A Coarse-to-Fine Data Generation Method for 2D and 3D Cell Nucleus Segmentation

Zhuo Zhao
Department of Computer Science and Engineering
University of Notre Dame
Notre Dame, IN 46556, USA
zzhao3@nd.edu

Hongxiao Wang
Department of Computer Science and Engineering
University of Notre Dame
Notre Dame, IN 46556, USA
hwang21@nd.edu

Yizhe Zhang
Department of Computer Science and Engineering
University of Notre Dame
Notre Dame, IN 46556, USA
yzhang29@nd.edu

Hao Zheng
Department of Computer Science and Engineering
University of Notre Dame
Notre Dame, IN 46556, USA
hzheng3@nd.edu

Siyuan Zhang
Department of Biological Sciences, Harper Cancer Research Institute
University of Notre Dame
Notre Dame, IN 46556, USA
szhang8@nd.edu

Danny Z. Chen
Department of Computer Science and Engineering
University of Notre Dame
Notre Dame, IN 46556, USA
dchen@nd.edu

***Abstract*—Cell nucleus segmentation is a fundamental task in biomedical image analysis. Generating realistic cell nucleus data with ground truth masks can help tackle difficulties such as insufficient training data for deep learning models and the need to deal with "hard" cases (e.g., tightly clumped nuclei). Known nucleus generation methods generated individual nucleus masks from parametric models or based on direct transformations of real masks. It is difficult for these methods to capture and simulate the distributions of real nuclei and interactions among hard nuclei. In this paper, we propose a new three-stage coarse-to-fine nucleus generation method for 2D and 3D nucleus segmentation. The first stage simulates the positions and sizes of nuclei; the second stage simulates the shapes of nuclei and interactions among clumped nuclei; the third stage simulates the textures of nuclei. We evaluate our method on 2D and 3D cell nucleus image datasets. Experimental results show that our new nucleus generation method considerably helps improve cell nucleus segmentation performance and outperforms known nucleus generation methods for nucleus segmentation with a small amount of training data.**

***Index Terms*—Data Generation, Nuclei Segmentation, Augmentation, Deep Learning**

I. Introduction

Cell nucleus segmentation is a fundamental task in biomedical image analysis. Although recent deep learning methods (e.g., [1], [2]) achieved very good performance in instance segmentation of natural scene images, instance segmentation of biomedical images such as nucleus segmentation still faces big challenges. In common practice, instance segmentation needs high workload to annotate sufficient instances for deep learning model training. Since trained experts are often needed to annotate biomedical image data, annotating biomedical images can be highly costly and time-consuming, thus making it difficult to attain sufficient amounts of biomedical image training data for instance segmentation tasks. Besides the annotation difficulties, due to the need to deal with "hard" cases (e.g., tightly clumped nuclei), generation methods for generating realistic data with ground truth masks are frequently desired to improve the performance of deep learning based nucleus segmentation.

Common data generation methods for cell nucleus segmentation often have two steps: generate individual nucleus masks and generate corresponding raw images. But, the artificially generated masks may not have similar distributions as the real data. Some known methods [3]–[5] generated individual nucleus masks from parametric models, and different carefully designed parametric models may be needed for different datasets with different distributions and behaviors in order to achieve good performance. Other methods [6] generated nucleus masks by performing transformations directly on real masks. However, without considering the connected areas and interactions among tightly clumped nuclei, it is difficult for these methods to capture and simulate the behaviors of such real cases.

To synthesize realistic images for 2D and 3D cell nucleus segmentation tasks, we examine various key issues and relations of real nuclei in a progressive manner, and propose a new three-stage coarse-to-fine nucleus generation method (see Fig. 1): (1) we simulate the positions and sizes of nuclei by generating boxes similar to the box annotation for real

2372-9198/20/$31.00 ©2020 IEEE
DOI 10.1109/CBMS49503.2020.00016

nuclei, using a new light-weight box representation, which stabilizes the training process; (2) we delineate the shapes of the nuclei and simulate the interactions among clumped nuclei to generate nucleus masks using a new training strategy based on CycleGAN [7], which retains the semantic correspondence between the generated boxes and the masks while maintaining the diversity of the masks; (3) we simulate the textures of the nuclei based on the generated masks to generate realistic raw images. The synthesized images and corresponding masks can be used to train deep learning models for improving nucleus segmentation performance.

We evaluate our new method on cell nucleus datasets with 2D and 3D images. Experimental results show that even with a small amount of training data, our generation method considerably improves cell nucleus segmentation performance and outperforms known nucleus generation methods for cell nucleus segmentation.

II. Methods

Our new nucleus generation method consists of three main stages. In the 1st stage, it learns the positions and sizes of the real nuclei from some given nucleus distributions and box annotation for real nuclei; in the 2nd stage, it learns the shapes of the nuclei and the interactions among clumped nuclei based on the box annotation and mask annotation of the nuclei; in the 3rd stage, it learns to synthesize photo-realistic images based on the mask annotation and the corresponding raw images of the nuclei. We use CycleGAN [7] for the 1st and 2nd stages and pix2pix [9] for the 3rd stage. Below we present these three stages of our method in detail.

A. Simulating the Positions and Sizes of Nuclei

A representation of boxes of real nuclei can provide information on their positions and sizes. The 1st stage of our method learns to generate such boxes based on the given artificial nucleus distributions and the box annotation for real nuclei, using a specially designed light-weight box representation. Realistic boxes are generated even if the given nucleus distributions deviate from real nucleus distributions.

On one hand, to represent the approximate distributions of the nucleus centers, a set of "seed points" is artificially arranged [3] in the images as the "point domain" (e.g., Fig. 2(a)). On the other hand, we represent the positions and sizes (boxes) of the real instances by ovals/ellipsoids. This box representation forms the "box domain". We utilize CycleGAN to learn a mapping from the "point domain" to the "box domain". This model is expected to generate boxes with realistic sizes and positions for any given seed points. By putting different seed points in the point domain, we can manipulate the neighborhood environments (crowded or sparse, clustered or separated nuclei). Note that even the artificially arranged seeds do not always have realistic distributions; CycleGAN can automatically correct the distributions by relocating instances, adding new instances, or deleting instances (Fig. 2). Such "mismatched" cases caused by CycleGAN corrections will not harm the generation results, since only the generated raw images and the masks are required to be aligned (Section II-C). On the contrary, such corrections help generate fake data with similar distributions as the real data.

Our proposed light-weight box representation is designed to accommodate the nucleus generation task. The known instance detection or segmentation methods (e.g., [1], [2]) tended to use multi-layer box representations with predefined multi-scale anchors as regression references, and complicated post-processing (e.g., Non-Maximum-Suppression (NMS)) was used to generate the final boxes from the regressed feature maps. Such representations work well for supervised instance detection or segmentation tasks, since the losses between the regressed boxes and real boxes are considered explicitly. However, for data generation tasks, it is difficult for the deep learning models to learn these complicated representations, and even minor changes of the representations can lead to generating totally different boxes, implying that the generation is not stable. Different from the known methods, our box representation is much simpler, yet without losing information for the positions and sizes of the nuclei.

Fig. 2(b) shows an example of our proposed box feature representation in the 1st stage. Our box representation contains only one layer by using ovals or ellipsoids as boxes for 2D or 3D instances. This one-layer representation is simple for model learning; thus it can stabilize the training process compared to the multi-layer box representations. The size of an oval/ellipsoid for a nucleus is chosen to be half the size of that nucleus, and this allows us to separate different nuclei with simple post-processing (e.g., computing the connected components of instances). To help the model locate different instances, the center areas of the ovals/ellipsoids are reinforced by giving them higher intensities than their boundary areas.

The intensity I of an arbitrary pixel (x, y) inside an oval of the box representation is computed as:

$$I(x,y) = \frac{1 - \left(\frac{(x-xc)^2}{(w/4)^2} + \frac{(y-yc)^2}{(h/4)^2}\right) + bv}{(1+bv)} \quad (1)$$

where xc and yc represent the center of the original box, w and h represent the width and height of the original box, and bv is the value for adjusting the intensities on the boundary of the oval to help segment the oval from the background. In our experiments, we set bv = 0.25. Similar calculations for voxel intensities are used for 3D data (i.e., ellipsoids).

B. Simulating the Shapes of Nuclei and Interactions among Clumped Nuclei

In the 2nd stage, we develop a new training strategy to generate nucleus shapes "constrained" by the boxes generated in the 1st stage. Our method not only generates realistic nucleus shapes, but also simulates interactions among clumped nuclei. For instance-level image generation, we emphasize on clear boundaries among instances, using a "boundary" class and a k-terminal cut algorithm [10].

The nucleus shapes are represented by colored masks, with different colors representing different instances (e.g.,

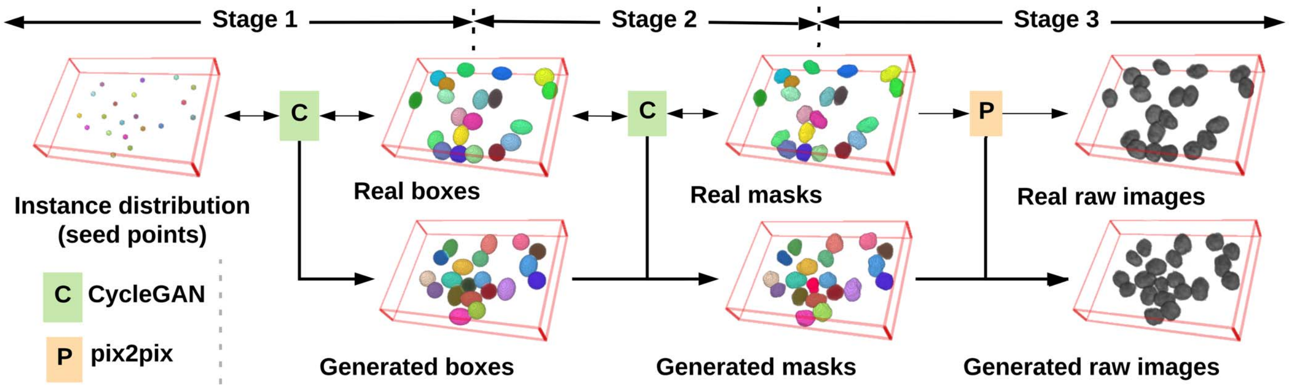

Fig. 1. Illustrating our workflow using images of 3D HL60 cell nuclei [8]. The top row shows the training data in each stage, and the bottom row shows the generated data in each stage. In the first stage, we generate boxes containing position and size information; in the second stage, we generate masks with realistic shapes and boundary behaviors; in the third stage, we generate realistic raw images.

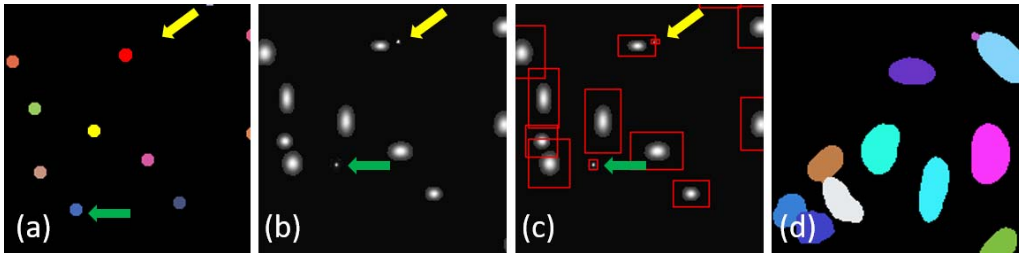

Fig. 2. Illustrating the generation of boxes and masks from a given nucleus distribution for nuclei of 2D U2OS cells [8]. (a) A given nucleus distribution represented by seed points; (b) the generated box representation (ovals) in the 1st stage for (a); (c) the processed boxes from (b); (d) some real masks (different instances marked by different colors). The generated boxes can be realistic even if the given nucleus distribution is not (e.g., a blue instance (marked by green arrows) shifts upward closer to a yellow one; a tiny point (marked by yellow arrows) is automatically generated near a red seed).

Fig. 2(d)). We denote the nucleus masks as the "mask domain". The CycleGAN framework can be used to map the box domain (defined in Section II-A) to the mask domain. Realistic nucleus shapes and interactions (e.g., boundary squeezing) among the clumped nuclei can be learned from real data. However, a difficulty to such a mapping is that CycleGAN is likely to cause mismatched cases, as discussed in Section II-A. Further, in our situation, the nucleus masks should have the same positions and size distributions as the boxes. Thus, we need to come up with a method for generating masks while maintaining the position and size information from the boxes.

We propose a new training strategy, combining the advantages of CycleGAN and pix2pix. In each training iteration, we select a mask m_i from the real data and represent it by a colored mask S_i in the mask domain, and for the box domain, we choose the corresponding oval/ellipsoid B_i. A same framework as CycleGAN is adopted, and we map the box domain to the mask domain with $F(\cdot)$ (mapping B_i in the box domain to $F(B_i)$ in the mask domain). Different from the original CycleGAN training, we do not feed arbitrary (B_i, S_j) pairs, but use the corresponding pairs (B_i, S_i). Such design could minimize mismatched cases, since the model would learn to generate masks "constrained" by the given boxes. Compared to pix2pix, our method only enforces $F(B_i)$ to fit the distribution of $\{S_i|\forall i\}$, instead of enforcing exactly $\{S_i = F(B_i)|\forall i\}$. This design is based on a consideration that one B_i could match with multiple possible S_i, i.e., boxes of the same size may likely generate different masks, so long as the generated masks look realistic. But, such a one-to-many mapping may potentially make the pix2pix model collapse and output boundaries with artifacts.

To explicitly utilize the box sizes, we let the ovals/ellipsoids representing the boxes in this stage have the same sizes as their corresponding boxes. Different ovals/ellipsoids are marked with different colors. Fig. 3(a) shows an example of such ovals representing the boxes in this stage. Our new training strategy is used to fine-tune the ovals/ellipsoids into realistic masks (e.g., Fig. 3(b)).

Note that generating masks from boxes faces a challenge of dividing the connected masks into individual ones, since the colors in a certain generated mask may not be unique. Instead of processing such connected masks directly out of CycleGAN, we adapt an additional "boundary" class to help CycleGAN divide the connected masks. Hence, by looking at the neighborhood of the masks, CycleGAN can learn to generate masks with similar shapes and interactions among clumped nuclei as the real data, and by looking at the masks and the corresponding boundaries, CycleGAN can divide the connected masks into individual ones with the help of the boundary class (Fig. 3(c)).

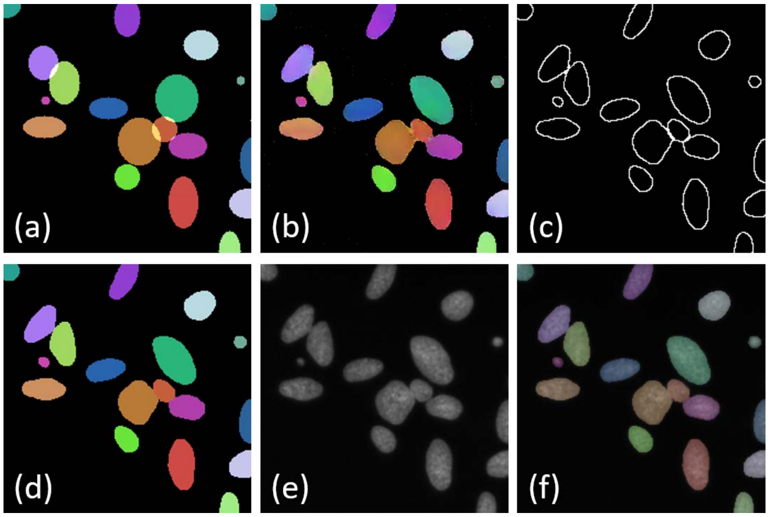

Fig. 3. Examples of generating masks from boxes and generating raw images from masks for nuclei of 2D U2OS cells [8]. (a) An example of the generated boxes in Section II-A, represented by colored ovals; (b)-(c) the generated raw masks and the corresponding boundaries based on (a); (d) the processed generated masks; (e) a generated raw image based on (d); (f) the generated masks match the generated raw instances exactly.

For those "difficult" instances that cannot be separated by the boundary class, we apply the iterative k-terminal cut algorithm [10] to separate them; we can use the centers of the boxes to indicate the presence of the "terminals", as in [10]. In the generated masks (Fig. 3(b)), the points belonging to the same mask tend to have similar colors. Thus, we compute the weights of the graph edges for the iterative k-terminal cut algorithm based on the color differences. Fig. 3(d) shows an example of the generated masks thus divided. The sizes and positions of the generated masks correspond to the initial boxes (Fig. 3(a)), while the shapes and interactions among the clumped nuclei are learned by our proposed training strategy.

C. Simulating the Textures of Nuclei

In the 3rd stage, we simulate the textures of the real nuclei. At the end of this stage, the generated raw images along with the generated masks from the 2nd stage are used to train deep learning models for cell nucleus segmentation.

Although one may use CycleGAN to generate raw images from masks [4], without being able to consider losses on the generated raw images directly but only considering the cycle consistency losses, there can be mismatched or distorted cases between the mask instances and the generated raw instances (see Fig. 4). However, training deep learning models for instance segmentation needs the boundaries of the masks aligned with the boundaries of the raw instances exactly. We apply pix2pix, and use the corresponding real masks and real raw images as training data to learn a mapping between mask instances and raw instances.

Taking advantage of the direct supervision on the generated raw images, the boundaries in the generated raw images (Fig. 3(e)) and the boundaries of the divided generated masks (Fig. 3(d)) are aligned exactly (Fig. 3(f)).

III. Experiments and Results

We evaluate our new data generation method for cell nucleus segmentation using two cell nucleus image datasets [8]. (1) 2D nuclei of U2OS cells, with 2038 instances for training and 14459 instances for testing. (2) 3D nuclei of HL60 cells, divided into a set with high signal-to-noise ratio (SNR) and a set with low SNR; for both these 3D sets with high SNR and low SNR, the training data contain 240 instances and the testing data contain 2160 instances. Note that to fully examine the performance of different generation methods for nucleus segmentation and simulate the situations of insufficient training data, we only use a small amount of data for training.

Realistic raw images along with the corresponding masks are generated for the above datasets using our method. The generators in our experiments adapt the encoder-decoder architecture with skip connections [11], and $Tanh$ is used as the activation function. The discriminators employ the design of the Markovian discriminator (patchGAN) [9]. Mean-Absolute-Error (MAE) loss is used for all the stages of our method.

The generated raw images and the corresponding masks, together with the real ones which are used by our generation method, are used to train several commonly-used deep learning segmentation models (U-Net [11] and Mask R-CNN [1] for 2D images, and 3D U-Net [12] and VoxResNet [13] for 3D images).

Tables I, II, and III show the performance of these models on the 2D dataset and 3D datasets, respectively. There are three comparative baselines: the experiments without data generation (none), the experiments with data generation generated by polygons from parametric models and CycleGAN (p + CycleGAN [4]), and the experiments with data generation generated by real masks and pix2pix (r + p2p [6]). We use the same real training data for all the baselines and our method,

TABLE I
RESULTS ON THE NUCLEI OF 2D U2OS CELLS.

Model	Aug Method	Pixel Seg	Instance Seg
U-Net	none	0.9605	0.9104
	p+CycleGAN	0.9601	0.9033
	r+p2p	0.9603	0.9272
	ours	0.9604	0.9386
Mask R-CNN	none	0.9203	0.9020
	p+CycleGAN	0.9150	0.9011
	r+p2p	0.9179	0.9088
	ours	0.9216	0.9182

TABLE II
RESULTS ON THE NUCLEI OF 3D HL60 CELLS WITH HIGH SNR.

Model	Aug Method	Voxel Seg	Instance Seg
3D U-Net	none	0.9697	0.9460
	p+CycleGAN	0.9665	0.9423
	r+p2p	0.9690	0.9520
	ours	0.9706	0.9625
VoxelResNet	none	0.9700	0.9441
	p+CycleGAN	0.9598	0.9268
	r+p2p	0.9698	0.9528
	ours	0.9705	0.9635

and basic data augmentation operations such as cropping, flipping, and rotation are applied to all the experiments.

For efficient evaluation of our generation method, we use two evaluation measures: pixel/voxel-level segmentation and instance-level segmentation. Both the pixel/voxel-level errors and instance-level errors are considered, and F1 scores are reported. For pixel/voxel-level segmentation, detected pixels/voxels are true positives if they correspond to some ground truth pixels/voxels. In instance-level segmentation, only the true positive pixels/voxels inside the correct detected instances (intersection over union (IOU) with a ground truth instance is ≥ 0.5) are taken as true positives.

From the experimental results, we do not observe consistent meaningful performance improvement or degradation on pixel/voxel-level segmentation (we still show these results for comparison purpose). But on instance-level segmentation, our method helps improve considerably cell nucleus segmentation performance (1.27% to $+2.82\%$) compared to the same models without data generation. Further, our method outperforms the same models using other data generation schemes. Note that generation data generated by polygons and CycleGAN (p+CycleGAN) can even harm the performance due to mismatched cases as we discussed in Section II-C. Fig. 4 shows some examples of the mismatched cases.

The experimental results show that our data generation method can generate realistic raw cell nucleus images that capture interactions among different real nuclei. With the assistance of the generated data, our method helps considerably improve the instance-level segmentation performance compared to the commonly used powerful segmentation models without any data generation, and outperforms other data generation methods for cell nucleus segmentation in both 2D and 3D images with a small amount of training data available.

TABLE III
RESULTS ON THE NUCLEI OF 3D HL60 CELLS WITH LOW SNR.

Model	Aug Method	Voxel Seg	Instance Seg
3D U-Net	none	0.9708	0.9468
	p+CycleGAN	0.9655	0.9512
	r+p2p	0.9700	0.9501
	ours	0.9718	0.9595
VoxelResNet	none	0.9691	0.9335
	p+CycleGAN	0.9645	0.9456
	r+p2p	0.9687	0.9505
	ours	0.9694	0.9588

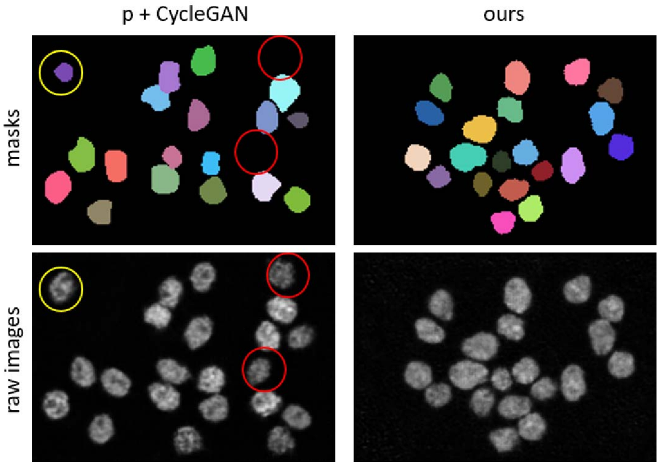

Fig. 4. Some examples of synthesized masks and the corresponding raw images for nuclei of 3D HL60 cells with high SNR [8] using the method in [4] and our method. For ease of observation, we only show 2D slices from 3D stacks. The left column shows the generated results for the method in [4], and the right column shows our generated results. In the left column, the red circles indicate that using the method in [4], nuclei in raw images may not have corresponding masks, and the yellow circles indicate that the boundaries of the generated nuclei may not be aligned well with the original masks. In comparison, in our results (the right column), the boundaries of the generated masks and the raw images are nicely aligned.

IV. CONCLUSIONS

In this paper, we proposed a new three-stage coarse-to-fine data generation method for cell nucleus segmentation. Our method can generate realistic images of cell nuclei and the corresponding masks, exploring interactions among clumped nuclei. Experimental results show that our method helps improve considerably the instance-level segmentation performance compared to the commonly used powerful segmentation models without any data generation, and outperforms other data generation methods for cell nucleus segmentation in 2D and 3D images with a small amount of training data. Our method can help reduce manual annotation effort on medical images.

ACKNOWLEDGMENT

This research was supported in part by NIH R01 CA222405 - 01A1 and NSF Grants CNS-1629914 and CCF-1617735.

REFERENCES

[1] K. He, G. Gkioxari, P. Dollár, and R. Girshick, “Mask R-CNN,” in *Proceedings of the IEEE international conference on computer vision*, pp. 2961–2969, 2017.

[2] R. Hu, P. Dollár, K. He, T. Darrell, and R. Girshick, “Learning to segment every thing,” *arXiv preprint arXiv:1711.10370*, 2017.

[3] A. Lehmussola, P. Ruusuvuori, J. Selinummi, H. Huttunen, and O. Yli-Harja, “Computational framework for simulating fluorescence microscope images with cell populations,” *IEEE transactions on medical imaging*, vol. 26, no. 7, pp. 1010–1016, 2007.

[4] F. Mahmood, D. Borders, R. Chen, G. N. McKay, K. J. Salimian, A. Baras, and N. J. Durr, “Deep adversarial training for multi-organ nuclei segmentation in histopathology images,” *IEEE transactions on medical imaging*, 2019.

[5] L. Hou, A. Agarwal, D. Samaras, T. M. Kurc, R. R. Gupta, and J. H. Saltz, “Robust histopathology image analysis: To label or to synthesize?,” in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 8533–8542, 2019.

[6] O. Bailo, D. Ham, and Y. Min Shin, “Red blood cell image generation for data augmentation using conditional generative adversarial networks,” in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops*, pp. 0–0, 2019.

[7] J.-Y. Zhu, T. Park, P. Isola, and A. A. Efros, “Unpaired image-to-image translation using cycle-consistent adversarial networks,” in *Proceedings of the IEEE international conference on computer vision*, pp. 2223–2232, 2017.

[8] V. Ljosa, K. L. Sokolnicki, and A. E. Carpenter, “Annotated high-throughput microscopy image sets for validation,” *Nature Methods*, vol. 9, no. 7, pp. 637–637, 2012.

[9] P. Isola, J.-Y. Zhu, T. Zhou, and A. A. Efros, “Image-to-image translation with conditional adversarial networks,” in *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 1125–1134, 2017.

[10] L. Yang, Y. Zhang, I. H. Guldner, S. Zhang, and D. Z. Chen, “3d segmentation of glial cells using fully convolutional networks and k-terminal cut,” in *International Conference on Medical Image Computing and Computer-Assisted Intervention*, pp. 658–666, Springer, 2016.

[11] 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*, pp. 234–241, 2015.

[12] Ö. Çiçek, A. Abdulkadir, S. S. Lienkamp, T. Brox, and O. Ronneberger, “3D U-Net: Learning dense volumetric segmentation from sparse annotation,” in *International Conference on Medical Image Computing and Computer-Assisted Intervention*, pp. 424–432, 2016.

[13] H. Chen, Q. Dou, L. Yu, and P.-A. Heng, “VoxResNet: Deep voxelwise residual networks for volumetric brain segmentation,” *arXiv preprint arXiv:1608.05895*, 2016.