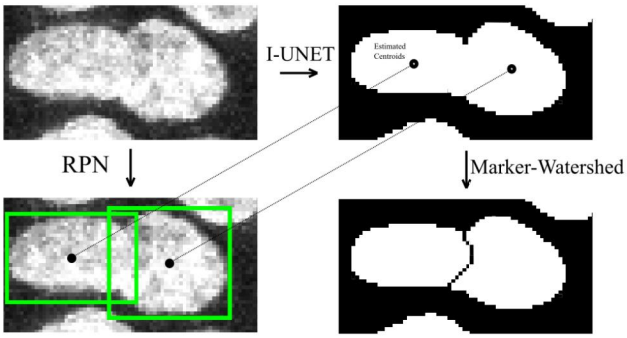

Figure 2: Improving segmentation results by implementing Watershed algorithm. Bounding boxes and segmentation masks are computed by RPN and Instance-U-Net. Then the estimated centroid of each cell is computed from the 4 coordinates as $(\frac{x_1+x_2}{2}, \frac{y_1+y_2}{2})$. The the Watershed line is estimated based on the binary mask and centroids.

results: most of the cells are separated and the boundaries are smoother. Since Instance-U-Net also learns to detect cells vs. non-cells dirt in the images, Instance-U-Net outperforms U-Net as it will not segment dirt/non-cell object as cells. Although has better shape estimations, the using of Dice loss fails to produce a comparable performance as weighted BCE, and it sometimes introduce artifacts at cell edges. The problem might be due to the unstable gradient flows introduced by Dice loss. Using the combination(sum) of BCE and Dice loss leads to the visually best results, as the evaluated cell boundaries are more accurate and cleaner.

Watershed algorithm further improves the Instance-U-Net in segmenting touching cell regions, where human could visually identify the number of cells in the region, but hard to evaluate the cell boundaries. Watershed algorithm helps the Instance-U-Net to further evaluate the boundaries of those regions. However, Watershed layer also introduces several errors in final segmentation (as shown in Fig 3 bottom, I-U-Net-Watershed-BCE results, red dash circle): if a single cell nucleus has been recognized as two objects by RPN, then 2 markers will be placed in the same cell, and force Watershed algorithm to generate a line in the middle of the cell.

5.3. Quantitative Evaluation

The comparison of metrics for different algorithms has been reported in Table.1. Instance-U-Net with BCE/Dice joint loss outperforms other models in all metrics except Instance error. The smallest instance error is achieved on the same model with the Watershed layer, which matches with our expectation: Watershed layer is designed to handle instances that are hard to segment for Instance-U-Net with BCE/Dice joint loss. The implementation of Watershed further decreases 2/3 of the instance error. Only a

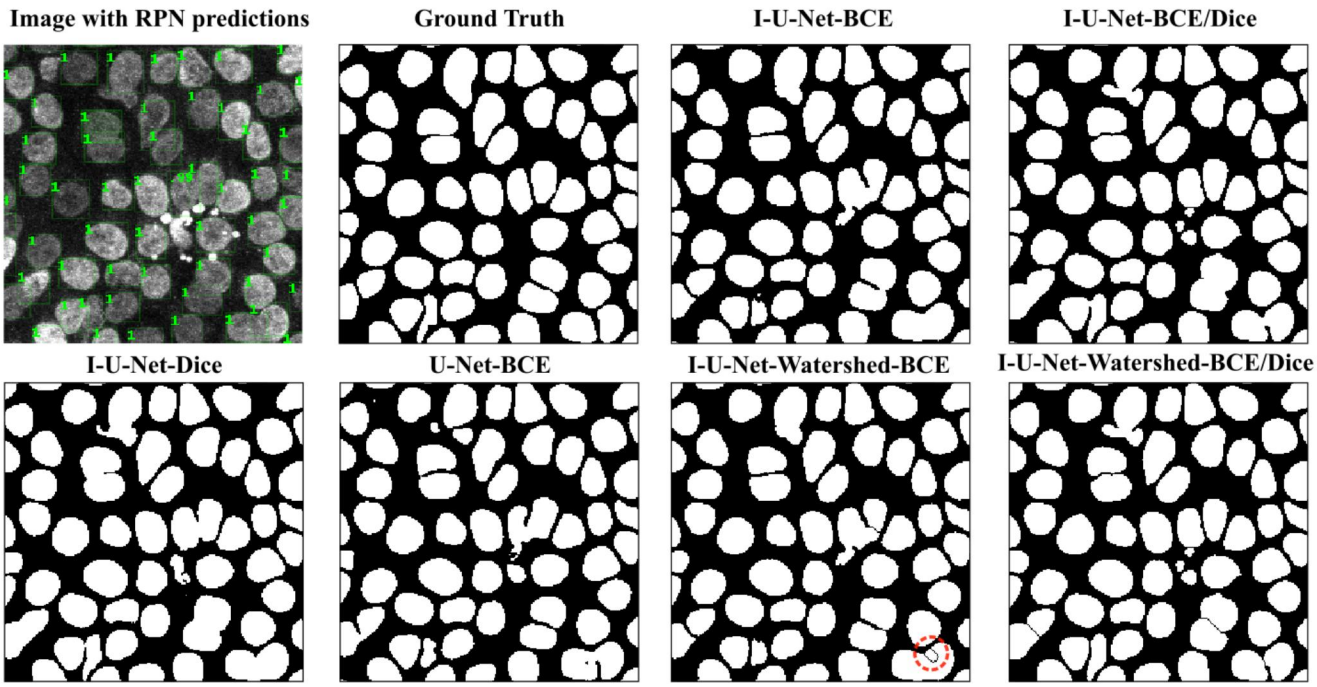

Figure 3: Sample results. For inference time, test images(top left without bounding boxes) are fed into different models and generate: 1. the object predictions(top left bounding boxes) with confidence scores and 2. the predicted masks by different models. Among all them, Instance-U-Net with joint BCE/Dice loss obtains the visually best results, it also has the best Watershed separation boundaries without introducing too many errors. Trained with BCE or Dice loss separately only achieved good shape or touching edge segmentations.

Table 1: Quantitative results associated with different approaches

	Pix. Err.	Mean IU	RMSE	Ins. Err.
IU-Dice	0.0301	0.9359	0.1735	0.3113
U-BCE	0.0292	0.9377	0.1710	0.1698
IU-BCE	0.0292	0.9378	0.1708	0.1132
IUW-BCE	0.0299	0.9356	0.1728	0.0660
IU-BCE/Dice	**0.0288**	**0.9385**	**0.1697**	0.1557
IUW-BCE/Dice	0.0297	0.9372	0.1724	**0.0566**

very small amount of touching cells remain connected after the Watershed transformation. The instance-U-Net (Table.1, row3) consistently outperform the baseline U-Net (Table.1, row2) in every metrics, which justifies the necessity of adding bounding box losses to help the network learn instances/objects. Fig.5 also shown that Instance-U-Net (blue line) converges much faster in mean IU than U-Net (red line). As previously mentioned and shown in Fig.3, directly training on Dice loss (Table.1, row1) fails to separate nearly 1/3 of all touching instances. The joint loss brings the smooth gradient flow from BCE and direct metrics training from Dice to achieve better performance. We did not observe any sign of over-fitting, as the training, validation and test performances are very similar.

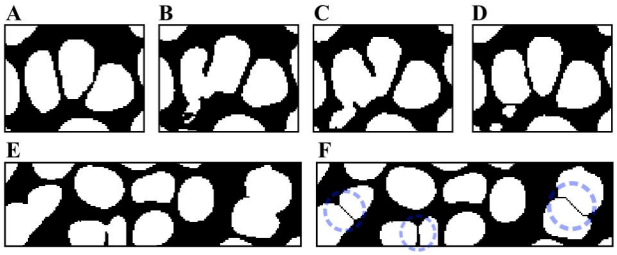

Figure 4: Zoomed-in analysis on the effect of instance training and Watershed algorithm in dense region. Fig 4 A-D: Ground truth, prediction from U-Net w/ BCE loss, prediction from Instance-U-Net w/ BCE loss and prediction from Instance-U-Net w/ BCE/Dice joint loss. Instance-U-Net outperforms U-Net by recognizing the cell shapes better (B and C), and joint training of BCE/Dice loss further improves the prediction. Fig 4 E,F: The prediction before Watershed layer(E) and after Watershed layer(F), blue dash circles indicates the boundaries estimated by Watershed.

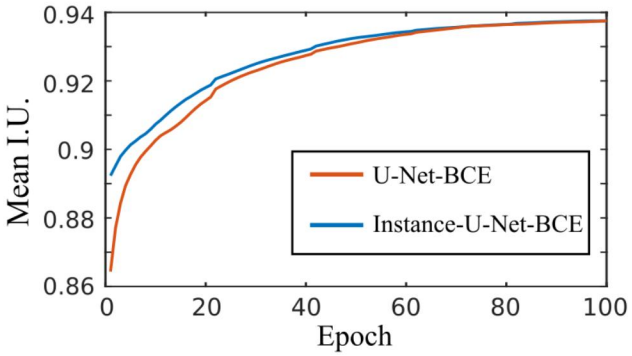

Figure 5: The effect of instance training on Mean I.U. on validation set. Compared with U-Net (red line), Instance-U-Net (blue line) achieved faster Mean I.U. convergence.

6. Conclusion and future work

The Instance-U-Net with joint loss achieves promising segmentation results in microscope images with densely touching cells. The implementation of Watershed at the end further improves the performance of separating touching cells by 3 folds. Our model only needs small training time (7 hours) with fast inference time (5fps) and could be easily applied to other datasets and tasks. Future work will be focused on improving single cell detection by engineering Region Proposal Network to generate less error introduced by Watershed.

7. Acknowledge

The authors would like to thank Prof. Jan Liphardt (Stanford University) for insightful input in building models, especially the help of forming the idea to implement the Watershed module to further improve the segmentation results. We would also like to thank J. Matthew Franklin, who kindly provides the dataset with training labels.

Furthermore, considering the medical privacy concern, source code is available from the corresponding author upon reasonable request. Meanwhile, we have uploaded the code on gradescope. In this project, two authors contributed equally to this work.

References

[1] M. Abadi, A. Agarwal, P. Barham, E. Brevdo, Z. Chen, C. Citro, G. S. Corrado, A. Davis, J. Dean, M. Devin, S. Ghemawat, I. Goodfellow, A. Harp, G. Irving, M. Isard, Y. Jia, R. Jozefowicz, L. Kaiser, M. Kudlur, J. Levenberg, D. Mané, R. Monga, S. Moore, D. Murray, C. Olah, M. Schuster, J. Shlens, B. Steiner, I. Sutskever, K. Talwar, P. Tucker, V. Vanhoucke, V. Vasudevan, F. Viégas, O. Vinyals, P. Warden, M. Wattenberg, M. Wicke, Y. Yu, and X. Zheng. TensorFlow: Large-scale machine learning on heterogeneous systems, 2015. Software available from tensorflow.org.

[2] S. Beucher. The watershed transformation applied to image segmentation. 1991.

[3] X. Chen and A. Gupta. An implementation of faster rcnn with study for region sampling. *arXiv preprint arXiv:1702.02138*, 2017.

[4] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei. Imagenet: A large-scale hierarchical image database. In *Computer Vision and Pattern Recognition, 2009. CVPR 2009. IEEE Conference on*, pages 248–255. Ieee, 2009.

[5] R. Girshick. Fast r-cnn. In *Proceedings of the IEEE international conference on computer vision*, pages 1440–1448, 2015.

[6] K. He, G. Gkioxari, P. Dollár, and R. Girshick. Mask r-cnn. In *Computer Vision (ICCV), 2017 IEEE International Conference on*, pages 2980–2988. IEEE, 2017.

[7] J. Long, E. Shelhamer, and T. Darrell. Fully convolutional networks for semantic segmentation. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 3431–3440, 2015.

[8] S. Ren, K. He, R. Girshick, and J. Sun. Faster r-cnn: Towards real-time object detection with region proposal networks. In *Advances in neural information processing systems*, pages 91–99, 2015.

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

[10] S. Van der Walt, J. L. Schönberger, J. Nunez-Iglesias, F. Boulogne, J. D. Warner, N. Yager, E. Gouillart, and T. Yu. scikit-image: image processing in python. *PeerJ*, 2:e453, 2014.

[11] C. You, Q. Yang, L. Gjesteby, G. Li, S. Ju, Z. Zhang, Z. Zhao, Y. Zhang, W. Cong, G. Wang, et al. Structurally-sensitive multi-scale deep neural network for low-dose ct denoising. *IEEE Access*, 6:41839–41855, 2018.

[12] C. You, Y. Zhang, X. Zhang, G. Li, S. Ju, Z. Zhao, Z. Zhang, W. Cong, P. K. Saha, and G. Wang. CT super-resolution gan constrained by the identical, residual, and cycle learning ensemble (gan-circle). *arXiv preprint arXiv:1808.04256*, 2018.