Optimize and Speed up U-Net Convolutional Network for Generalized Nuclear Segmentation for Computational Pathology

Bui Kim Dung, Quan Yu, Tan Xinli Steven, Meng Yang and Lwi Tiong Chai

***Abstract*—Nuclear Segmentation is the extraction of the cell boundary locations of the cells in the images taken of the tissue samples, a key step in extracting biomedical data from tissue samples. There are many challenges to extracting the cell boundary locations from the images due to the complex, crowded images and similarities in the appearance between each cell. We used the data set from the MoNuSeg Nuclear Segmentation challenge, which was taken from multiple hospitals and includes a diversity of nuclear appearances from several patients, disease states, and organs, techniques trained on it are likely to generalize well and work right out-of-the-box on other H&E-stained images. This paper summarizes the related work related to deep learning-based nuclear segmentation, then based on the state-of-the-art models, it proposes a customized architecture of U-net Convolutional Network to optimize and increase the accuracy while speeding up the nuclear segmentation model training process.**

***Index Terms*—Annotation, boundaries, dataset, deep learning, U-Net, convolutional neural network, nuclear segmentation, nuclei.**

I. Introduction

This paper aims to improve the process of nuclear segmentation of the cells for the MoNuSeg Nuclear Segmentation Challenge. The MoNuSeg is an official satellite event of MICCAR 2018, the 21st International Conference on Medical Image Computing and Computer Assisted Intervention. Nuclear segmentation is the process of identifying the boundaries of the cells in the digital image accurately and precisely by partitioning a digital image of the cells into multiple segments

Cell samples are taken from patients and stained with Haematoxylin and eosin (H&E). Haematoxylin renders nuclear dark bluish purple and epithelium light purple, while eosin renders stroma pink. This makes it easier for people and computers to identify the cell boundaries and nuclei in the cells. Studying the size, density, shape and other features of the cells, medical conditions such as cancer or sclerosis can be diagnosed and appropriate treatments can be recommended. Furthermore, studying the shape and size of the cells and nuclei can help give a better understanding of the human body. Nuclear Segmentation can also be applied to cell samples taken from plants and animals. In summary, the quantitative assessment of the size and shape of the cells has many significant research and industrial applications in the biomedical field.

II. Data Sets

A. Background

The training images were taken from the MoNuSEg Nuclear Segmentation challenge 2018, (https://monuseg.grand-challenge.org).

The dataset consists of tissue samples taken from several patients with tumours of different organs. The tissue samples were stained with H&E and captured at 40x magnification. As a result, these tissue samples have very different shapes and sizes, which should result in a more generalised nuclear segmentation algorithm.

The dataset also includes the manually annotated nuclear boundaries of the cells in the images, which can be used to assess the accuracy of the Nuclear Segmentation algorithm. These were included as xml files.

B. Data Preprocessing

The binary mask images of the nuclear boundaries were pre-processed and two-class mask images with the pixels inside the nuclear boundaries were filled using the OpenCV-Python algorithm. First, morphological dilation and closing were applied using cv2.dilate and cv2.MORPH_CLOSE functions consecutively. After the transformation of the nuclear boundaries into contours without gaps, function binary_fill_holes from Scipy was used to fill the pixels inside the nuclear boundaries, resulting in usable mask images with pixels inside the nuclei labeled. The result of dilation + closing and binary filling holes is shown in Fig 1.

The original dataset consisted of 30 images of 1000 x 1000 resolution. To increase the training data and reduce the computational complexity, the images were split into 64x64 pixels images with stride of 30 pixels to create the new training

September 2018. The authors are with National University of Singapore, Institute of System Sciences, 119615 Singapore (emails: kimdung.bui@u.nus.edu, e0146446@u.nus.edu, yang_meng@u.nus.edu, e0146862@u.nus.edu, e0146443@u.nus.edu).

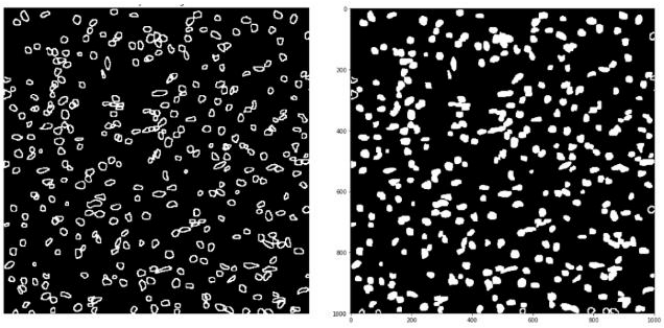

Fig. 1. Two-class mask image generation

sets with 7680 sample images. The 64x64 image size corresponds to the size of the largest nuclei, while the stride of 30 pixels create overlapped edges among the samples to prevent data loss as well as to produce a training set with higher resolution.

Another setting for this cropping process is to produce the cropped images of size 128x128 (total 1920 images) for testing the lightweight version of the proposed model.

III. RELATED WORK: DEEP LEARNING-BASED NUCLEAR SEGMENTATION

There are countless varieties of Convolutional Neural Networks (CNN) that have been used for image classification, and many of them have high accuracy and precision. [1]. For bio-medical images segmentation, the most popular technique is to apply a CNN on high resolution images to produce probability maps for detecting various tissue segments based on processing small patches of fixed sizes that have been sampled from the large original images.

A. Two-class CNN

The two-class CNN model tries to classify whether the central pixel of a patch belongs to a nucleus or not [2]. The result when applying this model to all possible overlapping patches of a fixed size is a binary or nuclear probability map. To resolve touching nuclei, it has been proposed that distance transform be computed on the binary map, which is likely to yield two separate local minimas near the centers of two touching nuclei due to the concavity in the shape of multiple nuclei. M. E. Plissiti, C. Nikou [3] and M. Veta, et al [4] in their experiments on automatic nuclei segmentation have also used shape concavity to separate touching nuclei. This technique works well for simple cases of two touching nuclei but not for more complex cases [5].

B. Three-Class CNN Emphasizing Nuclear Boundaries

Neeraj Kumar et al. [5] proposed the Three-Class CNN (CNN3). By explicitly introducing the nuclear *boundary* as a third class of pixels in addition to the usual binary of foreground (*inside* any nucleus) and background (*outside* every nucleus), this CNN produces a ternary probability map, which emphasizes detection of nuclear boundary and solves the complex cases of touching nuclei. The architecture of the CNN3 is shown in Table I.

The strategy in Neeraj Kumar et al. [5] has two drawbacks. First, it is quite slow because to estimate the 3-class probability assignment on each pixel, each 3-output node CNN took a patch of size 51 x 51 centered at that pixel as input. The network must be run separately on each patch (total 171,000 patches), and there is a lot of redundancy due to overlapping patches.

Secondly, the trade-off between localization accuracy and the use of context prevents the model from achieving better results. The size 51 x 51 was chosen as the smallest size to cover most of the large nuclei and to reduce computation complexity and training time. Patch-wise training is common [6] but lacks the efficiently of fully convolutional training.

TABLE I
CNN3 ARCHITECTURE FOR SEGMENTING NUCLEI

Layer	Filter size	Activation	Output size	Dropout rate
Input	-		51 x 51x 3	
Conv 1	4 x 4	ReLU	48 x 48 x 25	0.1
Pool 1	2 x 2	Max	24 x 24 x 25	-
Conv 2	5 x 5	ReLU	20 x 20 x 50	0.2
Pool 2	2 x 2	Max	10 x 10 x 50	-
Conv 3	6 x 6	ReLU	5 x 5 x 80	0.25
Pool 3	2 x 2	Max	3 x 3 x 80	-
FC 1	-	ReLU	1024	0.5
FC2	-	ReLU	1024	0.5
Output	-	SoftMax	3	

This architecture gave 92% accuracy for binary pixel classification and 0.76 average Dice's coefficient, compared to [8] which gave only 78% accuracy, CNN3 set the new benchmarking for generalized nuclear segmentation techniques.

C. Fully Convolutional Networks

Adapted contemporary classification networks (AlexNet [18], the VGG net [19], and GoogLeNet [20]) into *fully convolutional networks (FCN) [6]*, a group of authors from UC Berkeley has achieved state-of-the-art segmentation of PASCAL VOC. The model has been the first work to train FCNs end-to-end for pixelwise prediction and from supervised pre-training.

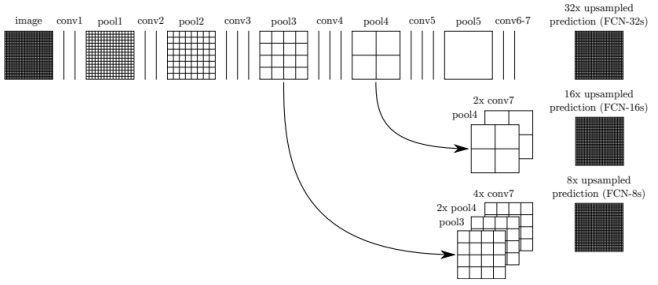

Fig. 2. FCN skip architecture which learns to combine coarse, high layer information with fine, low layer information. Pooling and prediction layers are shown as grids that reveal relative spatial coarseness, while intermediate layers are shown as vertical lines. [6]

By fine-tuning to the segmentation task, the model transfers the learned representations from the classical classification networks to dense prediction.

Semantic segmentation faces an inherent tension between semantics and locations: global information resolves what while local information resolves where [6]. Deep features hierarchies encode location and semantics in a nonlinear local-to-global pyramid. The skip architecture introduced in this paper as shown in Fig 2. that combines semantic information from a deep, coarse layer with appearance information from a shallow, fine layer to produce accurate and detailed segmentations.

However, although proven to achieve state-of-the-art segmentation accuracy, the deep architecture of these fully convolutional networks requires large amount of training input to produce good accuracy and generalize well on unseen samples, while most of the bio-medical segmentation problems have very limited number of sample images [5].

D. U-Net Convolutional Network

In 2015, Olaf Ronneberger, Philipp Fischer, and Thomas Brox introduced U-Net Convolutional Network [7] on the ISBI

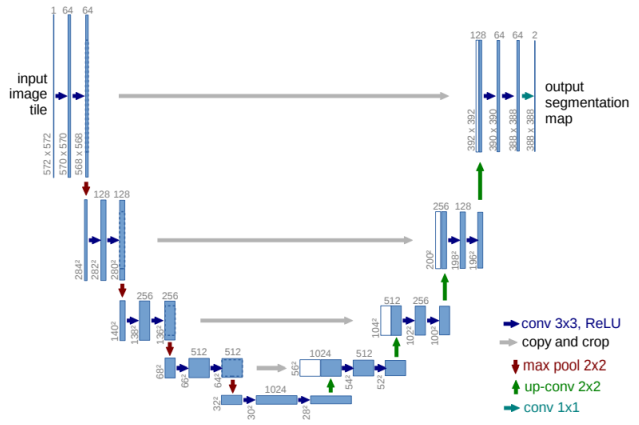

Fig. 3. U-net architecture. Each blue box corresponds to a multi-channel feature map. The number of channels is denoted on top of the box. The x-y-size is provided at the lower left edge of the box. White boxes represent copied feature maps. The arrows denote the different operations. [7]

challenge for segmentation of neuronal structures in electron microscopic stacks.

U-net has been built upon the fully convolutional network [6] architecture, with modification to adapt it to very few training images, while still yield precise segmentations. The architecture of U-net is depicted in Fig 3.

The important modification in U-net architecture is that in the up-sampling part, it still maintains large number of feature channels, which allow the network to propagate context information to higher resolution layers. The expansive path is symmetric to the contracting path and yields a u-shaped architecture.

The original U-net model produces total more than 31 million parameters, using the input images of size 512x512 pixels, training of 10 hours on a Nvidia Titan GPU (6GB).

IV. PROPOSED MODEL: OPTIMIZE AND SPEED UP U-NET FOR GENERALIZED NUCLEAR SEGMENTATION

This paper introduces a customized architecture of U-net which is more light-weight and faster-training, optimized for the MoNuSeg Nuclear dataset. It was able to achieve 0.96 average Dice's coefficient (compared to the CNN3 which achieved 0.76) and 97.8% accuracy for binary pixel classification (CNN3 achieved 92%) after only 30 training epochs which took 15 minutes (compared to the 10 hours for training original U-net model) on Nvidia GTX 1070 (8GB).

A. Network Architecture

The lightweight version of the proposed customized U-net architecture is depicted in Fig 4., which was train with 1920 images of size 128x128.

Similar to the original U-net, it consists of a contracting path (left side) and an expansive path (right side). The contracting path follows the typical architecture of convolutional network, which consists of the repeated application of two 3x3 convolutions, each followed by a rectified linear unit (ReLU) and a 2x2 max pooling operation with stride 2 for down-sampling. At each down-sampling step, the number of feature channels is doubled. Since the preprocessed images are small patches of 64x64 (or 128x128) cropped from original 1000x1000 pixels images [II.B], all the convolutional layers are using same padding to reserve the edge of the picture, which reserve the context for the edge pixels.

In the expansive path, while the original model using up-sampling layers for deconvolution, the customized model uses transpose convolution with trainable kernel and propagate the loss to later layers. This modification contributes to the increasement in Dice's coefficient and accuracy results.

Due to the loss of border pixels in successive convolution layers, the original model has to add a cropping step for up-sampling layers, but with the inherent edge reservation by same padding and overlapped cropping, the customized model simply concatenates the transposed map with the successive activation output through skipping path. This modification contributes to the reduce of training time and computation complexity.

B. Metrics

1) Loss function

This architecture does not support the three-class pixels as in CNN3. To address the challenge of separation of nuclei touching borders, the model makes use of weighted loss, where the separating background labels between touching cells is used to obtain a large weight in the loss function.

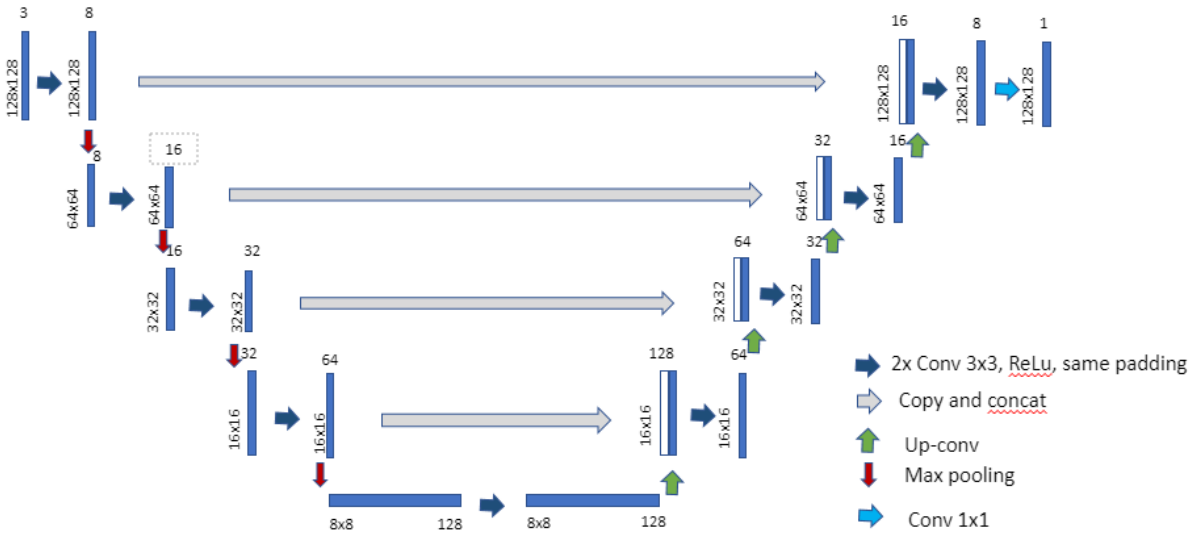

Fig. 4. Customized U-net architecture. Each blue box corresponds to a multi-channel feature map. The number of channels is denoted on top of the box. The x-y-size is provided at the lower left edge of the box. White boxes represent copied feature maps. The arrows denote the different operations. The setting of channel size is the setting of the lightweight version. The full-size version settings are described at Table II.

The binary cross entropy loss is calculated as:

$$-(ylog(p) + (1-y)\log(1-p) \quad (1)$$

In which, y is the binary indicator (0 or 1) and p is the predicted probability observation *o* is of class *C*.

2) Evaluation Metric

A commonly used object detection metric is the F1-score. For ground true objects G_i indexed by i and segmented objects S_j indexed by j, the F1-score is based on true positives TP (count of all ground truth objects G_i with an associated segmented (detected) object S_j, false positive FP (count of all segmented objects S_j without a corresponding ground truth object G_i), and false negatives FN (count of all ground truth objects G_i without a corresponding detected object S_j). F1-score is defined as follows:

$$F1 = \frac{2TP}{2TP + FP + FN} \quad (2)$$

A major shortcoming of F1-score using any object association criteria is that it does not take pixel-level (segmentation) errors into account. Thus, for this model, beside accuracy metric, it used Average Dice's coefficient metric to report shape concordance (pixel-level errors) between the ground truth objects and their associated segmented objects.

$$Dice = \frac{2|G_i \cap S_j|}{|G_i| + |S_j|} \quad (3)$$

Another reason the Dice's coefficient metric was chosen was to make it easier to compare with the result from the baseline model CNN3 [5].

C. Training and testing

The customized model can be trained on end-to-end by backward propagation and stochastic gradient descent (SGD). The total sample set was split into training set: evaluation set: test set with ratio 0.7:0.1:0.2 with random shuffling, using Adam optimizer and batch size of 32 samples.

Nuclei were detected by thresholding the class probability map at 0.5.

V. EXPERIMENTAL RESULTS

To estimate the customized architecture, two models have been built and tested with the two different ways of cropping sample images.

A. Light-weight customized model

The light-weight model architecture as depicted in Fig 4. has been built for testing on local machine (without GPU) for the training set of 1920 images of size 128x128. This lightweight model also uses same padding across all convolutional layers and the up-convolutional part was built with transpose2D layers.

The model is implemented using Keras [21] with TensorFlow [22] backend, the compiled model contains 485,817 trainable parameters, trained on laptop Macbook Pro (quad-core CPU, no GPU) over 50 epochs with batch size of 4, took 40s per epoch, which was roughly 33 minutes. On the GTX 1070 (8GB) GPU, it took only 1 minute. Exported model is only 5.8Mb.

This model achieved Dice coefficient of 0.8606 (while CNN3 is at 0.7623) and 92% accuracy. Sample results are given in Fig 5.

This lightweight model although is much more light-weight and took significantly less compute power and training time, already performed better than the baseline model CNN3 [5].

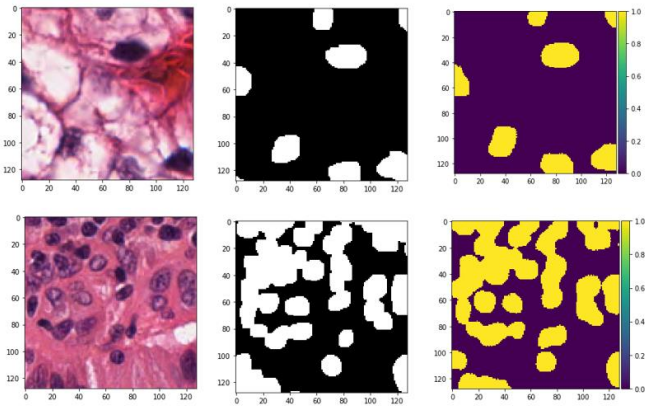

Fig. 5. Sample test results from the experimental lightweight customized U-net. The pictures on the left most column are original cropped images from the dataset. The middle column contains corresponding ground truth annotated nuclei. The right most column contains the predicted probability map segmented using the lightweight model.

B. Full size customized U-net model

The full size customized U-net model has been built to compare the training complexity with the original U-net model. Details configuration of this model are described in table II.

The input size for this model is 64x64x3, trained with batch size of 32 samples over total 7680 samples. With total 31,031,745 parameters, this model took 15 minutes over 30 epochs on Nvidia GTX 1070 (8GB).

It was able to achieve 0.96 average Dice's coefficient (while the CNN3 achieved 0.76) and 97.8% accuracy for binary pixel classification (CNN3 achieved 92%).

The sample test images are shown at Fig 6.

C. Original U-net model

To compare the customized model with the original model, an experiment with the original U-net model implemented based on [7] with same software framework (Keras [21] with TensorFlow [22] backend) was conducted. The training process use the same set of samples (7680 images of size 64x64), took roughly 30 mins on GTX1070 (8GB) with 50 epochs.

The original model achieves 92% accuracy (equals CNN3 baseline [5]).

VI. CONCLUSION

Table III shows details of comparison among the 4 models: the baseline CNN3, the lightweight customized U-net, the full size customized U-net and the original U-net architecture. The training time comparison is only for reference since the same machine configuration as the baseline was not available.

By exploring the different techniques and trying to overcome the drawbacks of existing architecture in biomedical image segmentation, this paper proposes a customized architecture of U-net model, which proven to be able to achieve better segmentation results in significant less epochs, computation complexity and reduce training time from hours to minutes compare to the baseline CNN3 [5], and the original U-net model [7].

The Keras-based (TensorFlow backend) [21][22] experimental implementation of both lightweight and full-size version of the customized U-net networks are available at https://github.com/kdung/cell_segmentation.

TABLE II
ARCHITECTURE OF FULL CUSTOMIZED U-NET

Layer	Filter size	Activation	Output size	Padding
Input	-	-	64 x 64 x 3	-
Conv 1	3 x 3	ReLU	64 x 64 x 64	same
Conv 2	3 x 3	Max	64 x 64 x 64	same
Pool 1	2 x 2	ReLU	32 x 32 x 64	-
Conv 3	3 x 3	Max	32 x 32 x 128	same
Conv 4	3 x 3	ReLU	32 x 32 x 128	same
Pool 2	2 x 2	Max	16 x 16 x 128	-
Conv 5	3 x 3	ReLU	16 x 16 x 256	same
Conv 6	3 x 3	ReLU	16 x 16 x 256	same
Pool 3	2 x 2	Max	8 x 8 x 256	-
Conv 7	3 x 3	ReLU	8 x 8 x 512	same
Conv 8	3 x 3	ReLU	8 x 8 x 512	same
Pool 4	2 x 2		4 x 4 x 512	-
Conv 5	3 x 3	ReLU	4 x 4 x 1024	same
Conv 6	3 x 3	ReLU	4 x 4 x 1024	same
Transpose+Concat	2 x 2	-	8 x 8 x 1024	same
Conv7	3 x 3	ReLU	8 x 8 x 512	same
Conv8	3 x 3	ReLU	8 x 8 x 512	same
Transpose+Concat	2 x 2	-	16 x 16 x 512	same
Conv 9	3 x 3	ReLU	16 x 16 x 256	same
Conv 10	3 x 3	ReLU	16 x 16 x 256	same
Transpose+Concat	2 x 2	-	32 x 32 x 256	same
Conv 11	3 x 3	ReLU	32 x 32 x 128	same
Conv 12	3 x 3	ReLU	32 x 32 x 128	same
Transpose+Concat	2 x 2	-	64 x 64 x 128	same
Conv 13	3 x 3	ReLU	64 x 64 x 64	same
Conv 14	3 x 3	ReLU	64 x 64 x 64	same
Output	1 x 1	Sigmoid	64 x 64 x 1	-

TABLE III
PERFORMANCE COMPARISON

Model	Dice's Coefficient	Accuracy	Epoch	Training time
CNN3	0.7623	92%	115	16.5 hours (6GB GPU)
Original U-net	0.8938	92%	50	30 mins (8GB GPU)
Lightweight customized U-net	0.8345	89.1%	50	1 mins (8GB GPU)
Full customized U-net	0.9278	95.1%	20	15 mins (8GB GPU)

ACKNOWLEDGMENT

The authors thank Dr. Matthew Chua and Dr. Tian Jing at National University of Singapore, Institute of Systems Science for their guidance and suggesting new experiments to strengthen the paper.

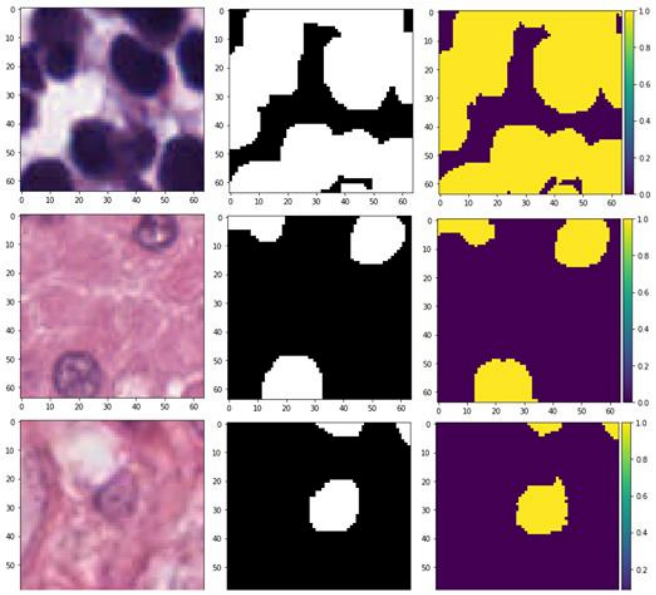

Fig. 6. Sample test results from the experimental full-size customized U-net. The pictures on the left most column are original cropped images from the dataset. The middle column contains corresponding ground truth annotated nuclei. The right most column contains the predicted probability map segmented.

REFERENCES

[1] Y. LeCun, Y. Bengio, and G. Hinton, "Deep learning," *Nature*, vol. 521, no. 7553, pp. 436–444, May 2015.

[2] Ian Goodfellow, Yoshua Bengio, and Aaron Courville, "Convolutional Networks," in *Deep Learning*, Cambridge: MIT Press, 2016, ch. *9*, sec. *2*, pp. *321–363*.

[3] M. E. Plissiti and C. Nikou, "Overlapping cell nuclei segmentation using a spatially adaptive active physical model," *IEEE Trans. Image Process.*,vol. 21, no. 11, pp. 4568–4580, Nov. 2012.

[4] M. N. Gurcan, L. E. Boucheron, A. Can, A. Madabhushi, N. M. Rajpoot, and B. Yener, "Histopathological image analysis: A review," *IEEE Rev. Biomed. Eng.*, vol. 2, pp. 147–171, Oct. 2009.

[5] N. Kumar, R. Verma, S. Sharma, S.Bhargava, A Vahadane and A. Sethi, "Dataset and a Technique for Generalized Nuclear Segmentation for Computational Pathology", *IEEE Trans. Med. Imag*, vol. 36, no. 7, Jul. 2017.

[6] J. Long, E. Shelhamer and T. Darrell "Fully Convolutional Networks for Semantic Segmentation" UC Berkeley. Avaliable: https://people.eecs.berkeley.edu/~jonlong/long_shelhamer_fcn.pdf [Accessed: 14 Aug 2018]

[7] O. Ronneberger, P. Fischer, and T. Brox, "U-Net: Convolutional Networks for Biomedical Image Segmentation", 2015. Computer Science Department and BIOSS Centre for Biological Signalling Studies, University of Freiburg, Germany. Available: https://arxiv.org/pdf/1505.04597.pdf. [Accessed on 14 Aug 2018]

[8] F. Xing, Y. Xie, and L. Yang, "An automatic learning-based framework for robust nucleus segmentation," *IEEE Trans. Med. Imag.*, vol. 35, no. 2, pp. 550–566, Feb. 2016.

[9] H. Llewellyn, "Observer variation, dysplasia grading, and HPV typing: A review," Amer. J. Clin. Pathol., vol. 114, pp. S21–S35, Nov. 2000.

[10] M. Veta, P. J. van Diest, R. Kornegoor, A. Huisman, M. A. Viergever, and J. P. W. Pluim, "Automatic nuclei segmentation in H&E stained breast cancer histopathology images," PLoS ONE, vol. 8, no. 7, p. e70221, 2013

[11] A. Vahadane and A. Sethi, "Towards generalized nuclear segmentation in histological images," in Proc. IEEE 13th Int. Conf. Bioinform. Bioeng. (BIBE), Nov. 2013, pp. 1–4.

[12] J. Odstrcilik et al., "Retinal vessel segmentation by improved matched filtering: Evaluation on a new high-resolution fundus image database," IET Image Process., vol. 7, no. 4, pp. 373–383, Jun. 2013.

[13] A. C. Ruifrok and D. A. Johnston, "Quantification of histochemical staining by color deconvolution," Anal. Quant. Cytol. Histol., vol. 23, no. 4, pp. 291–299, Aug. 2001.

[14] R. Collobert, S. Bengio, and J. Mariéthoz, "Torch: A modular machine learning software library," Idiap Res. Inst., Martigny, Switzerland, Tech. Rep. EPFL-REPORT-82802, 2002.

[15] A. E. Carpenter et al., "CellProfiler: Image analysis software for identifying and quantifying cell phenotypes," Genome Biol., vol. 7, no. 10, p. R100, 2006.

[16] F. Dong et al., "Computational pathology to discriminate benign from malignant intraductal proliferations of the breast," PLoS ONE, vol. 9, no. 12, p. e114885, Dec. 2014.

[17] J. Schindelin et al., "Fiji: An open-source platform for biological-image analysis," Nature Methods, vol. 9, no. 7, pp. 676–682, Jul. 2012.

[18] A. Krizhevsky, I. Sutskever, and G. E. Hinton. Imagenet classification with deep convolutional neural networks. In NIPS, 2012

[19] A. Krizhevsky, I. Sutskever, and G. E. Hinton. Imagenet classification with deep convolutional neural networks. In NIPS, 2012

[20] C. Szegedy, W. Liu, Y. Jia, P. Sermanet, S. Reed, D. Anguelov, D. Erhan, V. Vanhoucke, and A. Rabinovich. Going deeper with convolutions. CoRR, abs/1409.4842, 2014.

[21] Chollet, Francois and others, Keras, 2015, https://keras.io

[22] Martín Abadi, Ashish Agarwal, Paul Barham, Eugene Brevdo, Zhifeng Chen, Craig Citro, Greg S. Corrado, Andy Davis, Jeffrey Dean, Matthieu Devin, Sanjay Ghemawat, Ian Goodfellow, Andrew Harp, Geoffrey Irving, Michael Isard, Rafal Jozefowicz, Yangqing Jia, Lukasz Kaiser, Manjunath Kudlur, Josh Levenberg, Dan Mané, Mike Schuster, Rajat Monga, Sherry Moore, Derek Murray, Chris Olah, Jonathon Shlens, Benoit Steiner, Ilya Sutskever, Kunal Talwar, Paul Tucker, Vincent Vanhoucke, Vijay Vasudevan, Fernanda Viégas, Oriol Vinyals, Pete Warden, Martin Wattenberg, Martin Wicke, Yuan Yu, and Xiaoqiang Zheng. TensorFlow: Large-scale machine learning on heterogeneous systems, 2015. Software available from tensorflow.org