A Multi-Organ Nucleus Segmentation Challenge

Neeraj Kumar, et. *al.*

Supplementary Material

Table S1: *Information about the images used in this paper including their organ, disease type, tissue source site codes and hospital names where the tissue sample was prepared is given below. To know more about TCGA patient codes and hospital details, the readers can consult TCGA website*[1].

#	Patient ID	Organ	Disease type	Tissue Source Site Code	Hospital/Clinic
Training Data					
1	TCGA-A7-A13E-01Z-00-DX1	Breast	Breast invasive carcinoma	A7	Christiana Healthcare
2	TCGA-A7-A13F-01Z-00-DX1	Breast	Breast invasive carcinoma	A7	Christiana Healthcare
3	TCGA-AR-A1AK-01Z-00-DX1	Breast	Breast invasive carcinoma	AR	Mayo Clinic
4	TCGA-AR-A1AS-01Z-00-DX1	Breast	Breast invasive carcinoma	E2	Roswell Park
5	TCGA-E2-A1B5-01Z-00-DX1	Breast	Breast invasive carcinoma	E2	Roswell Park
6	TCGA-E2-A14V-01Z-00-DX1	Breast	Breast invasive carcinoma	E2	Roswell Park
7	TCGA-B0-5711-01Z-00-DX1	Kidney	Kidney renal clear cell carcinoma	B0	University of Pittsburgh
8	TCGA-HE-7128-01Z-00-DX1	Kidney	Kidney renal papillary cell carcinoma	HE	Ontario Institute for Cancer Research (OICR)
9	TCGA-HE-7129-01Z-00-DX1	Kidney	Kidney renal papillary cell carcinoma	HE	Ontario Institute for Cancer Research (OICR)
10	TCGA-HE-7130-01Z-00-DX1	Kidney	Kidney renal papillary cell carcinoma	HE	Ontario Institute for Cancer Research (OICR)
11	TCGA-B0-5710-01Z-00-DX1	Kidney	Kidney renal clear cell carcinoma	B0	University of Pittsburgh
12	TCGA-B0-5698-01Z-00-DX1	Kidney	Kidney renal clear cell carcinoma	B0	University of Pittsburgh
13	TCGA-18-5592-01Z-00-DX1	Liver	Lung squamous cell carcinoma	18	Princess Margaret Hospital (Canada)
14	TCGA-38-6178-01Z-00-DX1	Liver	Lung adenocarcinoma	38	University of North Carolina
15	TCGA-49-4488-01Z-00-DX1	Liver	Lung adenocarcinoma	49	Johns Hopkins
16	TCGA-50-5931-01Z-00-DX1	Liver	Lung adenocarcinoma	50	University of Pittsburgh
17	TCGA-21-5784-01Z-00-DX1	Liver	Lung squamous cell carcinoma	21	Fox Chase Cancer Center
18	TCGA-21-5786-01Z-00-DX1	Liver	Lung squamous cell carcinoma	21	Fox Chase Cancer Center
19	TCGA-G9-6336-01Z-00-DX1	Prostate	Prostate adenocarcinoma	G9	Roswell Park
20	TCGA-G9-6348-01Z-00-DX1	Prostate	Prostate adenocarcinoma	G9	Roswell Park

[1] https://wiki.nci.nih.gov/display/TCGA/TCGA+barcode

21	TCGA-G9-6356-01Z-00-DX1	Prostate	Prostate adenocarcinoma	G9	Roswell Park
22	TCGA-G9-6363-01Z-00-DX1	Prostate	Prostate adenocarcinoma	G9	Roswell Park
23	TCGA-CH-5767-01Z-00-DX1	Prostate	Prostate adenocarcinoma	CH	Indivumed
24	TCGA-G9-6362-01Z-00-DX1	Prostate	Prostate adenocarcinoma	G9	Roswell Park
25	TCGA-DK-A2I6-01A-01-TS1	Bladder	Bladder Urothelial Carcinoma	DK	Memorial Sloan Kettering
26	TCGA-G2-A2EK-01A-02-TSB	Bladder	Bladder Urothelial Carcinoma	G2	MD Anderson
27	TCGA-AY-A8YK-01A-01-TS1	Colon	Colon adenocarcinoma	AY	University of North Carolina
28	TCGA-NH-A8F7-01A-01-TS1	Colon	Colon adenocarcinoma	NH	Candler
29	TCGA-KB-A93J-01A-01-TS1	Stomach	Stomach adenocarcinoma	KB	University Health Network, Toronto
30	TCGA-RD-A8N9-01A-01-TS1	Stomach	Stomach adenocarcinoma	RD	Peter MacCallum Cancer Center
Testing Data					
1	TCGA-AC-A2FO-01A-01-TS1	Breast	Breast Invasive Carcinoma	AC	International Genomics Consortium
2	TCGA-AO-A0J2-01A-01-BSA	Breast	Breast invasive carcinoma	AO	Memorial Sloan Kettering Cancer Center
3	TCGA-2Z-A9J9-01A-01-TS1	Kidney	Kidney renal papillary cell carcinoma	2Z	Moffitt Cancer Center
4	TCGA-GL-6846-01A-01-BS1	Kidney	Kidney renal papillary cell carcinoma	GL	ABS-IUPUI
5	TCGA-IZ-8196-01A-01-BS1	Kidney	Kidney renal papillary cell carcinoma	IZ	ABS - Lahey Clinic
6	TCGA-EJ-A46H-01A-03-TSC	Prostate	Prostate Adenocarcinoma	EJ	University of Pittsburgh
7	TCGA-HC-7209-01A-01-TS1	Prostate	Prostate Adenocarcinoma	HC	International Genomics Consortium
8	TCGA-CU-A0YN-01A-02-BSB	Bladder	Bladder Urothelial Carcinoma	CU	University of North Carolina
9	TCGA-ZF-A9R5-01A-01-TS1	Bladder	Bladder Urothelial Carcinoma	ZF	University of Sheffield
10	TCGA-A6-6782-01A-01-BS1	Colon	Colon Adenocarcinoma	A6	Christiana Healthcare
11	TCGA-44-2665-01B-06-BS6	Lung	Lung Adenocarcinoma	44	Christiana Healthcare
12	TCGA-69-7764-01A-01-TS1	Lung	Lung Adenocarcinoma	69	Washington University - Cleveland Clinic
13	TCGA-FG-A4MU-01B-01-TS1	Brain	Brain Lower Grade Glioma	FG	Case Western
14	TCGA-HT-8564-01Z-00-DX1	Brain	Brain Lower Grade Glioma	HT	Case Western - St Joes

Table S2: *Variation of the manual annotation errors with the tissue (or organ) type for both training and testing images of MoNuseg 2018*

#	Organ	# of nuclei annotated	# of erroneous annotations
1	Breast	3,561	29
2	Liver	3,103	18
3	Kidney	9,609	71
4	Prostate	3,704	28
5	Bladder	2,049	15
6	Colon	2,165	14
7	Stomach	2,556	19
8	Lung	1,323	57
9	LGG	776	32
Total		**28,846**	**283**

Protocol used for the evaluation of the annotation quality by an expert pathologist

We sent annotated images to an expert pathologist for examination of annotation quality. In a PowerPoint® deck, we used one image per slide. On a slide, we put the unannotated and annotated images side by side to cover a large portion of the slide. The pathologist viewed the slide on a 25" monitor, and was instructed to place an arrow shape on every problematic annotation, whether it was a false positive, a false negative, an over-segmented, or an under-segmented nucleus. Examples of the pathologist's assessment are shown in Figure S2. Each of the sub-figures covered almost a complete PowerPoint® slide.

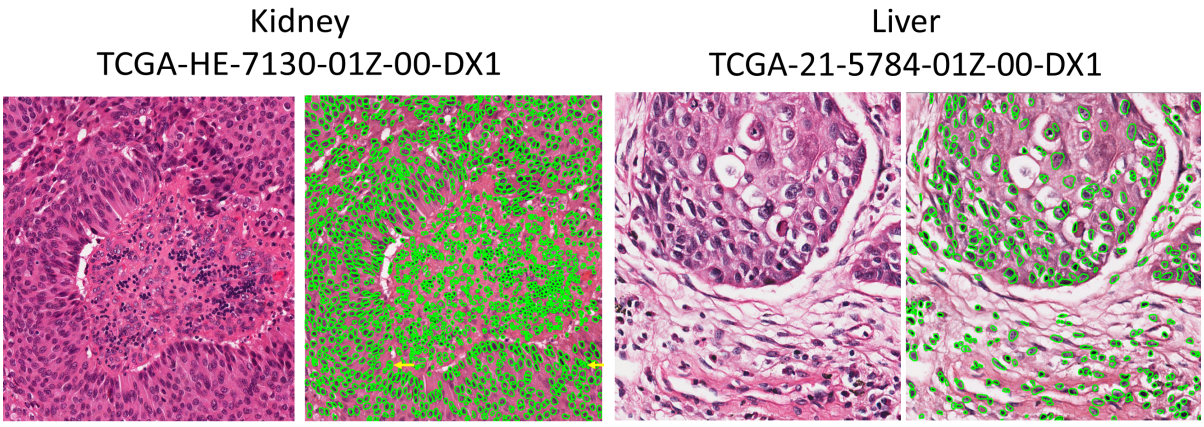


Figure S2 (a) Pathologist's Comment: "Tough due to extreme variation in nuclear size and shape – but good overall. Note over-segmentation (yellow arrows)." **(b) Pathologist's Comment**: "Very good recognition of overlapping nuclei. This is an unusual and difficult image."

We counted all the arrows and divided the count by the number of annotated nuclei in those images to estimate that our annotators made less than 1% errors on any given image. The total number of annotated nuclei and the annotation errors pointed by the expert pathologist for each tissue type are given in Table S2. We left these errors uncorrected due to their low count as evident from Table S2.

Table S3: *Comparison of organ level a-AJI and overall a-AJI*

Rank	Team	Organ Level Results a-AJI (95% CI)							Overall a-AJI (95% CI) n = 14
		Breast n=2	Kidney n=3	Prostate n=2	Bladder n=2	Colon[2] n=1	Lung n = 2	LGG n=2	
1	CUHK & IMSIGHT	0.646 (0.630-0.663)	0.707 (0.698-0.717)	0.706 (0.702-0.709)	0.731 (0.695-0.768)	0.641 (NA-NA)	0.696 (0.670-0.722)	0.673 (0.653-0.692)	0.691 (0.680-0.702)
2	BUPT.J.LI	0.644 (0.629-0.660)	0.701 (0.685-0.716)	0.705 (0.702-0.708)	0.723 (0.694-0.753)	0.615 (NA-NA)	0.696 (0.677-0.715)	0.681 (0.669-0.692)	0.687 (0.676-0.697)
3	pku.hzq	0.646 (0.637-0.655)	0.698 (0.681-0.715)	0.702 (0.702-0.703)	0.708 (0.682-0.735)	0.611 (NA-NA)	0.696 (0.673-0.718)	0.692 (0.689-0.695)	0.685 (0.675-0.695)
4	Yunzhi	0.632 (0.626-0.639)	0.694 (0.681-0.708)	0.704 (0.700-0.708)	0.695 (0.657-0.732)	0.619 (NA-NA)	0.695 (0.665-0.725)	0.675 (0.655-0.695)	0.679 (0.668-0.690)
5	Navid Alemi	0.621 (0.614-0.627)	0.692 (0.675-0.708)	0.703 (0.699-0.707)	0.712 (0.678-0.746)	0.612 (NA-NA)	0.684 (0.654-0.714)	0.682 (0.674-0.690)	0.678 (0.666-0.689)
6	xuhuaren	0.608 (0.606-0.611)	0.680 (0.665-0.696)	0.696 (0.695-0.697)	0.677 (0.638-0.715)	0.606 (NA-NA)	0.688 (0.660-0.716)	0.656 (0.636-0.676)	0.664 (0.652-0.676)
7	aetherAI	0.628 (0.613-0.643)	0.677 (0.662-0.691)	0.660 (0.652-0.668)	0.707 (0.683-0.731)	0.602 (NA-NA)	0.672 (0.658-0.685)	0.659 (0.637-0.680)	0.663 (0.653-0.673)
8	Shuang Yang	0.624 (0.611-0.637)	0.680 (0.666-0.695)	0.672 (0.665-0.679)	0.688 (0.660-0.716)	0.583 (NA-NA)	0.672 (0.647-0.698)	0.668 (0.654-0.683)	0.662 (0.652-0.673)
9	Bio-totem & SYSUCC	0.617 (0.612-0.623)	0.677 (0.665-0.690)	0.684 (0.678-0.690)	0.686 (0.658-0.715)	0.592 (NA-NA)	0.679 (0.662-0.695)	0.655 (0.629-0.680)	0.662 (0.652-0.672)
10	Amirreza Mahbod	0.628 (0.616-0.641)	0.673 (0.661-0.685)	0.680 (0.672-0.688)	0.687 (0.665-0.710)	0.601 (NA-NA)	0.666 (0.661-0.672)	0.630 (0.626-0.634)	0.657 (0.649-0.666)
11	CMU-UIUC	0.612 (0.611-0.613)	0.669 (0.657-0.681)	0.666 0.657-0.674)	0.694 (0.664-0.725)	0.580 (NA-NA)	0.685 (0.655-0.715)	0.639 (0.633-0.644)	0.656 (0.645-0.667)
12	Graham&Vu	0.624 (0.617-0.632)	0.674 (0.650-0.699)	0.652 (0.649-0.655)	0.639 (0.625-0.653)	0.580 (NA-NA)	0.682 (0.661-0.703)	0.673 (0.670-0.677)	0.653 (0.643-0.663)
13	Unblockabulls	0.594 (0.568-0.620)	0.663 (0.654-0.671)	0.665 (0.665-0.667)	0.714 (0.675-0.754)	0.557 (NA-NA)	0.648 (0.613-0.683)	0.666 (0.663-0.669)	0.651 (0.637-0.666)
14	Tencent AI Lab	0.596 (0.584-0.608)	0.668 (0.653-0.683)	0.659 (0.656-0.663)	0.684 (0.655-0.713)	0.574 (NA-NA)	0.651 (0.622-0.679)	0.642 (0.635-0.649)	0.646 (0.635-0.657)
15	DeepMD	0.576 (0.569-0.584)	0.643 (0.637-0.650)	0.679 (0.659-0.699)	0.682 (0.642-0.721)	0.602 (NA-NA)	0.651 (0.623-0.678)	0.578 (0.544-0.612)	0.633 (0.619-0.647)
16	Cannon Medical Research Europe	0.627 (0.626-0.629)	0.653 (0.633-0.672)	0.664 (0.629-0.699)	0.704 (0.672-0.737)	0.577 (NA-NA)	0.674 (0.645-0.702)	0.487 (0.344-0.629)	0.633 (0.604-0.661)
17	Johannes Stegmaier	0.547 (0.515-0.579)	0.646 (0.625-0.667)	0.665 (0.665-0.666)	0.664 (0.627-0.701)	0.610 (NA-NA)	0.659 (0.639-0.678)	0.553 (0.469-0.637)	0.623 (0.603-0.643)
18	Yanping	0.586 (0.572-0.600)	0.625 (0.611-0.638)	0.644 (0.636-0.651)	0.680 (0.630-0.731)	0.577 (NA-NA)	0.633 (0.614-0.652)	0.592 (0.568-0.616)	0.623 (0.610-0.636)
19	Philipp Gruening	0.570 (0.559-0.580)	0.644 (0.621-0.667)	0.612 (0.562-0.661)	0.683 (0.653-0.712)	0.540 (NA-NA)	0.614 (0.580-0.647)	0.634 (0.612-0.656)	0.621 (0.606-0.636)
20	Agilent Labs	0.570 (0.570-0.571)	0.650 (0.632-0.668)	0.655 (0.645-0.665)	0.667 (0.635-0.700)	0.560 (NA-NA)	0.647 (0.612-0.683)	0.532 (0.439-0.624)	0.618 (0.598-0.638)
21	Konica	0.569	0.623	0.615	0.583	0.552	0.657	0.643	0.611

[2] For n = 1, confidence intervals are indeterminate

	Minolta Laboratory Europe	(0.560-0.578)	(0.598-0.647)	(0.614-0.616)	(0.572-0.594)	(NA-NA)	(0.656-0.659)	(0.641-0.646)	(0.601-0.622)
22	OnePiece	0.538 (0.526-0.550)	0.617 (0.607-0.627)	0.635 (0.626-0.644)	0.673 (0.622-0.723)	0.551 (NA-NA)	0.604 (0.577-0.632)	0.591 (0.579-0.603)	0.606 (0.592-0.692)
23	Junma	0.541 (0.522-0.560)	0.602 (0.580-0.623)	0.601 (0.568-0.634)	0.629 (0.597-0.661)	0.516 (NA-NA)	0.595 (0.584-0.606)	0.626 (0.625-0.627)	0.593 (0.581-0.606)
24	Biosciences R&D, TCS Research	0.565 (0.560-0.569)	0.602 (0.572-0.631)	0.626 (0.578-0.674)	0.685 (0.656-0.713)	0.541 (NA-NA)	0.629 (0.590-0.668)	0.368 (0.160-0.577)	0.578 (0.538-0.619)
25	Azam Khan	0.514 (0.511-0.517)	0.610 (0.592-0.627)	0.614 (0.595-0.634)	0.606 (0.532-0.680)	0.546 (NA-NA)	0.515 (0.461-0.569)	0.588 (0.544-0.632)	0.575 (0.556-0.594)
26	CVBLab	0.526 (0.504-0.548)	0.562 (0.538-0.585)	0.612 (0.583-0.641)	0.640 (0.597-0.683)	0.523 (NA-NA)	0.570 (0.551-0.589)	0.566 (0.546-0.585)	0.574 (0.560-0.588)
27	Linmin Pei	0.551 (0.550-0.552)	0.517 (0.483-0.550)	0.592 (0.559-0.626)	0.608 (0.562-0.654)	0.565 (NA-NA)	0.582 (0.547-0.617)	0.545 (0.536-0.553)	0.562 (0.548-0.577)
28	DB-KR-JU	0.486 (0.475-0.497)	0.475 (0.462-0.487)	0.302 (0.176-0.429)	0.507 (0.475-0.539)	0.443 (NA-NA)	0.479 (0.444-0.514)	0.473 (0.459-0.488)	0.455 (0.428-0.481)
29	VISILAB	0.422 (0.376-0.467)	0.406 (0.371-0.441)	0.419 (0.393-0.444)	0.378 (0.377-0.378)	0.468 (NA-NA)	0.553 (0.544-0.563)	0.494 (0.471-0.517)	0.444 (0.425-0.463)
30	Sabarinathan	0.331 (0.294-0.368)	0.462 (0.444-0.480)	0.467 (0.453-0.481)	0.397 (0.371-0.424)	0.424 (NA-NA)	0.553 (0.535-0.572)	0.452 (0.404-0.501)	0.444 (0.424-0.464)
31	Silvers	0.445 (0.355-0.535)	0.136 (0.118-0.154)	0.171 (0.118-0.225)	0.394 (0.252-0.535)	0.073 (NA-NA)	0.368 (0.274-0.462)	0.329 (0.178-0.480)	0.278 (0.228-0.328)
32	TJ	0.111 (0.105-0.117)	0.096 (0.059-0.134)	0.089 (0.041-0.137)	0.122 (0.096-0.148)	0.077 (NA-NA)	0.284 (0.277-0.290)	0.122 (0.036-0.208)	0.130 (0.106-0.154)

MH-FCN: Multi-Organ Nuclei Segmentation Algorithm

Yanning Zhou[1], Omer Fahri Onder[2], Efstratios Tsougenis[2] and Hao Chen[2]

[1] The Chinese University of Hong Kong
ynzhou@cse.cuhk.edu.hk
[2] Imsight Medical Technology, Inc., Hong Kong
{fahri,etsougenis,chenhao}@imsightmed.com

1 Data Preprocessing

The dataset is divided into one training set and 2 validation sets based on organs, which includes 16, 8, 6 histopathology image respectively. The second validation set which contains unseen organs of bladder, colon and stomach according to [2]. In order to add contour prediction, we create contour labels based on the given ground truth. We dilate the ground truth mask using a 3×3 kernel with iteration of 1, then do subtraction operation to get contour labels.

As the difference in stain color might affect the training, the stain normalization method described in [4] is used to normalize the stain color of histopathology images of different organs.

We do data augmentation during training phase, including randomly crop patches with a size of $(256, 256)$, random affine transformation, random rotation and random color jitter with a range of ± 10. For validation and testing stages however, we use a sliding window with a stride of 128 to crop patches of same size and fuse the results together to get the final output.

2 Methodology

We designed a multi-head fully convolutional network(MH-FCN) to simultaneously segment nuclei and contour in histopathology images, which is inspired by FCN [3] and U-net [5] architecture. Considering that the sub-tasks of nuclei and contour segmentation are highly correlated, we use single encoder to encoder the information into feature space and one specific decoder for each task.

We use 50 layers resnet [1] as backbone of top-down encoder, which takes advantages of residual learning. Specifically, the original input with size of (H, W) is fed into a 7×7 convolution with stride 2, followed by a max pooling 3×3 with stride 2. Then, we hierarchically stack four modules, each containing $\{3, 4, 6, 3\}$ residual blocks. In each residual block $x_{i+1} = \mathcal{W}_s x_i + \mathcal{F}_i(x_i; \{\mathcal{W}_s\})$, we utilize bottleneck architecture which is a stack of $\{1\times1, 3\times3, 1\times1\}$ convolutional layers followed by batch normalization (BN) and ReLU nonlinearity to conduct the residual function $\mathcal{F}_i$. The $\mathcal{W}_s$ is used to downsample feature maps by 1×1 convolution with a stride of 2, otherwise, the $\mathcal{W}_s x_i$ is identity mapping. The output features' size

from each module are $(H/4, W/4)$, $(H/8, W/8)$, $(H/16, W/16)$ and $(H/32, W/32)$ respectively while the numbers of feature maps are $\{256, 512, 1024, 2048\}$. Then the highest output features are feed into a convolution with3×3 kernel to compress feature number into 256, which is considered as the input of top-down decoders.

Since lower layer's features preserve accurate location information while higher layer's features contain the highly abstract representations, we have implemented a structure to make full use of multi-resolution feature information. For each encoder module's output, it first passes through a 1×1 convolution to compress the feature number into 256, then the bottom-up decoder aggregates compressed features with the bilinear upsampling upper module features in decoders by summation operation, followed by a 3×3 convolution to fuse multi-level information.

The size of each module's output feature map is $(H/16, W/16)$, $(H/8, W/8)$ and $(H/4, W/4)$ respectively. For each decoder, a 1×1 convolution is served as a classifier to predict the probability map in each module stage. The probability maps are resized into ground truth size using bilinear upsampling, followed a $3\times3\}$ convolution without nonlinear function to smooth the prediction.

We use binary cross-entropy loss function during optimization. The loss function can be written as follow

$$\mathcal{L} = \mathcal{L}_{nuclei} + \lambda\mathcal{L}_{contour} \tag{1}$$

, where

$$\mathcal{L}_S = -\frac{1}{N}\sum_{i=1}^{N}\Big(\mathbb{1}(y_i=0)\log p_i^{bkg} + \alpha\mathbb{1}(y_i=1)\log p_i^{S}\Big)\ S = nuclei, contour \tag{2}$$

For the final output, we subtraction the nuclei prediction and contour prediction to split the cluster cells.

3 Experiment Result

We use aggregated Jaccard index(AJI) proposed in [2] to evaluate our results.

	train	val1	val2
MH-FCN	0.658	0.597	0.547

References

1. He, K., Zhang, X., Ren, S., Sun, J.: Deep residual learning for image recognition. In: IEEE CVPR

2. Kumar, N., Verma, R., Sharma, S., Bhargava, S., Vahadane, A., Sethi, A.: A dataset and a technique for generalized nuclear segmentation for computational pathology. IEEE TMI
3. Long, J., Shelhamer, E., Darrell, T.: Fully convolutional networks for semantic segmentation. In: IEEE CVPR
4. Macenko, M., Niethammer, M., Marron, J.S., Borland, D., Woosley, J.T., Guan, X., Schmitt, C., Thomas, N.E.: A method for normalizing histology slides for quantitative analysis. In: 2009 IEEE International Symposium on Biomedical Imaging: From Nano to Macro. pp. 1107–1110 (June 2009). https://doi.org/10.1109/ISBI.2009.5193250
5. Ronneberger, O., Fischer, P., Brox, T.: U-net: Convolutional networks for biomedical image segmentation. In: MICCAI. Springer

Center Vector For Nuclei Instance Segmentation

Jiahui Li

Beijing University of Post and Telecommunication
13261088750@163.com

1 Introduction

Nuclei segmentation is a problem of instance segmentation in which occlusal nuclei of same semantic class but different instance shall be saperated. To our knowledge there are mainly two types of solution: MaskRCNN [1] and FCN [2,3,4]. MaskRCNN firstly crop out each instance and then perform semantic segmentation for this instance. FCN preprocess the groundtruth to have gap between each instance and transform the problem from instance segmentation to semantic segmentation. In nuclei instance segmentation problem, usually crowded nuclei gather together with heavy instance occlusion. Towards this phenomenon we propose a new solution: Center Vector. For each pixel we simultaneously output the semantic prediction , instance center and geometric center vector then cluster each instance prediction. This methology not only runs much faster than MaskRCNN but also requires no boundary preprocess under prior knowledge when compared to FCN. According to our experiment this methology outperforms current state of art [4].

2 Data PreProcessing

We follow the same preprocess procedure as [4] to suppress variability. During training, we feed 256*256 images as input with augmentation: Random crop, flip, rotation, scaling , noise. During inference we take the entire 1000*1000 images as input.

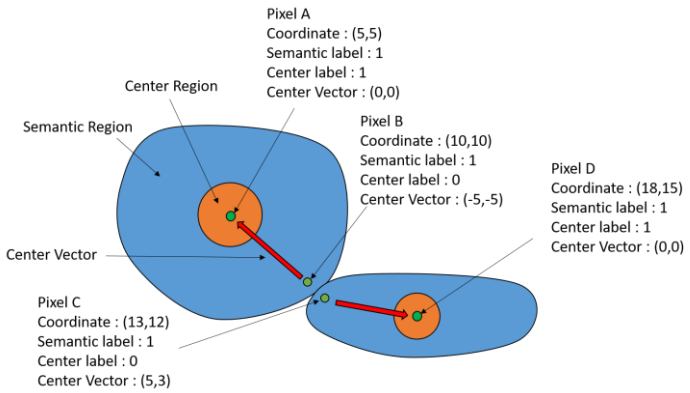


Fig. 1. Label definition of Center Vector.Pixels at edges of different instance differs quite a lot.

3 Proposed method

As shown in Fig.1 Semantic prediction means for each pixel predict as nuclei or background. Instance center groundtruth is generated in this procedure: For each instance mask firstly erode it with 5*5 kernel and calculate its geometric center as the average x,y of each instance pixel. Secondly keep the pixels whose distance from geometric center is smaller than 30% of the max distance within this instance as true. Geometric center vector for each pixel is its dx and dy towards belonging instance geometric center.

We utilized Deep Layer Aggregation (DLA) [2] as base model. At the output layer we give 2 channels for semantic prediction, 2 channels for instance center and 2 channels for geometric center vector regresion. The loss function is the sum of cross entropy loss, IOU loss and mean square loss.

4 Post-processing

At instance center prediction map we search connected component as each instance center. For each pixel, if its semantic prediction is true and geometric center vector head towards the region of one instance center, this pixel belongs to this instance otherwise to the nearest instance.We perform 8 test time augmentation (rotate&flip) and take average probability as output.

5 Results

As shown in Table.1, our methology outperforms the current state of art.

	CP2	Fiji	CNN3	Center Vector	Center Vector&TTA
AJI[4]	0.1232	0.2733	0.5083	0.5732	**0.5901**

Table 1. AJI comparasion with other methods. Center Vector with Test time augmentation is the best.

References

1. Kaiming He, Georgia Gkioxari, Piotr Dollár, Ross Girshick. “MaskRCNN” arXiv:1703.06870v3
2. Fisher Yu, Dequan Wang, Evan Shelhamer, Trevor Darrell “Deep Layer Aggregation” (2018) arXiv: 1707.06484
3. Olaf Ronneberger, Philipp Fischer, Thomas Brox, “U-Net: Convolutional Networks for Biomedical Image Segmentation”, arXiv:1505.04597
4. N. Kumar, R. Verma, S. Sharma, S. Bhargava, A. Vahadane and A. Sethi, "A Dataset and a Technique for Generalized Nuclear Segmentation for Computational Pathology," in IEEE Transactions on Medical Imaging, vol. 36, no. 7, pp. 1550-1560, July 2017
5. Abhinav Shrivastava, Abhinav Gupta, Ross Girshick. “Training Region-based Object Detectors with Online Hard Example Mining”. arXiv:1604.03540

Multi-Organ Nuclei Segmentation via incorporating UNet and Mask R-CNN

Zhiqiang Hu

Peking University, China
huzq@pku.edu.cn

1 Introduction

The Multi-Organ Nuclei Segmentation Challenge (MoNuSeg) aims to analyze histology images from multiple patients and organs of different hospitals, and identify different types of nuclei in the form of segmentation. Nuclei segmentation is important because accurate nuclei segmentation can enable extraction of high-quality features, such as density, size and counts, which in turn is important for cancer assessment and so on.

The task can be formulated as an instance-segmentation problem. The instance segmentation problem requires to detect and also segment each nucleus instance in the histology images. It combines elements from object detection and semantic segmentation and thus is challenging. There are two main ways to tackle the problem:

1. bottom-up. To firstly perform the semantic segmentation and then differentiate instances by post-processing;
2. top-down. To firstly detect each object instance and then perform the segmentation within the bounding box.

The two approaches both have their own advantages and disadvantages, and may be suitable for different situations. As a result, in this paper we incorporate the two approaches to make the model more robust. For the bottom-up approach we propose to apply UNet[1] while for the top-down approach we propose to apply Mask R-CNN[2], both of which are the state-of-art instance-segmentation frameworks. We apply extensive data augmentation to enhance the generalization ability. Experiments show that our model can effectively segment nuclei in diverse images.

2 Data Pre-processing

We normalize the image for each channel for training and inference. For training we utilize patch-based approach, and for inference we simple process the whole slides for efficiency. We apply extensive data augmentation to enhance the generalization ability, including random crop, random horizontal and vertical flip, random rotation, random scaling, noising and so on.

3 Proposed Model

For the bottom-up approach, we propose to apply UNet. UNet is firstly developed for semantic segmentation and merges information from multiple scales effectively. We utilize UNet for the instance segmentation task by predicting semantic masks and borders simultaneously. The borders are used to split overlapping nuclei. We also add relative position supervisions to facilitate training. For the top-down approach, we propose to apply Mask R-CNN. Mask R-CNN extends Faster R-CNN by adding head for the segmentation task. Also, the proposed RoIAlign layer in Mask R-CNN aligns the extracted features with the input. It avoids any quantization by bilinear interpolation. The alignment is important for pixel-level tasks such as segmentation.

4 Post-Processing

With the instance segmentation predictions by both UNet and Mask R-CNN, we firstly obtain semantic masks of each model and ensemble by voting. Then we calculate the center masks of each model by eroding operation, and ensemble the center masks as well. Finally, we utilize random walker algorithm to obtain instance segmentation masks from the ensembled semantic masks and center masks.

5 Results

We apply 3-fold cross validation for evaluation. We incorporate three UNet models with AJI[5] scores 0.605, 0.599, 0.589 and one Mask R-CNN model with AJI[5] score 0.565 and the final AJI[5] score is 0.616.

References

1. Ronneberger, O., Fischer, P., Brox, T.: U-net: Convolutional networks for biomedi- cal image segmentation. In: International Conference on Medical Image Computing and Computer-Assisted Intervention. pp. 234–241. Springer (2015)
2. He K, Gkioxari G, Dollar P, et al. Mask R-CNN[J]. IEEE Transactions on Pattern Analysis & Machine Intelligence, 2017, PP(99):1-1. Author, F., Author, S.: Title of a proceedings paper. In: Editor, F., Editor, S. (eds.) CONFERENCE 2016, LNCS, vol. 9999, pp. 1–13. Springer, Heidelberg (2016).
3. He K, Zhang X, Ren S, et al. Deep Residual Learning for Image Recognition[C]// IEEE Conference on Computer Vision and Pattern Recognition. IEEE Computer Society, 2016:770-778.Author, F.: Contribution title. In: 9th International Proceedings on Proceedings, pp. 1–2. Publisher, Location (2010).
4. Lin T Y, Dollar P, Girshick R, et al. Feature Pyramid Networks for Object Detection[J]. 2016:936-944.
5. Kumar N, Verma R, Sharma S, et al. A Dataset and a Technique for Generalized Nuclear Segmentation for Computational Pathology[J]. IEEE Transactions on Medical Imaging, 2017, 36(7):1550-1560.

Fully Convolutional Neural Network for Multi-Organ Nuclei Segmentation

Yunzhi Wang

University of Oklahoma

Data pre-processing

The images are normalized using the mean and standard deviation values obtained from ImageNet dataset for each color channel. 512 * 512 patches are randomly cropped from the original images as the input of the proposed segmentation model. Due to the limitation of the size of the dataset, we employed extensive data augmentation process to avoid over-fitting issue. The data augmentation includes contrast-limited adaptive histogram equalization (CLAHE), scaling, up/down and right/left flip, rotation of 90/180/270 degrees, color jitter, Gaussian noise and elastic transformation. Training our model requires a number of pixel-level labels for each image, including a nuclei label, an adjacent-boundary label and a pixel-to-center-vector label. The nuclei labels can be simply obtained by setting all background pixels as negative and all nuclei pixels as positive. The adjacent-boundary labels highlight the boundaries between adjacent nuclei and therefore they are important for separating different nuclei. The process of generating adjacent-boundary labels includes a morphological dilation operation to enlarge the nuclei area and a watershed algorithm to separate different nuclei. The dilated watershed lines are adopted as adjacent-boundary labels. The pixel-to-center-vector labels are obtained by calculating the unit vectors pointing from each positive pixel to the center of the nuclear it belongs to.

Proposed model

We employ a cascaded fully convolutional network (FCN) model for the nuclei segmentation task as shown in Figure 1. The model consists of two consecutive encoder-decoder structures which are similar to U-Net [1]. For the first network, we adopt the convolutional layers of a pre-trained res-net 34 [2] as the encoder. The output of the network consists of probability maps for predicting nuclei labels and pixel-to-center-vector labels respectively. The outputs and feature maps from the last layers are concatenated as the input of the second network, which is a standard UNet for predicting the adjacent boundary labels. In the testing phase, we only calculate the probability maps of nuclei labels and adjacent boundary labels. A testing-time-augmentation

(TTA) strategy is applied to ensemble the results. The TTA includes CLAHE transformation and up/down, right/left flip. Probability maps obtained by different augmentation operations are averaged as the final predictions.

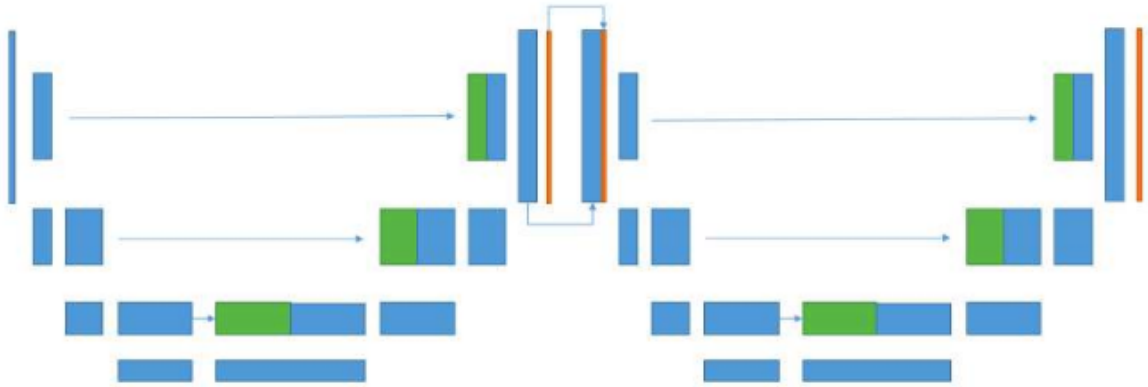


Figure 1: Cascaded U-Net structure.

Post-processing

By thresholding the feature maps generated by the proposed model, we can obtain a predicted nuclei mask and a predicted adjacent-boundary mask for each input image. A marker mask is obtained by subtracting the adjacent-boundary mask from the nuclei mask, followed by a morphological erosion operation. Ideally different nuclei are not connected to each other in the marker masks. A random walker or watershed algorithm is then applied on the nuclei masks and marker masks to obtain the final instance segmentation results.

Results

We use a 3-fold cross validation process to evaluate the performance of our proposed model. The average aggregated Jaccard index (AJI) [3] is 0.625 over the 30 cases in the dataset.

[1] Ronneberger, Olaf, P. Fischer, and T. Brox. "U-Net: Convolutional Networks for Biomedical Image Segmentation." 9351(2015):234-241.

[2] He, Kaiming, et al. "Deep Residual Learning for Image Recognition." *IEEE Conference on Computer Vision and Pattern Recognition* IEEE Computer Society, 2016:770-778.

[3] Kumar, Neeraj, et al. "A Dataset and a Technique for Generalized Nuclear Segmentation for Computational Pathology." *IEEE Transactions on Medical Imaging* 36.7(2017):1550-1560.

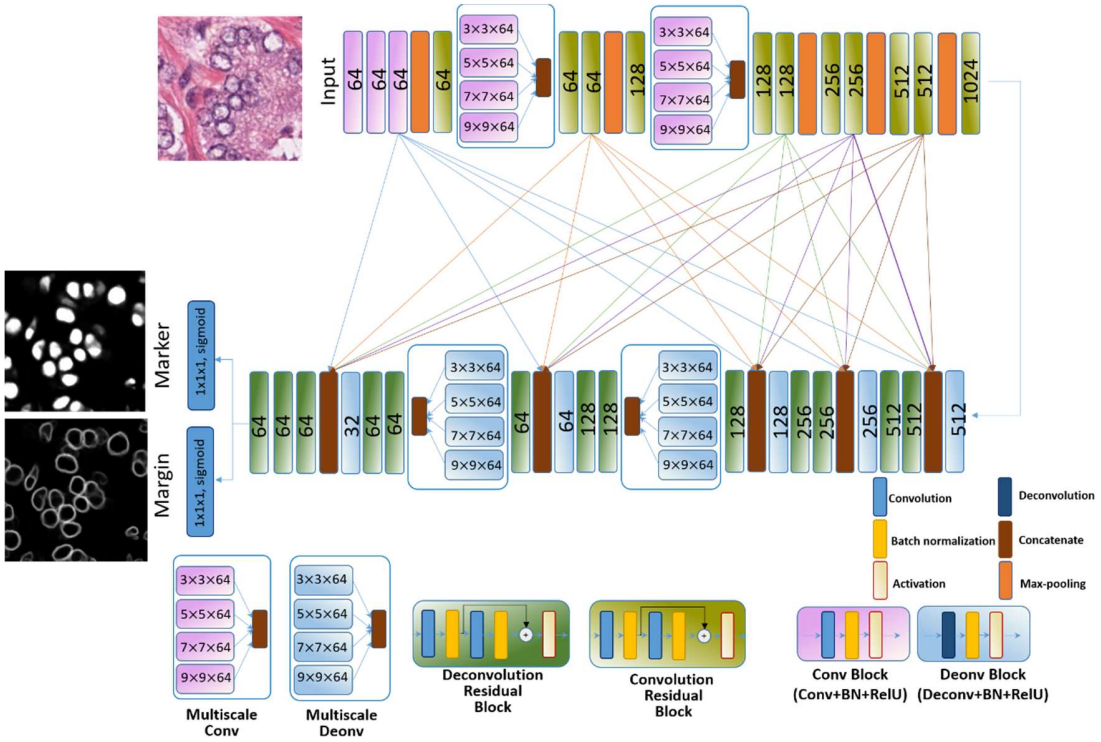
SpaghettiNet: A Multi-scale Feature Sharing Network for Nuclei Instance Segmentation in Histology Images

Navid Alemi Koohbanani, Mostafa Jahanifar, Neda Zamani Tajeddin, Ali Gooya, and Nasir Rajpoot

Proposed Approach:

We proposed to use a marker-controlled watershed algorithm to segment nuclei instances. To this end, nuclei masks and markers are required. Masks represent nuclei regions in the image (not necessarily separated) whereas markers are indicators of the individual nucleus. Marker-controlled watershed enables us to separate each nuclei region using its related marker.

To obtain masks and markers a multi-scale feature sharing network is utilized. We call this network SpaghettiNet due to large numbers of skip connections between encoder and decoder part, as illustrated in Fig. 1. The general architecture of SpaghettiNet comprises encoding and decoding paths. In encoding path, convolutional, residual, and a novel multi-scale component are stacked sequentially. Our proposed multi-scale component is formed by concatenating four convolutional layers with different kernel sizes. In the decoding path, the similar components are used, but instead of using convolutions, we incorporated transposed convolution (deconvolution) layers. We leveraged extensive inter-level skip connections through a novel shape adapting



component to form a multi-scale feature sharing network. This new property empowers the network in all levels of the decoding path to gain the abstract information from all levels of encoding path. This will enrich the features, facilitate the training process, and speed up the convergence.

For marker detection, a double-head SpaghettiNet was used, one head for predicting nuclei markers and the other one estimating the nuclei margin as an auxiliary output (Fig. 1). As the network has two different outputs, it will penalize two different loss functions.

For the marker head, a hybrid loss function, comprising a novel weighted Dice index and a binary cross entropy, is introduced to force the network learning the nuclei margins and separate touching nuclei properly:

$$L_{\text{marker}} = 1 - \underbrace{\frac{2\sum y_p y_t}{\sum w_{\text{map}} y_p + \sum y_t}}_{\text{Weighted Dice}} - \underbrace{\frac{1}{N}\sum y_t \log(y_p) + (1-y_t)\log(1-y_p)}_{\text{Binary Cross Entropy (BCE)}}, \tag{1}$$

In the above equation, y_p is the predicted output, y_t is the ground truth, and w_{map} is an adaptive weight map emphasizing on nuclei margins. This loss would strictly penalize the network if it predicts the nuclei margins as a part of nuclei markers. The proposed weighted Dice index will help the network predict the cell marker, compensate for the imbalance population of pixel label and more importantly avoids wrong prediction of nuclei margins (reducing the false positives). On the other hand, the binary cross entropy part of the loss function improves the pixel-wise classification and prediction accuracy.

For the margin head, a smooth Jaccard loss has been utilized:

$$L_{\text{margin}} = 1 - \frac{\sum y_t y_p}{\sum y_t^2 + \sum y_p^2 - \sum y_t y_p}. \tag{2}$$

For mask segmentation task, a single head SpaghettiNet with hybrid loss is utilized. Our proposed hybrid loss function for mask segmentation model is defined as below:

$$L_{\text{mask}} = 1 - \underbrace{\frac{2\sum y_p y_t}{\sum y_p + \sum y_t}}_{\text{Dice}} - \underbrace{\frac{1}{N}\sum y_t \log(w_{\text{map}} y_p) + (1-y_t)\log(1-w_{\text{map}} y_p)}_{\text{Weighted Binary Cross Entropy (WBCE)}} \tag{3}$$

This loss comprises of Dice index and a weighted cross entropy loss function. The weight map in the equation (3) emphasizes the nuclei with small areas (areas less than 20 pixels).

It must be noted that we have trained three different forms of SpaghettiNet for both tasks. Model variations (normal, shallow, deep) are based on the different number of elements and their configuration used in the architecture.

Preprocessing:

Due to memory limitation of GPUs, we crop the images to 256x256 patches with overlaps (each image forms 25 patches). In order to have richer data, we convert the RGB image to HSV and La*b* color spaces, then we concatenate RGB with HSV and L channel to obtain a 7 channel input for the network. Moreover, all image channels have been rescaled to have pixel intensities between 0 and 1.

We used a large number of online augmentations to enhance the input data and make the network robust against variations. These augmentations are:

Horizontal flip, vertical flip, rotation, zooming, shearing, elastic deformation, color shift, contrast adjustment, adding non-uniformity to the illumination, scaling the intensity, sharpness adjustment, and adding noise. Note that all these augmentations have been randomly applied during training.

Post-Processing:

After acquiring outputs from different models predicting nuclei masks, markers, and margins; the first post-processing step is to ensemble outputs of different models. We simply average the related outputs from different models. The margin map is cleaned up by applying Frangi vesselness filter, and then used to further refine the marker map:

$$Marker_{\text{refined}} = Marker \times [1 - \text{frangiVesselness}(Margin)] \quad (4)$$

An optimum threshold is applied on the mask ($T_{\text{mask}} = 0.5$) and marker ($T_{\text{marker}} = 0.325$) outputs to convert them to binary maps (thresholds are obtained through a grid search process). Afterward, the binary markers are converted to label maps in which every nuclei marker has its own unique integer value. Finally, these markers are used as initial seeds in a marker-controlled watershed algorithm to separate each nuclei region in the binary map of masks.

Results:

Model evaluation has been done in a five-fold cross-validation framework. The average values of the Aggregate Jaccard Index (AJI), simple dice and instance based dice on all folds are considered as the evaluation metrics. In the table below the performance of proposed instance segmentation method with different SpaghettiNet variations and their ensemble are reported.

Table 1- Evaluation results on 5-fold cross-validation experiments on MoNuSeg2018 training dataset

Model	Simple dice	Instance-based dice	Aggregated Jaccard index
SpaghettiNet regular	0.817	0.72	0.635
SpaghettiNet shallow	0.803	0.71	0.633
SpaghettiNet deep	0.829	0.72	0.638
Ensemble	0.834	0.74	0.642

Mask-RCNN for Cell Instance Segmentation

Xuhua Ren[1]#, Sihang Zhou[2]#, Dinggang Shen[2] and Qian Wang[1]

[1] Institute for Medical Imaging Technology, School of Biomedical Engineering, Shanghai Jiao Tong University, Shanghai 200030, China
[2] Department of Radiology and BRIC, University of North Carolina at Chapel Hill, Chapel Hill, North Carolina 27599
renxuhua@sjtu.edu.cn

Abstract. We proposed an automatic nucleus segmentation algorithm of H&E stained tissue microscopy images. Mask-RCNN is a recently proposed state-of-the-art algorithm for object detection and object instance segmentation of natural images. In this paper, we demonstrate that Mask-RCNN can be used to perform highly effective and efficient automatic segmentation of H&E microscopy images for cell nuclei. We propose a novel MASK Non-maximum suppression (NMS) module which can automatically ensemble classifiers results and increase the robustness of model.

Keywords: Cell Segmentation, H&E stained, Mask-RCNN, Deep Learning

1 Introduction

Cell instance segmentation is an important task in medical image analysis involving cell level pathology analysis. In H&E stained microscopy images, this task is challenging for various reasons, for example the relatively large variation in the intensity of captured signal, and the ambiguity boundary information when separating neighboring cells. It requires careful model configurations to ensure its robustness to capture certain feature information of cells in the images such as intensity, shape, and size. Manual segmentation can be time consuming. In contrast, deep learning-based approaches have shown the great power of automatic extraction and selection of cell image features. In this paper, we present a novel Mask-RCNN [1] algorithm to solve the problem of cell segmentation in H&E stained microscopy images.

The Mask-RCNN model was developed in 2017, which is extended from the Faster-RCNN [2] model for semantic segmentation and object instance segmentation of images. Mask-RCNN has shown its effectiveness among all existing single-model entries on every task in the 2016 COCO Challenge. Mask-RCNN relies on a region proposal which are generated via a region proposal network. Mask-RCNN follows the Faster-RCNN model of a feature extractor followed by this region proposal network, then followed by an operation known as ROI-Pooling to produce standard-sized outputs suitable for input to a classifier, with two important modifications. First, Mask-RCNN replaces the imprecise ROI-Pooling operation with an operation called ROI-Align that allows very accurate instance segmentation masks to be constructed; Sec-

ond, Mask-RCNN adds a network head (a small fully convolutional neural network) to produce the instance segmentations; c.f. Figure 1.

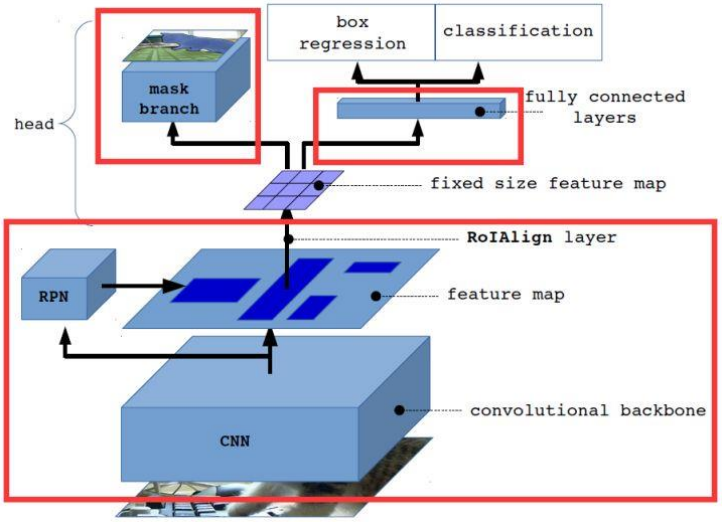


Fig. 1. The Mask-RCNN model.

2 Materia and Method

2.1 Dataset

The dataset for this challenge was obtained by carefully annotating tissue images of several patients with tumors of different organs and were diagnosed at different hospitals. The H&E stained tissue images were captured at 40x magnification from TCGA archive. H&E staining is a routine protocol to enhance the contrast of a tissue section and is commonly used for tumor assessment (grading, staging, etc.).

2.2 The Proposed Network

In this paper, we use a Mask-RCNN model with a ResNet-101[3] pyramid network backbone. It is based on an existing implementation by Matterport [1] and which is itself based on the open-source libraries Keras [4] and Tensorflow [5]. Rather than training the network end-to-end from the start, we initialize the model using weights obtained from pre-training on the MSCOCO dataset and proceed to train the layers in two stages: First, the model only trains the network heads, which are randomly initialized, then training the upper layers of the network, and then reducing the learning rate by a factor of 10 and training end to end. In total we trained 300 epochs using sto-

[1] https://github.com/matterport/Mask_RCNN

chastic gradient descent with momentum of 0.9. We use a batch size of 5 on a single NVIDIA Titan XP GPU.

Besides, we have implemented the following pre-processing works to ensure the validity of the experiments: We fill holes in masks by image morphology. We also split nuclei masks that are fused by applying morphological erosion and dilation. Zero mean unit variance normalization also utilized in each image. To help avoid overfitting, we implemented data augmentation using random crops, random rotations, gaussian blurring, random horizontal and vertical flips.

There are several changes made in the Mask-RCNN model to improve the segmentation performance: We reduced RPN (region proposal network) anchor sizes, since the nuclei are mostly small; Increased number of anchors to be used, since the nuclei are small and can be found anywhere on an image; Increased maximum number of predicted objects, since an image can contain 1000 or more nuclei. Increased POST_NMS_ROIS_TRAINING to get more region proposals during training. Added extra parameter DETECTION_MASK_THRESHOLD to model configuration. Cropped images and masks to 600x600.

Furthermore, to improve the robustness on segmenting cells with irregular shape (like, strip) and size (smaller than 6 pixels in radius), we introduce a multi-task U-Net like network [2]. The code of the network is based on the release version on github. Specifically, the backbone of the u-net-like network is a pre-trained ResNet 101. Three tasks are conducted in the network. In the first task, we do the common segmentation, and try to segment all the fore-ground regions. In the second task, we segment the adjacent boundaries of between cells. In the third task, we conduct erosion operation to the instance masks and construct the interior portions of each cell, then do the segmentation on these interiors. The three tasks share the same back bone but different decoders. In the testing time, the foreground segmentation, adjacent boundaries and the interior of cells are inputted to a watershed algorithm for final segmentation.

For post-processing, we combined predictions from five-fold cross training models: took unions of masks with maximum overlap and removed false positive masks with small overlap. We called this module as MASK-NMS. Moreover, our NMS starts with a list of segmentation results I with scores S. After selecting the segmentation with the maximum score M, it removes it from the set I and appends it to the final segmentation set D. It also removes any segmentations which has an overlap greater than the threshold N_t compared with M in the set I, we address intersection over union (IOU) as overlap metric. This process would repeat until set I become empty. This module could ensemble multi-models results together and reduce false positive or false negative situation.

We also tried some unsuccessful strategies. E.g., Trained the model with Dice Coefficient Loss instead of default binary cross-entropy loss for the masks heads. Trained with random Gaussian noise for image augmentation. It hurt overall model performance. Tried assembling actual image predictions with horizontal and vertical flip predictions. Used non-maximum suppression for removing overlaps. Did not

[2] https://github.com/samuelschen/DSB2018

improve prediction accuracy on the validation set. Trained end-to-end without initializing with pre-trained ImageNet weights. It would be worse than pretrained model. Trained on preprocessed images with adaptive histogram equalization (CLAHE). The model performed way worse. Trained with mask dilations and erosions, this did not have any improvement in the segmentation in my experiments. And soft-NMS did not improve accuracy also.

There are some strategies that we didn't have time to try. E.g., Hyperparameter search on thresholds and network architecture. Different layer normalization techniques, with batch size more than one image at a time. Augmentation smoothing during training, i.e., Increase the noise and augmentation slowly during the training phase.

3 Experimental Results

The proposed Mask-RCNN model with ResNet-101 backbone obtains an average mask intersection over union (IOU) of 45.02% on the five-fold validation dataset. The image with nuclei detections and segmentations are illustrated in Figure 2.

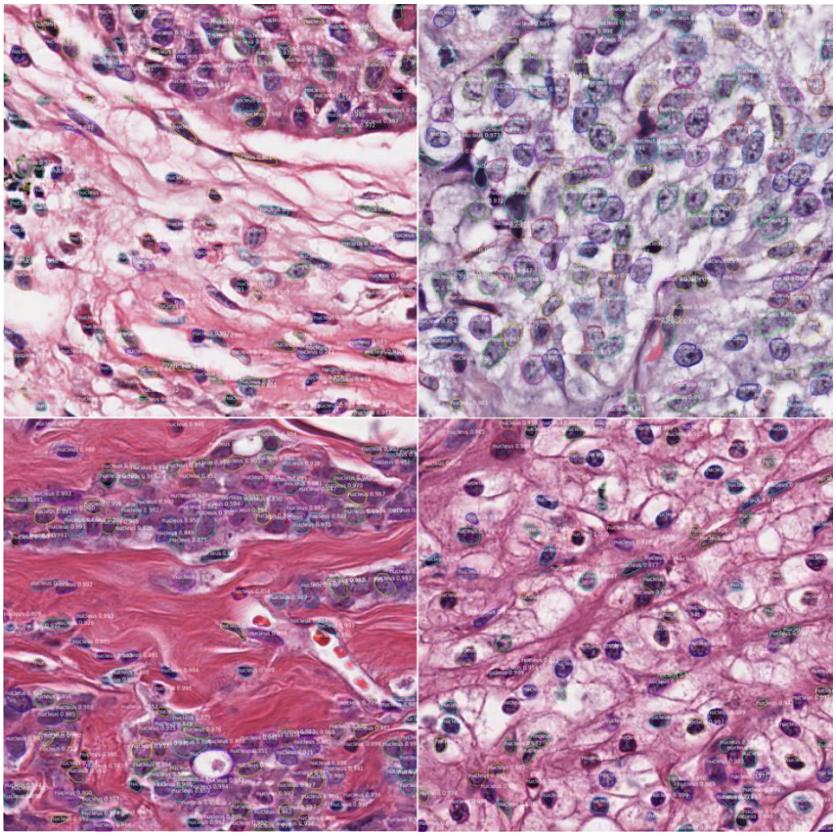


Fig. 2. Segmentation result in validation dataset.

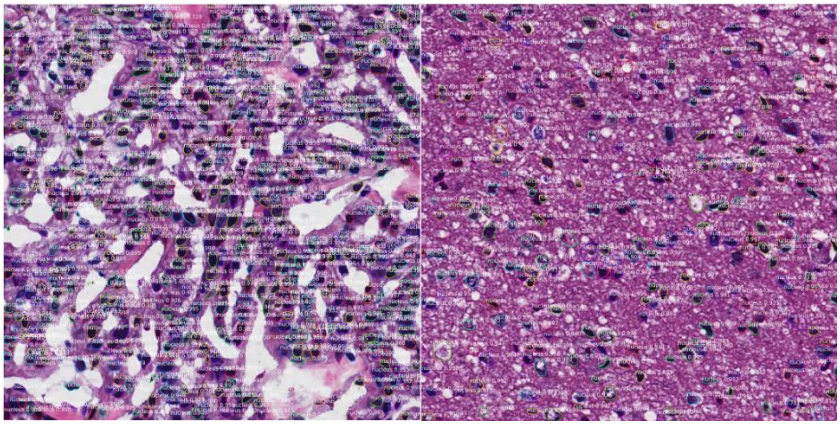


Fig. 3. Segmentation result in test dataset.

Conclusion

Cell segmentation is an important step for cell-level analysis of biomedical images. In this paper we demonstrate that the Mask-RCNN model, can be used to produce high quality results for the challenging task of segmentation of nuclei. We also designed a MASK-NMS for model ensemble which could reduce false positive or false negative situation.

References

1. He, K., Gkioxari, G., Dollár, P., Girshick, R.: Mask r-cnn. In: Computer Vision (ICCV), 2017 IEEE International Conference on, pp. 2980-2988. IEEE, (Year)
2. Girshick, R.: Fast r-cnn. In: Proceedings of the IEEE international conference on computer vision, pp. 1440-1448. (Year)
3. He, K., Zhang, X., Ren, S., Sun, J.: Deep residual learning for image recognition. In: Proceedings of the IEEE conference on computer vision and pattern recognition, pp. 770-778. (Year)
4. Chollet, F.: Keras. (2015)
5. Abadi, M., Agarwal, A., Barham, P., Brevdo, E., Chen, Z., Citro, C., Corrado, G.S., Davis, A., Dean, J., Devin, M.: Tensorflow: Large-scale machine learning on heterogeneous distributed systems. arXiv preprint arXiv:1603.04467 (2016)

PANet for Automatic Nuclear Segmentation

Rank- 7, aetherAI- MoNuSeg
Cheng-Kun Yang, Chi-Hung Weng, Wei-Hsiang Yu, and Chao-Yuan Yeh
aetherAI
{jimmy,chihung,seanyu,joeyeh}@aetherai.com

In this work, we tested Path Aggregation Network (PANet) and its predecessor, Mask-RCNN, for their performance on nuclear segmentation. Several modifications to these models were made to achieve better performance, including (1) the use of different feature-extractor backbones and (2) replacement of batch normalization (BN) with group normalization (GN).

1. **Pre-Processing**
    a. Color normalization
    NOT APPLICABLE.
    b. Intensity Transformation
    NOT APPLICABLE.
    c. Data Augmentation
    Applied methods of augmentation are: random image translation, random scaling, random flipping, random rotation (one of 0/90/180/270 degrees) and random RGB channel shifting. These methods will be applied to randomly-cropped images of size 512×512.
    d. Other pre-processing steps
    NOT APPLICABLE.

2. **Model details**
    a. CNN architecture
    PANet[1] is the first-place winner of 2017 COCO instance segmentation challenge. It has made some effective improvements to Mask-RCNN[2], a state-of-art method that originated from the R-CNN family. Improvements made in PANet include: (1) the creation of an additional bottom-up path directly after the Feature Pyramid Network (FPN)[3] and (2) a proposed Region of Interest (RoI) will be extracted from several spatial scales of feature maps. For the former, extra skip-connections introduced by the new path enables better information flow of the overall network. For the latter, aggregating feature maps from different spatial scales potentially increase the quality of predictions. PANet (and Mask-RCNN) is consisted of three building blocks:
        i. *Feature extractor*. In this work, FPN was used and two backbone models of FPN were experimented, i.e. ResNet101 and DenseNet169. These two models were pre-trained with the ImageNet dataset and can be retrieved from popular Deep-Learning frameworks such as Keras.

ii. *Region Proposal Network* (RPN). This shallow network is responsible for the generation of RoIs.
iii. *Sub-network after RPN*. This sub-network generates predictions of masks, classes and bounding boxes for the proposed RoIs.

b. Loss Function

There are five loss functions in PANet (or Mask-RCNN):

i. $L_{RPN-box}$: smooth L1 loss of RPN, used for bounding box regression.
ii. $L_{RPN-class}$: cross-entropy loss of RPN, used for foreground/ background classification.
iii. L_{mask} : cross-entropy loss, used for predictions of binary masks.
iv. L_{box} : smooth L1 loss, used for bounding box regression.
v. L_{class} : cross-entropy loss, used for object classification.

These loss functions are added together and the network is trained in an end-to-end manner.

c. Hyper-parameter settings (training)

During the whole training process, weight decay and momentum of SGD are fixed to 10^{-5} and 0.9, respectively. We use two GPUs for training and the batch size is set to 2 per GPU. The training is split into two phases:

i. Warm-up phase: pre-trained weights of the backbone model (ResNet101 or DenseNet169) were loaded. During this phase, the pre-trained weights were frozen and the other weights were updated via SGD. The learning rate of SGD was set to 10^{-3}. This phase took 20 epochs.
ii. Full-training phase: we unfroze weights of the backbone model so that this part of the network can be further fine-tuned. The learning rate of SGD was set to 10^{-4} and was dropped by half in the middle of this phase. This phase took 80 epochs.

d. Hyper-parameter settings (model)

The width (W) and height (H) of anchor boxes of RPN can be controlled via $W = \alpha\sqrt{\gamma}$ and $H = \alpha\sqrt{1/\gamma}$, where α and γ are size and aspect ratio of the anchor boxes, respectively.

To increase the probability of matching anchor boxes to GT boxes, α and γ have to be chosen in accordance with possible sizes and aspect ratios of nuclei. In this work, we use $\alpha \in \{12,24,36,48,72\}$ and $\gamma \in \{0.5,1,2\}$ for generation of anchor boxes.

e. Other innovations

Due to the hardware constraint, the maximum batch size per GPU we can set is 2, which is small and inappropriate for the use of BN. In this work, we replace all the BN[4] layers with GN[5] layers, as GN is known to perform well even at small batch sizes. Here, we use 32 groups to calculate GN.

3. **Post-processing**
   NOT APPLICABLE.

4. **Computational Complexity**
   a. Hardware
      We use 2 GPUs (NVIDIA Tesla V100; 32GB RAM/GPU) for training.
   b. Training time
      8 hours and 19 mins (100 epochs).
   c. Testing time
      17 minutes and 34 seconds for 14 images of size 1024×1024 (pre-processing included).

5. **Code Release**
   NOT APPLICABLE.

---

[1] S. Liu, L. Qi, H. Qin, J. Shi, and J. Jia, in The IEEE Conference on Computer Vision and Pattern Recognition (CVPR) (2018).
[2] K. He, G. Gkioxari, P. Dollr, and R. Girshick, in 2017 IEEE International Conference on Computer Vision (ICCV) (2017) pp. 2980–2988.
[3] T. Y. Lin, P. Dollr, R. Girshick, K. He, B. Hariharan, and S. Be- longie, in 2017 IEEE Conference on Computer Vision and Pat- tern Recognition (CVPR) (2017) pp. 936–94.
[4] S. Ioffe and C. Szegedy, in Proceedings of the 32nd International Conference on Machine Learning, ICML 2015, Lille, France, 6- 11 July 2015 (2015) pp. 448–456.
[5] Y. Wu and K. He, in The European Conference on Computer Vision (ECCV) (2018).

Path Aggregation Network for Multi-Organ Nuclei Segmentation

Rank- 8, Team Name- Shuang Yang

Shuang Yang[1]

[1]Zhejiang University

1. **Pre-Processing**
    a. Data Augmentation. The data augmentation includes scaling, shift, up/down and right/left flip, rotation of 90/180/270 degrees, illumination, filter/noise. Considering the big size of the image, 512 * 512 patches are randomly cropped from the original images as the input of the proposed model.
    b. We also fill holes (if any) in all of the provided ground-truth masks to reduce the error of labeling.we keep images corresponding to 16 patients equally divided among four organs – breast, kidney, liver, and prostate – in the training and validation set. We have divided rest of the images into two test sets - same organ test set and different organ test set.

2. **Model details**
    a. We employ a modified Path Aggregation Network (PANet) [1] which is improved from Mask R-CNN [2] aiming at boosting information flow in proposal-based instance segmentation framework, the structure of PANet is shown in Figure 1. PANet enhance the entire feature hierarchy with accurate localization signals in lower layers by bottom-up path augmentation, which shortens the information path between lower layers and topmost feature. Also at the bottom of FPN, we add a mask branch for supervision. In this network, adaptive feature pooling which links feature grid and all feature levels is adopted to make useful information in each feature level propagate directly to following proposal subnetworks. Furthermore, a complementary branch capturing different views for each proposal is created to further improve mask prediction, it yields a little improvement in terms of mask AP. We use Resnet50 and Resnet101 as a backbone encoder.

b. Loss Function. We used a combination of cross-entropy loss and dice coefficient for the mask branch of FPN. The other loss functions are the same as Mask RCNN.
c. Hyper-parameter settings. We take 2 images in one batch for training. SGD optimizer with a learning rate of 1e-3 is applied to optimize the parameters.

3. **Post-processing**- First, we combine predictions on actual image and horizontally flipped image: take unions of masks with maximum overlap and remove false positive masks with small overlap. Then, overlaps between predicted nuclei are removed based on their objectness score. In other words, we remove intersections from the masks with lower scores. If this intersection removal results in multiple objects in that mask, then removing all the small pieces. Finally, we close small holes inside the masks using morphological operations (dilation followed by erosion).

4. **Computational Complexity**
   a. Hardware. We used a computer with one Intel i7-6557 hexa-core processor with 32 GB RAM and 8 NVIDIA 1080ti graphics cards with 11 GB memory."
   b. Training time- About 20 Hours per model.
   c. Testing time- About 2.5s for a 1000x1000 image for end-to-end processing.

5. **Code Release**- Maybe in the future.

References

[1] Liu S, Qi L, Qin H, et al. Path aggregation network for instance segmentation[C] Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. 2018: 8759-8768.

[2] He K, Gkioxari G, Dollár P, et al. Mask r-cnn[C] Computer Vision (ICCV), 2017 IEEE International Conference on. IEEE, 2017: 2980-2988.

[3] Kumar, Neeraj, et al. "A Dataset and a Technique for Generalized Nuclear Segmentation for Computational Pathology." *IEEE Transactions on Medical Imaging* 36.7(2017):1550-1560.

Multi-Organ Nuclei Segmentation Using Mask-RCNN with Test Image Augmentation

Rank- 9, Team Name- Bio-totem&SYSUCC
Pak Hei YEUNG[1,3], Peng Sun[2], Shuoyu Xu[2,3]
[1]The University of Hong Kong, [2] Sun Yat-Sen University Cancer Center, [3]Bio-totem Pte Ltd

1. Pre-Processing

For each image, 25 250x250 patches were obtained. A variety of data augmentations, including random horizontal and vertical flipping, rotation (-10 °, +10 °) and resizing (0.8 to 1.2 of the original size), were implemented to each patch during training. In addition, due to the staining variation among different organs and patients, we further implemented color augmentation proposed in [1]. Each channel of every training patch was multiplied by a random constant falling between 0.3 to 1.5. This random modification of color space may enhance the robustness of segmentation when staining variation exists. No staining normalization is needed after implementing the color augmentation.

2. Model details

The dataset was divided into training set (23 images) and validation set (7 images). The exact grouping is presented in Table 1. Mask R-CNN [2] was used and ResNet [3] network of depth 101 layers pretrained with COCO was adopted as the convolutional backbone architecture. The network head, which functions as bounding-box recognition, was trained for 20 epochs. Then, the whole network was trained for another 80 epochs, with a learning rate of 0.001 and a weight decay of 0.0001. Due to the small size of nucleus in the WSI, we modified the Region Proposal Network anchors area scale to be $\{8^2, 16^2, 32^2, 64^2, 128^2\}$ pixels [4]. The batch size is 2. The other hyper-parameters followed the settings of [2]. The output of the network was every segmented nucleus with the corresponding instance nucleus probability.

3. Post-processing-

We implemented test image augmentation for generating multiple segmentation masks for every nucleus detected. The augmentation included horizontal and vertical flipping, 90^0 rotation, resizing (0.8 and 1.2 of the original size) and color channel multiplication. All nuclei from the masks were ranked by their instance nucleus probability. If two nuclei overlapped and the overlapping area was larger than half of the area of either nucleus, only the one with higher instance nucleus probability would be preserved. This selection process continued until all nuclei with probability larger than 0.5 were processed. Test image augmentation with the above selection process increases the chance of potential nucleus being detected and segmented. However, this may result in false positive segmentation. Therefore, a probability map was generated by summing up all the segmentation masks with the corresponding instance nucleus probability being multiplied. In this map, region with higher value means that nuclei are detected there in multiple augmented images, which suggests that nuclei are more likely to be there. Any nucleus that is selected in the previous

process but locates in region with value smaller than 3 in the probability map would be eliminated. This may rule out some potential false positive nuclei. Finally, any nucleus with average intensity larger than 220 were filtered out to eliminate detected nucleus that appeared to be 'white', which is obviously false positive detection. The remaining segmented nuclei would be the final segmentation result.

4. **Computational Complexity**
    a. The training was performed on Amazon AWS cloud with a p2.xlarge instance with 2.3 GHz Intel Xeon E5-2686 v4 Processor and one NVIDIA K80 graphics card with 12 GB memory.
    b. The testing was performed on a local computer with one Intel I7-6800K processor with 32 GB RAM and NVIDIA GTX 1080Ti graphic card with 11GB memory.
    c. The entire training time is around 6 hours.
    d. Testing time required for a 1000x1000 image is around 1800 seconds with test augmentation or 600 seconds without test augmentation.

5. **Code Release**- NOT APPLICABLE

Table 1

	Training Set			Validation Set		
Organ	Name	Original Image	Test Image Augmentation	Name	Original Image	Test Image Augmentation
Kidney	TCGA-B0-5711-01Z-00-DX1	**0.679**	0.664	TCGA-B0-5698-01Z-00-DX1	0.587	**0.592**
	TCGA-HE-7128-01Z-00-DX1	**0.657**	0.650	TCGA-HE-7130-01Z-00-DX1	0.553	**0.566**
	TCGA-HE-7129-01Z-00-DX1	0.538	**0.546**			
	TCGA-B0-5710-01Z-00-DX1	**0.688**	0.681			
Breast	TCGA-A7-A13E-01Z-00-DX1	**0.660**	0.653	TCGA-E2-A14V-01Z-00-DX1	0.664	**0.691**
	TCGA-A7-A13F-01Z-00-DX1	0.660	**0.678**			
	TCGA-AR-A1AK-01Z-00-DX1	0.627	**0.630**			
	TCGA-AR-A1AS-01Z-00-DX1	0.712	**0.721**			
	TCGA-E2-A1B5-01Z-00-DX1	**0.633**	0.615			
Liver	TCGA-18-5592-01Z-00-DX1	**0.716**	0.713	TCGA-49-4488-01Z-00-DX1	0.631	**0.633**
	TCGA-38-6178-01Z-00-DX1	0.566	**0.569**			
	TCGA-50-5931-01Z-00-DX1	**0.585**	0.571			
	TCGA-21-5784-01Z-00-DX1	**0.611**	0.606			
	TCGA-21-5786-01Z-00-DX1	**0.595**	0.591			
Prostate	TCGA-G9-6336-01Z-00-DX1	**0.700**	0.691	TCGA-G9-6348-01Z-00-DX1	**0.678**	**0.678**
	TCGA-G9-6356-01Z-00-DX1	**0.746**	0.744			
	TCGA-G9-6363-01Z-00-DX1	**0.701**	0.692			

	TCGA-CH-5767-01Z-00-DX1	0.602	**0.612**			
	TCGA-G9-6362-01Z-00-DX1	0.672	**0.676**			
Colon	TCGA-NH-A8F7-01A-01-TS1	**0.645**	0.642	TCGA-AY-A8YK-01A-01-TS1	0.524	**0.541**
Bladder	TCGA-DK-A2I6-01A-01-TS1	**0.760**	0.750	TCGA-G2-A2EK-01A-02-TSB	0.574	**0.599**
Stomach	TCGA-KB-A93J-01A-01-TS1	**0.769**	0.764			
	TCGA-RD-A8N9-01A-01-TS1	**0.779**	0.771			
Overall AJI Score		**0.669**	0.665		0.597	**0.610**

* AJI score was calculated slightly differently when compared to the MATLAB code provided. If two nuclei are overlapping, they would be saved in two different 'frames' to ensure that the overlapping region would not be just assigned to one nucleus. This matches with the Algorithm 1 presented in 'A Dataset and a Technique for Generalized Nuclear Segmentation for Computational Pathology'.

[1] A. Janowczyk. (2018). *ON STAIN NORMALIZATION IN DEEP LEARNING*. Available: http://www.andrewjanowczyk.com/on-stain-normalization-in-deep-learning/

[2] K. He, G. Gkioxari, P. Dollár, and R. Girshick, "Mask r-cnn," in *Computer Vision (ICCV), 2017 IEEE International Conference on*, 2017, pp. 2980-2988: IEEE.

[3] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2016, pp. 770-778.

[4] T.-Y. Lin, P. Dollár, R. Girshick, K. He, B. Hariharan, and S. Belongie, "Feature pyramid networks for object detection," in *CVPR*, 2017, vol. 1, no. 2, p. 4.

A Two-Stage U-Net Algorithm for Segmentation of Nuclei in H&E-Stained Tissues

Rank-10, Amirreza Mahbod-MoNuSeg

Amirreza Mahbod*‡, Gerald Schaefer†, Isabella Ellinger*, Rupert Ecker‡, Örjan Smedby§ and Chunliang Wang§
*Institute for Pathophysiology and Allergy Research, Medical University of Vienna, Vienna, Austria
†Department of Computer Science, Loughborough University, Loughborough, United Kingdom
‡Department of Research and Development, TissueGnostics GmbH, Vienna, Austria
§Department of Biomedical Engineering and Health Systems, KTH Royal Institute of Technology, Stockholm, Sweden

I. Method

A generic workflow of our proposed algorithm is shown in Fig. 1. In the following, we describe the details of the algorithms.

*1) **Pre-processing**:*

(a) Color normalization: in order to decrease the staining variability among the images, we used a color normalization technique based on Macenko et al. method [1]. By histogram analysis of all training samples, we chose an appropriate reference image and then normalize the color separation vectors of all other training and test images.

(b) Intensity transformation: all images were normalized to have intensity values in $[0; 1]$ range.

(c) Data augmentation: we applied color perturbation as suggested in [2] to deal with existing color variation in the images. We converted the image intensities from the RGB space to HSV space and then randomly changed the values of each channel in the intervals of $[H \cdot 0.96; H \cdot 1.04]$, $[S \cdot 0.8; S \cdot 1.25]$ and $[V \cdot 0.96; V \cdot 1.04]$. The images were then converted back to the RGB space. Beside the color augmentation, we also applied standard random rotation (0, 90, 180 and 270 degrees) and random horizontal flipping as additional augmentation techniques.

(d) Other pre-processing steps: all images (raw input images and the corresponding ground truth) were resized to 1024×1024 to fit in the utilized deep models. Moreover, we modified the binary ground truth (GT) masks to have better instance segmentation performance. To alter the GT, we removed all touching borders from the masks and then applied an erosion operation to slightly shrink each nucleus.

*2) **Model details**:*

(a) CNN architecture: we trained two encoder-decoder-based models inspired by the U-Net architecture [3] in two sequential stages. In the first stage, a standard U-Net model (referred to as segmentation U-Net in Fig. 1) was trained from scratch. In the second phase (referred to as distance U-Net in Fig. 1), we trained another U-Net based on the distance map of the provided ground truth to find the candidates of all nuclei. The network structures were similar in both stages except the loss function as we aimed to perform semantic segmentation in the first stage, but dealing with a regression task in the second stage. We used the results from both stages to generate final instance segmentation masks in an automatic way. Initially, we calculated the mean average nuclei size from the segmentation U-Net predications in the first stage. Then we applied a Gaussian smoothing filter on the distance map predictions. The Kernel size of the Gaussian filters was determined by the derived average nuclei sizes in the first stage. Finally, we found the local maxima from the filtered predicted distance maps and used them as seed points for a marker-controlled watershed method [4]. Thus, by training both stages in an end-to-end manner, the instance segmentation was performed in a fully automatic approach.

(b) Loss function: for the segmentation U-Net, we used combination of binary cross-entropy (bce) and binary Dice loss to train the model as follows:

$$Loss_{total} = 0.5 Loss_{bce} + Loss_{Dice} \quad (1)$$

For the distance U-Net, as we aimed to solve a regression task in this phase, we chose a standard mean square error loss function to predict the distance maps of each individual nucleus.

(c) Hyper-parameter setting: we used the U-Net-based architecture in both models with four max pooling layers in the encoder part and four transpose convolutional layers in the decoder part. After first convolutional layer in each abstraction level of either encoder or decoder section, a 10% dropout layer was added to the networks. Adam optimizer [5] was utilized in both phases and the batch size was set to 2 in the training phase. To train the segmentation U-Net, we used an initial learning rate of 0.001 which was dropped by a factor of 0.4 after 80-th epochs. We trained the first model for 240 epochs. For the distance U-Net, a similar initial learning rate was chosen, but the it was dropped at 60-th epoch and the model was trained for 120 epochs. In both phases, the

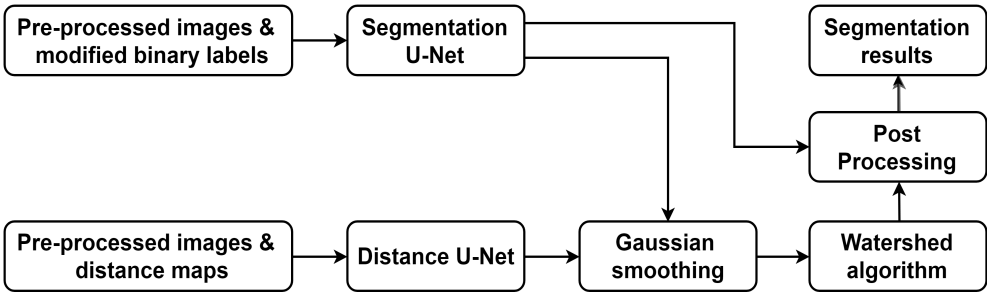


Fig. 1: Generic workflow of the proposed method.

networks were trained with full resolution resized images (i.e. 1024×1024) and produced the segmentation mask or distance maps with the same size.

(d) Other innovation: our idea to utilize distance map for separating touching nuclei was developed in parallel to [6]. However, unlike [6], where a single regression network was used and final instance segmentation results were generated by finding an optimized threshold for merging the local maxima, in our approach we used the results from both stages to perform nuclei segmentation as described in part a in a fully automatic manner.

*3) **Post-processing**:* We applied 4 post-processing steps as following:

(a) For background identification, the results from the first stage were utilized to detect all background pixels as indicated in Fig. 1.

(b) Very small detected objects with an area of less than 20 pixels were removed from the segmentation mask.

(c) Any holes inside the detected nuclei were filled by applying morphological operations.

(d) Finally the generated segmentation masks were resized back to 1000×1000 for quantitative evaluation of the results.

*4) **Computational Complexity**:*

(a) Hardware: we used a computer with an Intel Corei5-6600k 3.50 GHz CPU with 16 GB RAM and NVIDIA GTX 1070 graphics card with 8 GB memory.

(b) Training time: training both stages took around 15 hours.

(c) Testing time: the inference time for each individual test image depended on the number of detected cells. On average generating a segmentation mask for a test image took around 38 seconds including all pre-processing and post-processing steps.

*5) **Code Release**:* NOT APPLICABLE

REFERENCES

[1] M. Macenko, M. Niethammer, J. S. Marron, D. Borland, J. T. Woosley, X. Guan, C. Schmitt, and N. E. Thomas, "A method for normalizing histology slides for quantitative analysis," in *IEEE International Symposium on Biomedical Imaging: From Nano to Macro*, 2009, pp. 1107–1110.

[2] I. Arvidsson, N. C. Overgaard, F.-E. Marginean, A. Krzyzanowska, A. Bjartell, K. Åström, and A. Heyden, "Generalization of prostate cancer classification for multiple sites using deep learning," in *15th IEEE International Symposium on Biomedical Imaging*, 2018, pp. 191–194.

[3] 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*, 2015, pp. 234–241.

[4] X. Yang, H. Li, and X. Zhou, "Nuclei segmentation using marker-controlled watershed, tracking using mean-shift, and Kalman filter in time-lapse microscopy," *IEEE Transactions on Circuits and Systems I: Regular Papers*, vol. 53, no. 11, pp. 2405–2414, 2006.

[5] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," in *3rd International Conference for Learning Representations*, 2015.

[6] P. Naylor, M. Laé, F. Reyal, and T. Walter, "Segmentation of nuclei in histopathology images by deep regression of the distance map," *IEEE Transactions on Medical Imaging*, 2018.

U-Net with Residual Blocks and Nuclear Training Augmentation for Nuclear Segmentation

That-Vinh Ton and Benjamin Chidester

1. Method

Our method for the MoNuSeg challenge is a U-Net, modified with residual blocks, enhanced with morphological post-processing steps, and aided with intelligent data augmentation.

1.1. Data augmentation by nuclear deformation

To increase the amount of training data and thereby aid the network in generalization, we created augmented images from the originals by slightly deforming and translating nuclei. Nuclear pixels are first removed and the resulting void is synthetically reconstructed using the inpainting method proposed by [1]. Each extracted nucleus is deformed by affine, spline, and elastic transformations, yielding a new orientation and shape. They are then overlaid on the reconstructed background, with a possible translation of up to 5 pixels. Finally, added noise, blurring, and illumination change are applied. Fig. 1 shows examples of new data generated by this method which are used for training.

1.2. Data pre-processing

We use the method proposed in [2] to reduce the color variation in H&E stained images. One image is used as the target and all other images including the new augmented data are converted to its color space. We use $\lambda = 0.1$ as recommended in [2]. The last column in figure 1 visualizes example normalized images.

1.3. U-Net architecture

Fig. 3 visualizes our neural network architecture inspired by the U-Net [3]. We formulate segmentation as a three-class problem, for which U-Net will produce a probability map for each label, namely, nuclear interior, nuclear boundary, and background. Residual blocks [4] are used to increase the depth of network. In [5], some experiments evaluated the performance of different types of residual blocks. We inherit from their work the architecture producing the best performance mentioned in [5]. Residual blocks are also used in long skip connections of U-Net. We hypothesized that low-level features are especially important for capturing the boundary class, so we added more residual blocks at this level to extract a richer set of features.

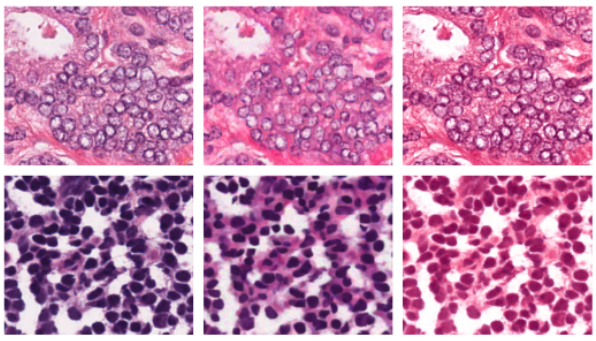


Figure 1: Data augmentation and data pre-processing. The first column is the input. The second column is an example augmented image. The last column is the image normalized by [2].

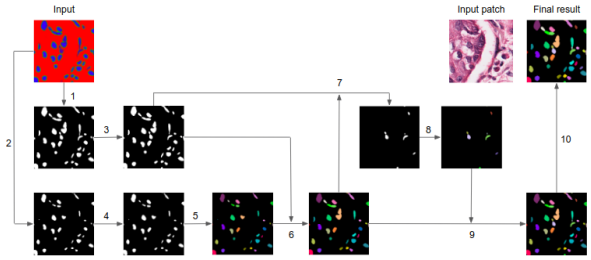


Figure 2: Our proposed post-processing method. Each step is visualized by its result.

1.4. Post-processing

To aid the network in learning invariance to rotations and flips, each image is manipulated by rotations of 90 degrees and vertical flipping and the result of the network for all eight orientations is averaged to yield a single map indicating the probability of each pixel belonging to each label.

We then apply morphological operations to finalize the segmentation result. Following the numbering in Fig. 2, which shows the post processing method: a mask of confident interior pixels is created (1) of pixels for which the inside probability is greater than other labels, followed by

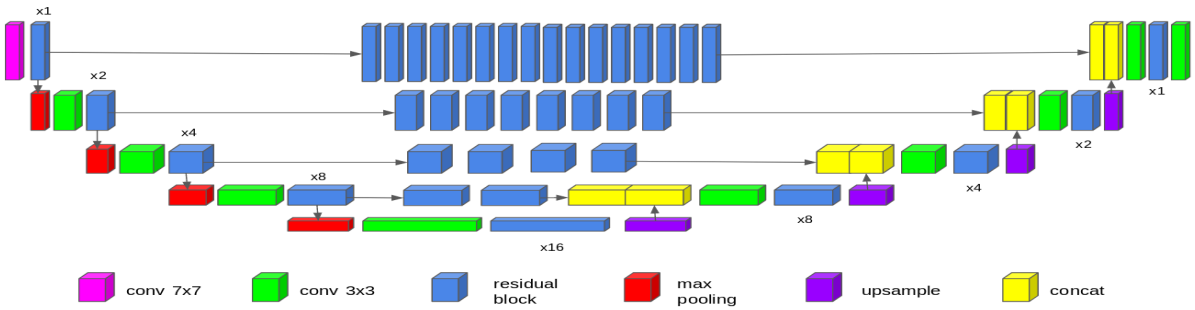


Figure 3: Our proposed architecture. More residual blocks [4] are added on long skip connections of U-Net [3] to extract richer features in low-level layers.

Model	AJI Training set	AJI Validation set	AJI Overall
Ours	**0.6461**	**0.6114**	**0.6380**
Ours - R.S	0.6282	0.5870	0.6186
Ours - D.A	0.5875	0.5681	0.5830

Table 1: Comparison between different model. -R.S: without residual blocks on long skip connections. -D.A: without new data augmentation.

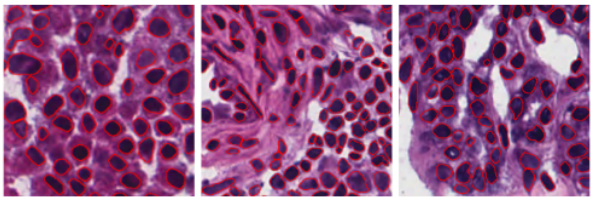


Figure 4: Our segmentation results on example test images. The estimated nuclear boundary is visualized in red.

morphological opening (3); a map of nuclear seeds is created (2) by thresholding ($thres = 0.85$) the probability of the interior class of these pixels, followed by opening (4); each seed is assigned a unique index (5), which is propagated to all connected interior pixels (6); regions not covered (7) will be assigned new index (8) and combined with the previous result (9); then, we apply binary dilation to create the final result (10).

2. Experiments & Results

2.1. Experimental setup

We extract patches of size 256×256 from augmented, pre-processed images for training data. For validation, we use a subset of the original images, one of each organ, leaving the other 23 images for training. Rotation and scaling are used when extracting these patches.

The network is trained with the cross entropy loss plus generalized dice loss [6]. The weights of network are initialized by Xavier initialization [7]. Adam [8] is used for optimization. We trained this network about 410k steps with a batch size of four. We used the aggregated Jaccard index (AJI) [9] to evaluate our model's performance.

2.2. Experimental results

We performed some ablation experiments to see how each component affects the final results. As can be seen in Table 1, with both residual blocks on long skip connections and new data augmentation, our network achieves 63.80% accuracy. Without residual blocks, the accuracy drops by around 2% to 61.86%, while the data augmentation contributes about 5.5% in accuracy. Figure 4 visualizes some results from our method on test data.

References

[1] J. Yu, Z. Lin, J. Yang, X. Shen, X. Lu and T. S. Huang, *arXiv:1801.07892* (2018).

[2] A. Vahadane, T. Peng, S. Albarqouni, M. Baust, K. Steiger, A. M. Schlitter, A. Sethi, I. Esposito and N. Navab, Structure-preserved color normalization for histological images, in *ISBI*, April 2015.

[3] O. Ronneberger, P. Fischer and T. Brox, *CoRR* (2015).

[4] K. He, X. Zhang, S. Ren and J. Sun, *CoRR* (2015).

[5] D. Han, J. Kim and J. Kim, *CoRR* (2016).

[6] C. H. Sudre, W. Li, T. Vercauteren, S. Ourselin and M. J. Cardoso, *CoRR* (2017).

[7] X. Glorot and Y. Bengio, Understanding the difficulty of training deep feedforward neural networks, in *AISTATS*, (PMLR, 13–15 May 2010).

[8] D. P. Kingma and J. Ba, *CoRR* (2014).

[9] N. Kumar, R. Verma, S. Sharma, S. Bhargava, A. Vahadane and A. Sethi, *IEEE Transactions on Medical Imaging* **36**, 1550 (July 2017).

Unifying by Distance -
An Ensemble Approach for Nuclear Segmentation in Histology Images

Rank- 12, Team Name- Graham&Vu
Simon Graham[1*], Quoc Dang Vu[2*], Jin Tae Kwak[2+] and Nasir Rajpoot[1+]
[1]University of Warwick, [2]Sejong University
[*]Equal contribution of first authors, [+]Equal contribution of last authors

1. Methodology Details

In a simplistic view, the nuclear instance segmentation task consists of 1) separating nuclei from the background and then 2) separating the instances from each cluster. We propose two distinct methods for nuclear segmentation. The first method uses a region proposal approach via a variant of the Mask-RCNN architecture [1], where the above two steps are performed at the same time. The second method follows a sequential approach, where each step is solved separately using a fully convolutional neural network, named ED-RD-Net, and is later combined to get the final prediction. We then combine their strength using an ensemble method for a more robust and accurate result. Each method produces two outputs: (i) a nuclear probability map and (ii) a nuclear distance map. These two distance maps are then unified, leading to a refined energy landscape for optimal seed point detection and watershed. We summarise the entire processing pipeline in **Figure 1**.

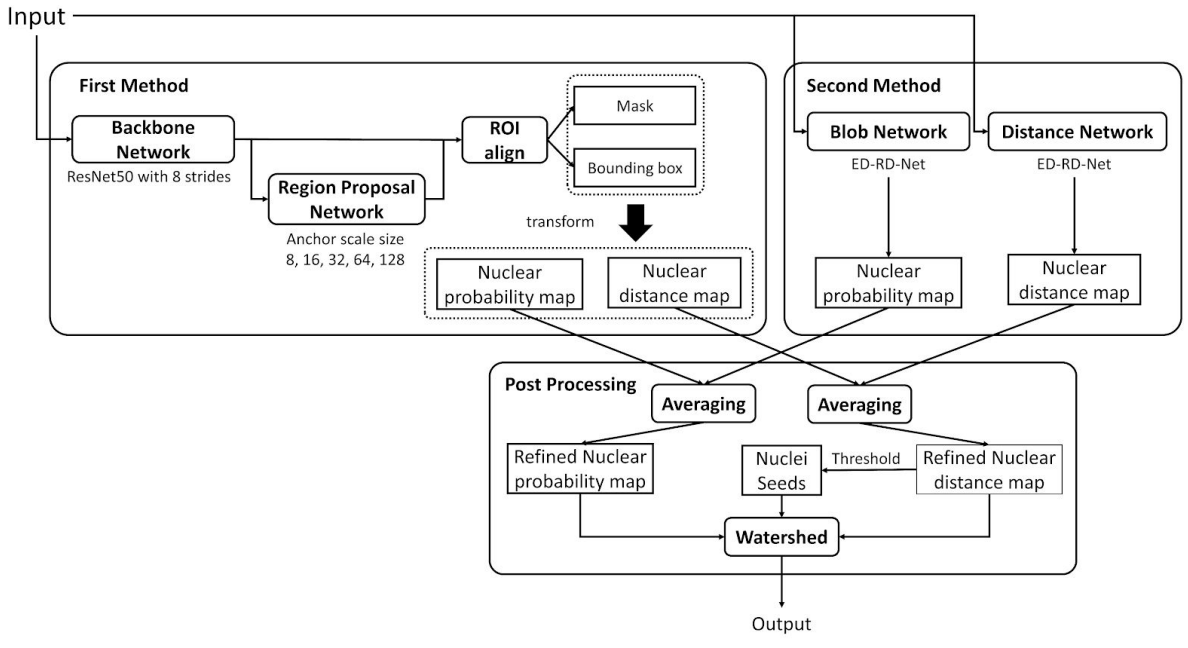


Figure 1. Overview of the entire processing pipeline.

a. First Method - Concurrent Processing with Mask-RCNN

A variant of the Mask-RCNN architecture is adopted, we use a ResNet50 [7] model for the encoder so that it has an output stride of 8 for dense feature extraction. We also utilise the feature pyramid network [2] to incorporate multi-scale information and empower the detection of nuclei with varying

size. For accurate nuclei detection, we use anchor scales of sizes 8, 16, 32, 64 and 128. Using a region proposal based network enables us to achieve a good instance segmentation, especially for areas with clustered nuclei.

Loss Function.

The Mask-RCNN approach optimised multiple loss terms, corresponding to the different branches of the network architecture. Overall, the loss function is defined as:

$$L = L_{rpn} + L_{mask} + L_{class}$$

Where L_{rpn} is the loss term of the region proposal network, L_{mask} is the loss term of the mask branch and L_{class} is the loss term for the classification branch. All loss terms are optimised together with equal weighting of each term. L_{rpn} and L_{class} both consist of a regression loss and a classification loss. The regression loss utilises a smooth L_1 loss to regress the bounding box coordinates, whilst the the classification loss utilises the cross-entropy loss. L_{mask} denotes the average pixel-based binary cross entropy loss.

Hyper-parameter settings.

We initialised the model with pre-trained weights on the Coco dataset and trained all weights of the network together for 50 epochs. We used Adam optimisation with an initial learning rate of 10^{-4} and then reduced it to a rate of 10^{-5} after 25 epochs.

Other innovation. NOT APPLICABLE

b. Second Method - Sequential Processing with ED-RD-Net

Two dedicated CNNs are utilized for solving each step: 1) a blob network where the CNN will predict whether a pixel belongs to the nuclei or background class and 2) a distance network where the CNN will regress the nuclear instance centroids, which are encoded in a distance map. Both networks share the same ladder style design, like U-net [3]. In particular, the encoder is a Pre-activated ResNet50 [4] variant with 3 down-sampling levels (by removing the max pooling and setting 7x7 stride to 1), whereas the decoder portion adopts a DenseNet [5] design. Addition is used as a replacement for concatenation to integrate information via skip connections from the encoder to the decoder. We name this architecture ED-RD-Net for simplicity. Both of these networks are trained separately on RGB images as input as opposed to existing techniques where the distance network requires the blob prediction as additional input [6].

Loss Function. The blob network was trained with the standard cross-entropy loss. On the other hand, by denoting *N* as total number of pixel within an image and *k* as the *k-th* pixel, the distance network is trained with the following loss functions:

$$L = \sum_{k}^{N}(d_p(k) - d_t(k))^2 + \sum_{k \in Nuclei}^{N} ||cos^{-1} < \frac{\nabla d_p(k)}{|\nabla d_p(k)|}, \frac{\nabla d_t(k)}{|\nabla d_t(k)|} > ||^2$$

where d_p and d_t are the predicted and ground truth distance map respectively. The first term is simply the mean square error loss between d_p and d_t. Meanwhile, to further emphasise the discontinuities between instances, a squared angular loss is calculated in the second term so that the distance network focuses more on learning the separation between nuclei instances.

Hyper-parameter settings. We initialised the model with pre-trained weights on the ImageNet dataset, trained only the decoders for the first 60 epochs, and then fine-tuned all layers for another 60 epochs. We used Adam optimisation with an initial learning rate of 10^{-4} and then reduced it to a rate of 10^{-5} after 30 epochs. This strategy was repeated for fine-tuning.

Other innovation. NOT APPLICABLE

2. Pre-Processing

a. Color Normalization

We used Vahadane stain normalisation [8] to normalise the stain of one image to that of the image being used as target.

b. Intensity Transformation

NOT APPLICABLE

c. Data Augmentation

We sampled patches of size 256x256 and 270x270 from the original tissue images to respectively train Mask-RCNN and ED-RD-Net. The following augmentations were applied during training: random rotation (-179 to 179 degree), random scale (scale factor ranging from 0.8 to 1.2), random shift horizontal and vertical (-0.1 to 0.1 percentages with respect to the patch height and width), random shear (from -5 to 5 degree), random Gaussian and median blur (random kernel size from 3x3 to 7x7), random Gaussian noise, random brightness adjustment (randomly add values in range -26 to 26), random hue adjustment (randomly add values from -8 to 8 in HSV), random contrast adjustment (with random factor from 0.75 to 1.25), random saturation adjustment (with random factor from -0.2 to 0.2).

d. Other processing steps

NOT APPLICABLE

3. Post-processing

Test time augmentation was utilised during evaluation to increase the robustness of each method. We used flip and rotation test time augmentations and combined the results. In addition, we mapped test images to four chosen target images, that were chosen due to their contrasting stains. Similar to training, we used Vahadane stain normalisation and combined the predictions of each stain. The distance and probability maps of each method were then averaged together. Multi-level thresholding was later applied on the aggregated distance map to obtain the seed points. We used the combined distance map as the energy landscape and the combined probability map as the mask. Then, with the energy landscape, mask and seed points, we performed marker-controlled watershed to obtain the final instances.

4. Computational Complexity

a. Hardware

For Mask-RCNN, we used a computer with Intel® Core i9-7900X CPU @ 3.30GHz with 128 GB RAM and 2 NVIDIA GeForce Titan X GPUs, each with 12 GB memory. For the ED-RD Network, we used a computer with one Intel® Xeon® CPU E5-2630 v3 (2.40GHz) with 64 GB RAM and 2 NVIDIA GeForce GTX 1080Ti GPUs with 11GB memory.

b. Training time

It takes around 8 hours to finish training the Mask-RCNN. Likewise, ED-RD-Net requires about 5 hours 30 minutes. It is important to note that software configurations can heavily influence the training time. The reported measurements rely on the hardware and software setup and therefore, these values are prone to vary with different configurations.

c. Testing time

We utilise a total of 3 GPUs for inference, two GPUs for ED-RD-Net to generate the blob and distance predictions in parallel (each GPU contains 1 network) and another one for Mask-RCNN. Additionally, Mask-RCNN requires around 30 minutes, whereas the ED-RD-Net takes around 10 minutes for processing the entire testing set. Furthermore, we utilise 6 test time augmentations and process each image with 5 different stains, resulting in a total of 30 inference runs on the testing set. Therefore, it would take around 1200 minutes to obtain the final predictions on the testing set using our approach. Similar to the training time, it is important to note that the measurements reported here rely on the hardware and software setup. As a result, the reported values will vary with different configurations.

5. Reference

[1] Kaiming He, Georgia Gkioxari, Piotr Dollar, and Ross Girshick, “Mask R-CNN”
[2] T. Lin and P. Dollar and R. Girshick and K. He and B. Hariharan and S. Belongie, “Feature Pyramid Networks for Object Detection”
[3] Olaf Ronneberger, Philipp Fischer, and Thomas Brox, “U-Net: Convolutional Networks for Biomedical Image Segmentation”
[4] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun, “Identity Mappings in Deep Residual Networks”
[5] Gao Huang, Zhuang Liu, Laurens van der Maaten, and Kilian Q.Weinberger, “Densely Connected Convolutional Networks”
[6] Bai Min and Urtasun Raquel, “Deep Watershed Transform for Instance Segmentation”
[7] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun, “Deep residual learning for image recognition”
[8] Vahadane, Abhishek, et al. "Structure-preserving color normalization and sparse stain separation for histological images." *IEEE transactions on medical imaging* 35.8 (2016): 1962-1971.

H&E Stained Multi Organ Nuclei Segmentation using two Hematoxylin channel U-nets

Rank- 13, Team Name- Unblockabulls
Akshaykumar Gunda[1] and Raviteja Chunduri[2]
[1]Indian Institute of Technology Madras, [2]Indian Institute of Technology Bombay

1. Pre-Processing

a. *Color normalization:* We used Macenko et. al. 's color normalization method where it considers the projection of pixels onto the 2D plane defined by the two principle eigenvectors of the optical density covariance matrix. It then considers the extreme directions (in terms of angular polar coordinate) in this plane.
b. *Intensity Transformation:* Not applicable
c. *Data Augmentation:* Patches of size 128x128 with stride of 40 pixels where sampled from the processed full-size tissue images, which were then Rotated, Flipped and Scaled randomly for a subset of them. The range for rotation was set from -30 deg. To 30 deg., and scale factor from 0.75 to 1.25
d. *Other pre-processing steps*: We included a Hematoxylin channel in addition to the RGB channels of the image, as an input to the process because Hematoxylin stains nuclei acids dominantly and helps the algorithm to segment the nuclei better. This process works on color deconvolution as per [1] implemented by Scikit-image.

2. Model details

a. *CNN architecture:* The deep learning architecture used is U-Net. The network has 5 convolutions with ReLU activation and 5 up convolutions with ReLU activation with skip connections. We performed batch normalization before every pooling & up sampling. A dropout layer in the U-net architecture is included just before the up-convolution step to reduce over fitting. The entire U-net was trained from scratch and no base/pre-trained network was used. We have used 2 U-net's and both are of them have the same architecture. (Please refer to figure 1 in appendix)
b. *Loss Function:* We used 2D binary cross entropy as the cost/loss function, which is a standard implementation of Keras (Tensor-flow).
c. *Hyper-parameter settings:*
    i. Learning rate: 1e-4
    ii. Input/output patch size: 128x128
    iii. Batch size: 16
    iv. Dropout value: 0.2
    v. Optimizer: Adam
d. *Other innovations:* Rather than using a single U-net to segment the nuclei, we used 2 different U-nets and combined their outputs to get the final segmented output. The first U-net is used to segment the nuclei using the pre-processed patches mentioned above, which is a conventional one. A second U-net, which is introduced here, is trained

separately using samples where the ground-truth nuclei size is shrunk by 70% (35% inside border and 35% outside border). The output of the 2nd U net is used as an input to the 1st U-net for the application of watershed segmentation.

3. **Post-processing**
    a. *Test time augmentation (TTA):* We used 4 test time augmented images (horizontal flip, vertical flip, transpose horizontal flip and transpose vertical flip) and combined them by calculating their mean.
    b. *Watershed Segmentation:* We performed the watershed segmentation on the output images of U-net1 after TTA using output images of U-net2, which are considered as local maxima for the implementation of watershed.
    (please refer to figure 2 in appendix)

4. Computational Complexity
    a. Hardware: We used AWS server – p2 X large (which runs on Nvidia Tesla K80 with 12 GB GPU memory) for end to end training and testing of the model.
    b. Training time - For the specified hardware and model, it took 02 hr: 44 min for each U-net to be trained. So, the total training time is 05 hr: 28 min.
    c. Testing time - In 00 hr:01 min for a 1000x1000 image for end-to-end processing (pre-processing and post-processing included). The color normalization process takes 70% of the testing time.

5. **Code Release**- Not applicable

[1] A. C. Ruifrok and D. A. Johnston, "Quantification of histochemical staining by color deconvolution.," Analytical and quantitative cytology and histology / the International Academy of Cytology [and] American Society of Cytology, vol. 23, no. 4, pp. 291-9, Aug. 2001.

Generalized Nuclear Segmentation using a Deep Convolutional Neural Network Method

Corey Hu

July 14, 2018

1 Data Pre-processing

With the aim of better generalizing our model and supplementing for a shortage of training data, we tested a variety of data augmentations.

We color normalized the original images to standardize color variation between different organ types. This appears to yield greatly improved segmentation results among liver cells.

Furthermore, during training, we select a random 64x64 crop from each image, with each crop subject to several augmentations. Each crop is randomly flipped vertically and/or horizontally as well as randomly rotated and scaled. We also add random noise sampled from a normal distribution, and "drop" a small percentage of pixels (randomly selected between 1% and 10%) by converting them to black. The data is passed over a Gaussian blur with a kernel radius of 3 pixels. Furthermore, the contrast of each image is randomly increased or decreased. Lastly, we create an edge mapping from each crop by convolving the image with an edge detection kernel. The edge mapping combined with an embossed image is overlaid on each crop with randomly chosen transparencies.

2 Proposed Model

We propose using a MaskRCNN model architecture for instance segmentation. MaskRCNN is a 2-stage model, with the first stage being the region proposal network (RPN) that is responsible for generating regions of interest (ROI) which contain foreground objects. The second stage is essentially a Faster RCNN model that classifies and predicts bounding boxes given each ROI. Compared to Faster RCNN, MaskRCNN incorporates an additional prediction branch responsible for generating segmentation masks of each ROI that we utilize to create instance masks of each cell.

MaskRCNN is capable of generating precise segmentation masks for individual nuclei instances, allowing us to not only find accurate successfully detect neighboring and overlapping cells. Our model uses a resnet50 backbone and pretrained resnet50 weights.

2.1 Loss

Traditionally, the MaskRCNN loss function is defined in the following manner:

$$L = L_{rpn\ class} + L_{rpn\ bbox} + L_{mrcnn\ class} + L_{mrcnn\ bbox} + L_{mrcnn\ mask}$$

Where $L_{rpn\ class}$ and $L_{rpn\ bbox}$ represent the loss of the RPN's object classification and bounding box offset predictions respectively. $L_{mrcnn\ class}$, $L_{mrcnn\ bbox}$, and $L_{mrcnn\ mask}$ represent the losses of the ROI classification, ROI bounding box, and ROI mask predictions respectively. Given the large number of instances within each image, we place more weight on the region proposal network's losses to encourage more reliable ROI generation from the RPN:

$$L = W_r L_{rpn\ class} + W_r L_{rpn\ bbox} + W_m L_{mrcnn\ class} + W_m L_{mrcnn\ bbox} + W_m L_{mrcnn\ mask}$$

where $W_r = 1.5$ and $W_m = 1$. To optimize our loss, we use SGD with momentum with a learning rate of .001.

2.2 Training

During training, the model is fed augmented 64x64 random crops of the training image. The provided annotations are also converted into instance masks for each cell and are used as ground truth masks when training the model's mask prediction head. Our model is trained in three stages using 25 training images, 5 validation images, and a batch size of 2. In the first stage, we train the RPN separately for 30 epochs. We then train the network heads and region proposal network together for an additional 30 epochs. The third stage involves training the entire model for the last 60 epochs.

2.3 Prediction Rule

To generate predictions, we zero-pad our image to 1024x1024 and color normalize it before feeding it into the model. The model generates a set of 1024x1024 segmentation masks that can be used for evaluation.

3 Data Post-processing

We do not currently use any additional post-processing steps.

4 Results

Our current model's AJI score is .32 on our validation set. We believe this can improved by increasing the number of ROI predictions from our model's RPN.

TernausNet-16 for Nuclei Segmentation

Rank- 15, DeepMD- MoNuSeg
Dariush Lotfi[1*], Reza Safdari[2*]

[1] Science and Research Branch, Islamic Azad University, Tehran, Iran
[2] Qazvin Branch, Islamic Azad University, Qazvin, Iran

1. Pre-processing

a. To overcome large variation in H&E stained images due to H&E reagents, staining process, scanner and the specialist who performs the staining, color normalization proposed in [1] were applied. We chose one best stained H&E image as the target and converted other images into its color space. According to the recommendation in [1], the hyper-parameter λ should be set between 0.01 and 0.1. In our experiment, λ was set to 0.1.

b. Used augmentation methods include horizontally and vertically fliping, geometric affine transformations, piecewise affine to distort local areas with varying strength, blurring each image with varying strength using Gaussian blur, average/uniform blur and median blur, sharpening or embossing each image, adding Gaussian noise to some images and contrast normalization.

2. Model details

a. Our approach is mainly based on a flavor of U-Net [2] architecture named TernausNet [3]. U-Net architecture consists of a contracting path to capture context of the input and a symmetrically expanding path that produces the output mask with skip-connections connecting these two paths. In TernausNet, the contracting path was replaced with a simple CNN of the VGG [4] family that consists of 11 sequential layers known as VGG11. We used a pretrained VGG16 without its fully connected layers as contracting path of the U-Net architecture.

b. The model was trained using the class-weighted cross-entropy loss added by class-weighted dice coefficients loss:

$$loss = \sum_{k=1}^{K} [\sum_{p} (w_k t_{p,k} \log(softmax(y_{p,k}))) + w_k (1 - softdice(k))]$$

where K is the number of channels of the network output (here K=4), $t_{p,k}$ is the k-th element of pixel p's one-hot target vector, $y_{p,k}$ is the k-th channel of the network output at p and w_k is the class weight used to overcome class imbalance and control the loss focus on the classes. We used more weights on touching borders and centroids and less weights on background and inside masks.

c. Our model consists of 13 convolution layers each followed by a ReLU and five max-pooling operations. The expanding path contains 5 alternating convolution and transposed convolution layers followed by a 1x1 convolution which produces the output masks. We trained the model end-to-end using the stochastic gradient descent (SGD) optimizer with a batch size of 4, and a constant learning rate of 5e-3 and L2 weight penalty of 1e-4.

d. The output of the model is a map of 4 binary masks including background, nuclei touching borders, nuclei inside masks and nuclei centroids. Model would learn the distinguishing characteristics of the borders between adjacent nuclei by trying to learn touching borders.

---

* The authors contributed equally.

3. Post-processing

The output of the model is a four channel map containing background, nuclei touching border, nuclei inside masks and centroids. The last two of them are used during post-processing. First, the normalized image is partitioned into patches with overlap. Then, the model output is computed on each patch and all patches predictions are reassembled together averaging on overlaps. Next, the watershed transform [5] is applied to the inside masks, using centroids as markers. Finally, a simple dilation operation is performed to the labeled map of nuclei because the inside masks used in post-processing are smaller than nuclei whole masks.

4. Computational Complexity

a. We used a computer with one Intel i7-4790 quad-core processor with 32 GB RAM and NVIDIA GeForce GTX 1080 Ti graphics card with 11 GB memory.
b. The training set has 24 randomly chosen images. The remaining 6 images have been used to build the validation set. 1000 patches of size 256 * 256 are randomly extracted from each 24 training images to train our model.
c. Training the model approximately took 40 minutes for each epoch.
d. Testing time for a single 1000x1000 image using 256x256 patches with stride 64 was approximately 5 seconds.

References

[1] A. Vahadane, T. Peng, S. Albarqouni, M. Baust, K. Steiger, A. M. Schlitter, A. Sethi, I. Esposito, and N. Navab, “Structure-preserved color normalization for histological images,” in Biomedical Imaging (ISBI), 2015 IEEE 12th International Symposium on. IEEE, 2015, pp. 1012–1015.

[2] 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. Springer, 2015, pp.234–241.

[3] V. Iglovikov, A. Shvets, “TernausNet: U-Net with VGG11 Encoder Pre-Trained on ImageNet for Image Segmentation,” arXiv:1801.05746, 2018.

[4] K. Simonyan and A. Zisserman, Very Deep Convolutional Networks for Large-Scale Image Recognition, arXiv:1409.1556, 2014.

[5] S. Beucher. The watershed transformation applied to image segmentation. Scanning Microscopy International, Suppl:6(1):299–314, 1991.

Multi-Organ Nuclei Segmentation by Prediction of Nuclei Quadrants

Rank - 16, Team Name - Antanas Kascenas
Antanas Kascenas[1] and Alison Q. O'Neil[1]

[1]Canon Medical Research Europe

1. Pre-Processing

a. Color Normalization
NOT APPLICABLE.

b. Intensity Transformation
Intensity normalization was applied by scaling the original pixel value range of [0, 256) to [-1, 1] for each color channel separately.

c. Data Augmentation
We build the training dataset by extracting 256×256 patches uniformly sampled from the original tissue images. The training patches are augmented by random combinations of 0°, 90°, 180°, 270° rotations, horizontal and vertical flips as well as elastic transformations following the method of Simard et al [1].

d. Other pre-processing steps
NOT APPLICABLE.

2. Model details

a. CNN Architecture
Our model is based on the U-Net architecture [2]. We use the available pretrained VGG-16 network [3] as the encoder part of our U-Net. The decoder part is constructed to reverse the downsampling layers present in the encoder by using alternating transposed convolutions and convolutions, with a ReLU activation and batch normalisation on each convolutional layer. Following the U-Net philosophy, the convolution layers in the decoder take the outputs of the corresponding convolutions in the encoder as additional inputs (via skip connections).
The network performs a five-class pixel-wise classification task via a softmax activation. Each pixel belongs to either the background class or one of the four quadrant classes. The quadrant class depends on the pixel's quadrant relative to the nucleus center of mass (northeast, southeast, southwest or northwest) as shown in Figure 1 [4]. In addition, auxiliary predictions of the same type are made using a label model [5]. The auxiliary predictions regularise the network to make more structured predictions and are not used at test time.
The label model itself is the decoder part of a standard autoencoder network which is trained beforehand. The autoencoder is trained to reproduce labels from the training set through a learned bottlenecked representation. The label model is connected to the final convolution

in the second stage (8× downsampling) of the decoder of the U-Net. The label model is frozen during the training of the main network.

b. Loss Function
The whole network is trained by minimizing the weighted sum of two 5-class cross-entropy losses of the main and auxiliary predictions of the network: $Loss = CE_{main}(main\ pred.) + 0.5CE_{aux}(aux\ pred.)$

c. Hyper-Parameter Settings
Networks are trained with the Adam optimizer [6]. In order to preserve information contained in the pretrained part of the CNN, discriminative fine-tuning [7] is used. The main network encoder layers before the first downsampling are trained using a learning rate of 0.000001. The layers after the first downsampling and before the second downsampling use a rate of 0.00001. All other layers use a rate of 0.0001. Training was done using a batch size of 8.

d. Other Innovations
6-fold cross-validation was performed to evaluate the model performance before test data release.

3. Post-processing

a. Predictions are obtained by passing whole 1000×1000 test images through the 6 cross-validation networks.
b. The test predictions are made by averaging the class probabilities predicted by the 6 cross-validation networks.
c. The final classification probability maps are obtained by averaging the results of test predictions for differently augmented test image versions. The test-time augmentation consists of all 8 possible flip and rotation combinations.
d. Each pixel is assigned the class of highest probability indicated in the final probability map (background or one of the four quadrants).
e. The nuclei centers are designated by finding the local minima of the sum of the four distance transforms. The four distance transforms represent distances from every pixel to the closest pixel of each of the four quadrant classes.
f. The image is divided into background and quadrant regions. The quadrant regions are contiguous regions homogeneous in assigned quadrant class.
g. Each nuclei center is assigned the closest quadrant region of each kind.
h. The groups of 4 quadrants for each center constitute the final predicted nuclei.

4. Computational Complexity

a. Hardware
We used a computer with an Intel Xeon processor with 32 GB of RAM and NVIDIA 1080 Ti graphics card with 11 GB of memory.

b. Training time
   Approximately 6 hours for the specified model and hardware.
c. Testing time
   Approximately 20 seconds for a 1000×1000 image.

5. Code Release - Code not publicly available.

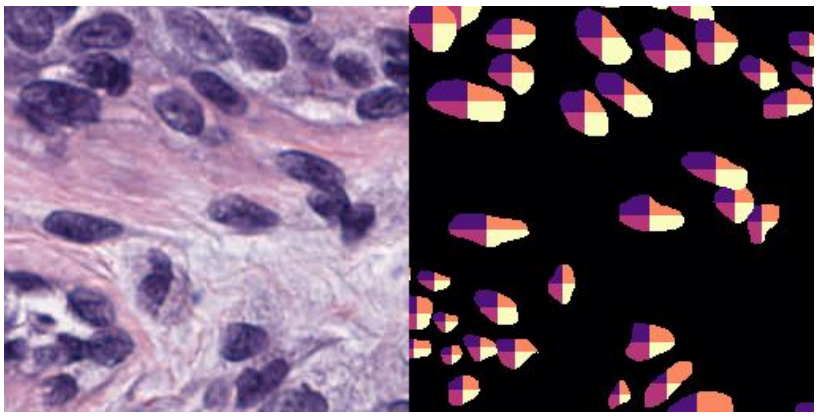


Figure 1. A patch of an image (left) and of the corresponding classification mask (right). Different colors represent different pixel classes.

References

[1] P. Y. Simard, D. Steinkraus, and J. C. Platt, "Best practices for convolutional neural networks applied to visual document analysis," in *Seventh International Conference on Document Analysis and Recognition, 2003. Proceedings.*, Edinburgh, UK, 2003, vol. 1, pp. 958–963.

[2] O. Ronneberger, P. Fischer, and T. Brox, "U-Net: Convolutional Networks for Biomedical Image Segmentation," in *Medical Image Computing and Computer-Assisted Intervention – MICCAI 2015*, 2015, pp. 234–241.

[3] K. Simonyan and A. Zisserman, "Very Deep Convolutional Networks for Large-Scale Image Recognition," presented at the International Conference on Learning Representations (ICLR), 2015.

[4] J. Uhrig, M. Cordts, U. Franke, and T. Brox, "Pixel-Level Encoding and Depth Layering for Instance-Level Semantic Labeling," in *Pattern Recognition*, 2016, pp. 14–25.

[5] M. Mostajabi, M. Maire, and G. Shakhnarovich, "Regularizing Deep Networks by Modeling and Predicting Label Structure," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2018, pp. 5629–5638.

[6] D. P. Kingma and J. Ba, "Adam: A Method for Stochastic Optimization," presented at the International Conference on Learning Representations (ICLR), 2015.

[7] J. Howard and S. Ruder, "Universal Language Model Fine-tuning for Text Classification," in *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, Melbourne, Australia, 2018, pp. 328–339.

Multi-Organ Nuclei Segmentation using a 3-Class U-Net with Watershed-Based Postprocessing

Rank-17, Team Name- Johannes Stegmaier
Dennis Eschweiler[1], Johannes Stegmaier[1]
[1]Institute of Imaging and Computer Vision, RWTH Aachen University, Aachen, Germany

1. Pre-Processing

a. Color normalization: NOT APPLICABLE.
b. Intensity Transformation: For data normalization raw image intensities were limited to the range [0,1] by division of 255.
c. Data Augmentation: The training data was augmented on-the-fly using the Keras image data generator feature. Input images were transformed randomly using rotation range of 40°, horizontal and vertical image flipping and mirroring for border filling. Moreover, height shift, width shift, shear range and zoom range were all set to 0.2 and the intensity range was set to [0.9, 1.1].
d. Other Pre-processing Steps: Images were rescaled to a resolution of 1024x1024 using bilinear interpolation for the raw images and a nearest neighbor interpolation for the label images. This was required to avoid odd dimensions after multiple pooling operations during the contraction and expansion paths of the neural network. The manually labeled nuclei outlines provided as ground truth were converted to 3 labels, namely a background label for all exterior regions, a dilated boundary label and a seed label.

2. Model Details

a. CNN architecture: We used an adapted version of the U-Net architecture [4] that was implemented using Keras (https://keras.io). Slight modifications were applied to the network, including halving the number of all feature channels, adding batch normalization (BN) before max-pooling and up-convolution layers, a dropout layer after the lowest layer and reducing the amount of convolution layers in the expanding path to one.
b. Loss function: We used a weighted binary cross entropy loss [4].
c. Hyper-parameter settings: Dropout probability was set to 0.2. Training of the network was performed using the ADAM optimizer. We trained for 500 epochs with 200 steps per epoch and used a mini batch size of 2 (larger values were not feasible due to memory limitations of the GPU). The network used five repetitions of the following type on the contraction path: Conv (3x3) – Conv (3x3) – BN – MaxPool (2x2), with feature maps of 32 – 512 feature maps, doubling after each max pooling layer. The dropout layer was added after the lowest level of the contraction path. The expansion path was constructed symmetrical to the contraction path, including skip connections but only a single convolutional layer per level.
d. Other innovations: NOT APPLICABLE.

3. Post-Processing

a. The post-processing pipeline was implemented in XPIWIT (https://bitbucket.org/jstegmaier/xpiwit) [1].
b. The trained network yielded probability maps for the background, the nuclei boundaries and seeds located in the center of the nuclei. First, the background and the seed probability maps were binarized using a threshold value of 0.5.
c. The binarized background image were converted to a Euclidean distance map [3] with intensity minima located at nuclei centers. We added the probability map of the nuclei boundaries to the distance map image and multiplied it with the binary mask of the background to prevent objects exceeding the nuclei boundaries.
d. Finally, we used the binarized seed point image and the modified distance map image to obtain the nuclei instances using a seeded watershed algorithm [2] and excluded background segments if they intersected mostly (> 50%) with the identified background probability map. Result images were rescaled to 1000x1000 pixels with nearest neighbor interpolation to preserve the integer labels.

4. Computational Complexity

a. Hardware: We used a computer with one Intel i7-7820X quad-core processor with 64 GB RAM and an NVIDIA GeForce GTX 1080 Ti graphics card with 11 GB memory.
b. Training time: About 42 hours (5 minutes per epoch) for the specified model and hardware.
c. Testing time: Average testing time per image amounts to 0.726 s (0.01 seconds for preprocessing, 0.129 seconds for the CNN prediction and 0.587 seconds for postprocessing).

5. Code Release

NOT APPLICABLE

6. References

[1] Bartschat, A., Hübner, E., Reischl, M., Mikut, R., Stegmaier, J.: XPIWIT - An XML pipeline wrapper for the Insight Toolkit. Bioinformatics 32(2), 315–317 (2015). https://doi.org/10.1093/bioinformatics/btv559

[2] Beare, R., Beare, R., Lehmann, G., Lehmann, G.: The watershed transform in ITK - discussion and new developments. Insight journal pp. 1–24 (2006)

[3] Danielsson, P.E.: Euclidean distance mapping. Computer Graphics and Image Processing 14(3), 227–248 (1980). https://doi.org/10.1016/0146-664X(80)90054-4

[4] Ronneberger, O., Fischer, P., Brox, T.: U-Net: Convolutional Networks for Biomedical Image Segmentation pp. 1–8 (2015). https://doi.org/10.1007/978-3-319-24574- 4 28, http://arxiv.org/abs/1505.04597

A Joint Model of U-Net and Mask-RCNN for Nuclear Segmentation

Rank-18, Team Name-Yanping
Yanping Cui[1], Baocai Yin[2], Xinmei Tian[1], and Kailin Chen[2]
[1] University of Science and Technology of China, China, [2] IFLYTEK AI Research

We proposed a novel method, which combines U-net and Mask-RCNN to obtain a better performance of nuclear segmentation. This method is fully automated and does not require interaction. The overall framework of our method is illustrated in Fig. 1. We will detail the proposed method in the following sections.

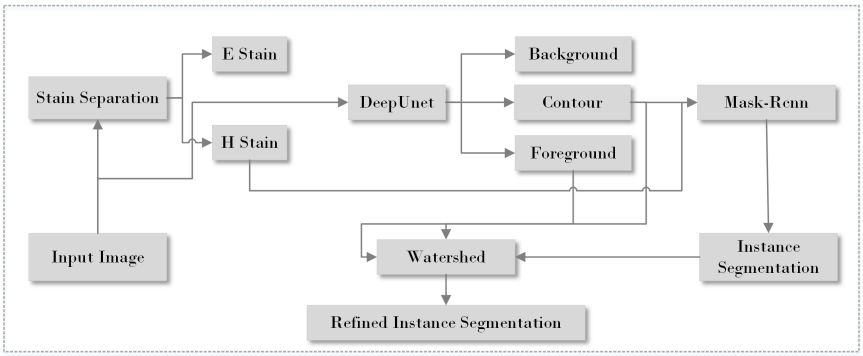


Fig. 1. The overall framework of our method

1 Pre-Processing

a. Color normalization. In our experiments, we use the method proposed in [5], which is currently an state-of-the-art method of color normalization. We also perform staining separation on original RGB image, which can obtain a background and a foreground image according to staining content. The foreground image will be used as part of the input of Mask-RCNN. The value of λ is set as 0.02.
b. Intensity Transformation. NOT APPLICABLE.
c. Data Augmentation. As there are just 30 training images, we need to come up with specific augmentations to prevent our models from overfitting and increase their generalization ability. We use a lot of heavy augmentations, eg. sharpening with lightness between 0.9 and 1.1, embossing, adding gaussian noise with a sigma between 0.01 and 0.1, median blur with kernel size 3, contrast and brightness, randomly scaling with scaling factor between 0.8 and 1.2, rotating by -20 to +20 degrees and flipping. We use the imgaug [1] to implement the above data augmentation.
d. Other pre-processing. NOT APPLICABLE.

2 Model details

a. CNN architecture. Firstly, we use a segmentation network to segment nucleus, boundary of nucleus and background of the pathological image. The backbone of the segmentation network is U-net[4]. We make the following modifications, the first is to increase the depth of U-net to 7, allowing it to have more learning and expression capacity. The second is to, similar

to [3], change the number of output channels of the network to 3, which represent the nucleus, boundary of nucleus and background respectively. The modified U-net is denoted as DeepUnet. Secondly, we obtain the nucleus instance segmentation by Mask-RCNN[2]. The backbone of Mask-RCNN is resnet101. We merge the foreground image from staining separation, boundary of nucleus from DeepUnet into a 3-channel image as the input of Mask-RCNN. In this way, we associate Mask-RCNN with DeepUnet, which will provide guiding information to assist the Mask-RCNN to achieve finer instance segmentation. And the size of input image of Mask-RCNN is set to 256x256.

b. Loss Function. We use weighted softmaxLoss to optimize DeepUnet.
c. Hyper-parameter settings. We train the DeepUnet with SGD optimizer for 300 epochs: 100 epochs with learning rate 1e-4, 100 epochs with learning rate 1e-5, 100 epochs with learning rate 1e-6. Batch size, momentum and weight decay are set to 4, 0.95 and 5e-4 respectively. We train the Mask-RCNN for 150 epochs: 50 epochs with learning rate 1e-3, 50 epochs with learning rate 1e-4, 50 epochs with learning rate 1e-5. We firstly train the head of Mask-RCNN for 50 epochs and the backbone of Mask-RCNN is pretrained on COCO. Then train all layers of Mask-RCNN. Batch size, momentum and weight decay are set to 1, 0.9 and 1e-4 respectively.
d. Other innovations. NOT APPLICABLE.

3 Post-processing

a. Test. The nucleuses are small and dense, thus we set the anchor stride to 1 when generating nuclear candidate regions. We test one image in three steps. In the first step, we need to estimate the size of the nucleus in the test image. Based on the estimated result, we adjust the test image to the appropriate scale so that our Mask-RCNN can work well. Specifically, we first scale test image with three different scaling factors: 0.8, 1 and 1.5. We estimate the size of nucleuses according to the statistical information of all nucleuses we segment from these three kinds of images mentioned above. In the second step, we perform test time data augmentations on the resize image: flipping up and down, flipping left and right, scaling with three factors of 0.8, 1 and 1.2. In other words, there are a total of 12(2x2x3) images to be tested. In the third step, we integrate these predicting results by voting. Before voting, we remove the pixels with low confidence to reduce false positive samples.
b. Merge. We perform erosion operations with kernel size 4x4 on the predicted results of Mask-RCNN, and the result of erosion operations will be used as the makers of watershed algorithm to label the semantic segmentation result of DeepUnet.

4 Computational Complexity

a. Hardware. We used a computer with one Intel E6-2650 processor with 32 GB RAM and NVIDIA GTX 1080 Ti graphics card with 12 GB memory

to train model. We used a computer with one Intel i7-4790k processor with 16 GB RAM and NVIDIA GeForce GTX 750 Ti graphics card with 2 GB memory to perform color separation and color normalization.

b. Training time. The training time of DeepUnet is about 10 hours. The training time of Mask-RCNN is about 16 hours.

c. Testing time. The time of performing color normalization and color separation on a 1000x1000 image is 39.9s and 18.9s respectively. The predicting time of DeepUnet on a 1000x1000 image is 0.13s. The entire test time is about 10 minutes for a 1000x1000 image for end-to-end processing.

References

1. https://github.com/aleju/imgaug
2. He, K., Gkioxari, G., Dollár, P., Girshick, R.: Mask r-cnn. In: Computer Vision (ICCV), 2017 IEEE International Conference on. pp. 2980–2988. IEEE (2017)
3. Kumar, N., Verma, R., Sharma, S., Bhargava, S., Vahadane, A., Sethi, A.: A dataset and a technique for generalized nuclear segmentation for computational pathology. IEEE transactions on medical imaging **36**(7), 1550–1560 (2017)
4. Ronneberger, O., Fischer, P., Brox, T.: U-net: Convolutional networks for biomedical image segmentation. In: International Conference on Medical image computing and computer-assisted intervention. pp. 234–241. Springer (2015)
5. Vahadane, A., Peng, T., Sethi, A., Albarqouni, S., Wang, L., Baust, M., Steiger, K., Schlitter, A.M., Esposito, I., Navab, N.: Structure-preserving color normalization and sparse stain separation for histological images. IEEE transactions on medical imaging **35**(8), 1962–1971 (2016)

Nuclei Segmentation using Sequential Multi-Objective U-Nets

Rank- 19, Philipp Gruening- MoNuSeg
Philipp Grüning, Erhardt Barth
University of Lübeck, Institute for Neuro- and Bioinformatics

Considering the microscopic images of human tissue, an image contains in many cases objects of interest of the same type and scale. This lack of variance seemingly facilitates the overall segmentation task in this context. However, in instance segmentation, segmenting each cell/nucleus separately instead of simple background/foreground separation, occluding cells can pose a severe problem. Often, there is no visual cue of an edge between two occluding cells. In this case deciding which pixel belongs to which cell is rather a matter of shape and geometry than of local texture. While convolutional neural networks (CNNs) are very capable in learning foreground/background segmentation based on image texture, finding cell edges remains a challenge due to occlusions.
In this manuscript, we describe our approach of overcoming this particular problem by incorporating layers that specifically learn shape information. We train three u-nets [1] which are connected in series. The first one learns a simple segmentation defining if a pixel belongs to a cell or background. In the second network, we learn each foreground pixels' normalized direction vector to its nearest boundary pixel. The last network classifies each pixel either as background, cell or cell-to-cell boundary. Finally, we obtain the instance segmentation by using a watershed method as post processing.

1. **Pre-Processing**
    a. NOT APPLICABLE.
    b. Intensity transformation: since the first part of the used network architecture is based on an ImageNet-trained VGG16 model [2], we subtract each image by the ImageNet [3] mean pixel values for each color channel.
    c. For data augmentation, first each image is randomly cropped to size 448x448x3. Then we apply each of the following data augmentations with a probability of 50 percent: (i) compute the inverse image, (ii) randomly swap the color-channels (iii) apply contrast limited histogram equalization (CLAHE) [4] with randomly drawn parameters, (iv) additive Gaussian noise, (v) change of contrast, image is flipped along the (vi) horizontal and/or (vii) vertical axis, (viii) rotate the image by 90°, 180° or 270°.
    d. NOT APPLICABLE.

2. **Model details**
    a. The standard building blocks of our architecture are u-nets, which provide an encoder-decoder structure of consecutive convolution and pooling/upsampling layers. Unlike the original paper, we simplify the models by using only one convolution instead of the double convolutions at each decoder level. For the encoder of the first network, we use the

VGG16 with weights pre-trained on ImageNet. We do not use the fully connected layers fc6 and fc7. Both the second and the third network use a custom encoder, that is not pre-trained, which consist of five blocks. The first 4 blocks contain a convolution-, batchnormalization- and pooling-layer each. The fifth block only consists of convolution with batchnormalization. The weights of those blocks and all decoder layers are randomly initialized with Xavier-initialization [5]. The first network provides a simple foreground/background segmentation with a softmax output minimizing the cross entropy. We concatenate the output with the previous feature layer and apply a 1x1 convolution. Hence, network two gets a HxWx3 Tensor with information on cell texture- and structure. As in [6], the second network learns the direction of the watershed descent as an intermediate task. For each pixel that belongs to a cell, we compute the direction to the nearest boundary pixel, which is either part of the background or of another cell as indicated by the labels. This direction is encoded as a simple 2D unit vector. The output layer of the second network learns to match those vectors by minimizing the squared angles between the actual and the predicted vectors. The third network receives the direction vector map as input. It needs to learn to separate the pixels into the classes background, cell and cell-to-cell edge.

b. Each of the three networks has a specific loss function that is later weighted and summed up to one loss value. The first and third networks minimize the cross-entropy loss, first on two classes (foreground and background) and then on three (background, cell, edge between two cells). The second network learns two dimensional normalized direction vectors and minimizes the squared angle between a predicted vector $\vec{u}_{p_{pred}}$ and a ground-truth-vector $\vec{u}_{p_{gt}}$:

$$loss_{dir} = \sum_p w_p \left\| \cos^{-1} \langle \vec{u}_{p_{gt}}, \vec{u}_{p_{pred}} \rangle \right\|^2.$$

We only compute the loss for those pixels that are labeled as cell pixels, expressed by the binary weight w_p.

c. We train all three networks end-to-end in one run, using a weighted sum of each network's loss function. The weights are 0.5, 0.2, 0.9 for the binary cross-entropy, squared angle distance and output class cross-entropy respectively. We train for roughly 1000 epochs with a batch size of 2, learning rate of 0.01 and a learning rate decay of 0.1 every 330th epoch. We use a normalized stochastic gradient optimizer. Here, in each step the gradient is divided by the gradient's maximum.

d. Our main contributions over the original deep watershed paper [6] are: first, we use end-to-end training in contrast to employing a pre-trained foreground/background segmentation network. Second, we do not use a gated input for the second network, but rather give the network a chance to derive a fitting input from the feature maps and output of the first network itself. Third, instead of learning different height levels, we simplify

the output of the third network to three classes, because this approach showed promising results in baseline experiments with a single u-net.

3. **Post-processing**- To accommodate the full network's output to different input sizes, we adopt the seamless tiling strategy presented in [1]. The input image is of size 448x448 while the network produces an output of size 432x432. The output of the network contains, for each pixel, a probability distribution of either being background, cell, or cell edge. We derive the pixels of an instance segmentation map via watershed segmentation. We create a likelihood map for separated cells: $P_{sc} = P_{cell} - P_{edge}$. Pixels with values that exceed a threshold of 0.8 are marked as starting points. Additionally, pixels with values below 0.2 are used as background starting points. We derive the height map for the segmentation from P_{cell}. Values below 0.3 are set to 0, and values above to 255. After running the watershed algorithm we compute the mean likelihood of a segment belonging to a certain class. If the mean probability of being a cell is below 60 percent, the segment is marked as background

4. **Computational Complexity**
    a. We used a computer with one Intel Xeon E3 Processor with 16 GB RAM and a NVIDIA Geforce GTX 1070 with 8 GB memory.
    b. For the given hardware, training the model took 10 hours.
    c. Inference of one 1000x1000 image takes 5 seconds. Note that the main influence here is the seamless tiling strategy, which leads to a certain number of redundant operations. If necessary, the amount of time can be significantly reduced by using the full initial input size with additional padding (e.g. 1128x1128).

5. **Code Release**- NOT APPLICABLE

References:

[1] O. Ronneberger, P. Fischer, and T. Brox, "U-Net: Convolutional Networks for Biomedical Image Segmentation," in *Medical Image Computing and Computer-Assisted Intervention -- MICCAI 2015*, 2015, pp. 234–241.

[2] K. Simonyan and A. Zisserman, "Very Deep Convolutional Networks for Large-Scale Image Recognition," *CoRR*, vol. abs/1409.1, 2014.

[3] R. Socher *et al.*, "ImageNet: A large-scale hierarchical image database," *2009 IEEE Conf. Comput. Vis. Pattern Recognit.*, pp. 248–255, 2009.

[4] S. M. Pizer *et al.*, "Adaptive histogram equalization and its variations," *Comput. Vision, Graph. Image Process.*, 1987.

[5] X. Glorot and Y. Bengio, "[F] Understanding the difficulty of training deep feedforward neural networks," *Aistats*, 2010.

[6] M. Bai and R. Urtasun, "Deep watershed transform for instance segmentation," in *Proceedings - 30th IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2017*, 2017, vol. 2017–Janua, pp. 2858–2866.

Deep Learning Based Instance Segmentation via Regression Layers

Elad Arbel Itay Remer Amir Ben-Dor
Agilent Labs, Tel-Aviv

1. Preprocessing

1.1 Dataset preparation: Dataset was supplied as part of MoNuSeg H&E stained multi organ nuclei segmentation in digital pathology challenge held on MICCAI 2018 conference. Training data set is composed of 30 1000x1000 image tiles cropped from WSI (captured at 40X magnification) and downloaded from TCGA archive. To ensure dataset diversity, each image corresponds to one patient, where images were taken from 18 hospitals and cover 7 types of organs. In every image tile, nuclei segmentations (ground truth) were provided. For training purposes, since no validation set was provided, We have Selected 11 of the images for validation (those images were not used in the training phase.

1.2 Dataset augmentation: Due to the small number of images for training and their diversity, we use extensive data augmentation that include both standard augmentation procedures such as rotation, mirroring and small resizing, as well as elastic image transformation. In addition, RGB color variation applied through H&E stain intensity absorption domain.

2. Proposed model

Our approach is composed of three main steps (detailed below, see also figure 1). First, encoding the ground truth as a set of two surfaces (see 2.1 below). Second, train a fully convolution network (FCN) based on the UNet architecture (proposed by Ronneberger *et al* in 2015) to predict those surfaces (see 2.2 below). Lastly, in post processing, we use the predicted surfaces to perform constrained watershed segmentation and predict nuclei segmentation (see section 3).

2.1 Ground truth encoding: For each train image we have an associated ground truth segmentation of the pixels into non-overlapping objects (nuclei). We further compute for each nucleus its centroid (see image). We now compute two distance measures, for each pixel. A) distance (in pixels) to the nuclei centroid B) distance to the nearest nuclei edge pixel. Following the approach of Philipp Kainz et. Al. (Miccai 2015), we transform these distances using $d(\mathrm{x}) = \begin{cases} e^{\alpha\left(1-\frac{\mathcal{D}_C(\mathrm{x})}{d_M}\right)} - 1 & \text{if } \mathcal{D}_C(\mathrm{x}) < d_M \\ 0 & \text{otherwise} \end{cases}$, where α and d_M control the shape of the exponential function and DC(x) is the distance to the cell centroid (or to edges) respectively. In addition, we assign a weight for each Pixel. Intuitively, we want to assign higher weights to "critical" pixels, where a mis-prediction will result in an over segmentation. Specifically, we follow similar weighting scheme of UNet, and assign higher weight to pixels that are close to two different nuclei.

2.2 Network architecture: We replace the last UNet layer (a classification layer, for semantic classification), with regression layer that outputs two surface maps. As a loss function we

use weighted sum of squared differences between encoded ground truth and model output.

3. Post processing

To convert the output network surfaces to nuclei segmentation label map we first apply several morphological operations such as open-with-reconstruction and regional H-minima transform to find foreground and background markers from the centroid surface. Finally, predicted label map was generated by markers-controlled watershed using the edge surface regression layer. Parameters for morphological operations were set after applying Bayesian optimization with AJI score as objective function.

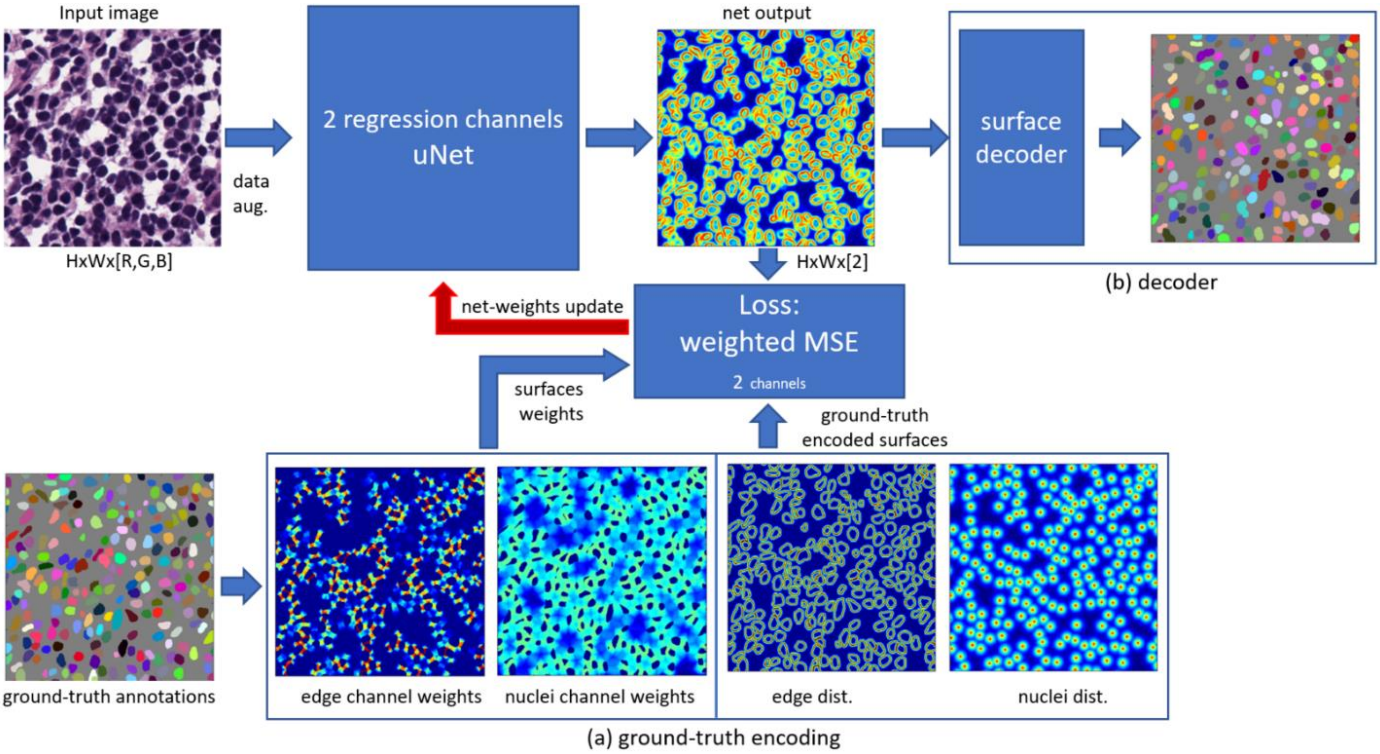


Figure 1*:* **instance segmentation - via 2 channel regression-style fully convolution network**. (a) Bottom - Ground truth encoding as two surfaces (right), and two corresponding weight matrices (left). Middle – training the network using a mean-squared-error as the loss function. (b) decoding net output into (two surfaces), into a final predicted label map

4. Results (validation set)

Image name	aji	Image name	aji
TCGA-21-5786-01Z-00-DX1	0.528	TCGA-E2-A1B5-01Z-00-DX1	0.539
TCGA-50-5931-01Z-00-DX1	0.475	TCGA-G9-6336-01Z-00-DX1	0.620
TCGA-A7-A13F-01Z-00-DX1	0.662	TCGA-G9-6362-01Z-00-DX1	0.654
TCGA-B0-5710-01Z-00-DX1	0.633	TCGA-HE-7130-01Z-00-DX1	0.515
TCGA-DK-A2I6-01A-01-TS1	0.694	TCGA-KB-A93J-01A-01-TS1	0.677
TCGA-NH-A8F7-01A-01-TS1	0.506	**mean score**	**0.591**

Boundary Aware Segmentation of Nuclei in Digital Pathology

Rank: 21, Team Name: Konica Minolta Laboratory Europe
Ekaterina Sirazitdinova[1], Matthias Kohl[1], Stefan Braunewell[1]
[1]Konica Minolta Laboratory Europe

1. Pre-Processing

a. *Color normalization:* NOT APPLICABLE
b. *Intensity Transformation:* NOT APPLICABLE
c. *Data Augmentation:* To cope with the problem of data limitation and to avoid over-fitting, we apply data augmentation. The used augmentation techniques include simple random transformations of input and target images as rotation by maximum two multiples of 90° in both directions, horizontal and vertical flipping, shear and elastic deformation [1] with alpha=150 and sigma=16. Since pathology images do not have fixed orientations, such transformations are valid.
d. *Other pre-processing steps:* We perform supervised learning on three classes: background, boundary and nuclei. The annotated ground truth data contains pixel coordinates of nuclei boundaries. We convert these coordinates into a mask object. The area inside the boundaries yields pixel-wise labels for nuclei, while the area outside the boundaries corresponds to the background. The target images are one-hot encoded so that each target layer consists of binary values corresponding to the represented class.

2. Model details

a. *CNN architecture:* We utilize the U-net architecture [2] which proved to be useful when applied for challenging segmentation tasks with the problem of data limitation. It is a modified fully convolutional network (FCN) architecture consisting of the series of encoding and decoding layers designed to extract different levels of features and to combine these features accordingly. We applied transfer learning, training our model on a pre-trained VGG-16 model (https://www.kaggle.com/keras/vgg16).
b. *Loss Function:* Digital pathology datasets often suffer from class imbalance: nuclei are small sparse objects which cover less of the image area compared to the background. The boundaries, even after dilation, cover even smaller image area. To combat this imbalance, we propose our custom loss minimization strategy. During training, we use a batch size of two, and each time we apply our loss function so, that in the first image the loss is minimized for the complete image and in the second image we compute bounding boxes around individual connected components representing nuclei and minimize the loss on the object level. This strategy helps to focus on nuclei, the objects of interest, and at the same time to consider the effects of background information. Similar to [3], we utilize Dice coefficient as an objective function.

c. *Hyper-parameter settings:* We trained our network on a GPU with the Adam [4] optimizer in Pytorch (https://pytorch.org/) with the initial learning rate of 0.0001 with a scheduled decay of 0.5 every 100th epoch; number of epochs = 300; batch size = 2; input and output image size 992 x 992; depth = 5; number of filters in the first layer = 6.
d. *Other innovations:* loss minimization strategy (see b).

3. **Post-processing**: We apply the learned parameters to the test images and produce three-channel probability maps, where each channel corresponds to one of the target classes. With further simple post-processing (probability treshold of 0.5 and filtering out all connected components containing less than 100 pixels) we are able to produce binary masks of segmented nuclei.

4. **Computational Complexity**
   a. *Hardware:* We used a computer with one AMD Ryzen Threadripper 1900X 8-Core Processor with 62 GB RAM and NVIDIA GeForce GTX 1060 with 6 GB memory.
   b. *Training time:* 3:45:00 hours
   c. *Testing time:* In 00:00:14 for a 1000x1000 image for end-to-end processing

5. **Code Release**: NOT APPLICABLE

References

[1] Simard, P.Y., Steinkraus, D., Platt, J.C.: Best practices for convolutional neural networks applied to visual document analysis. In: Proceedings of the Seventh International Conference on Document Analysis and Recognition -Volume 2. pp. 958–. ICDAR '03, IEEE Computer Society, Washington, DC, USA (2003)
[2] Ronneberger, O., Fischer, P., Brox, T.: U-Net: Convolutional Networks for Biomedical Image Segmentation. In: Medical Image Computing and Computer-Assisted Intervention MICCAI 2015. pp. 234–241. Lecture Notes in Computer Science, Springer, Cham, Munich, Germany (Oct 2015)
[3] Milletari, F., Navab, N., Ahmadi, S.A.: V-net: Fully convolutional neural networks for volumetric medical image segmentation. In: Proceedings of the Fourth International Conference on 3D Vision (3DV). pp. 565–571 (2016)
[4] Kingma, D.P., Ba, J.: Adam: A Method for Stochastic Optimization. In: Proceedings of the 3rd International Conference on Learning Representations (ICLR). pp. 1–15. San Diego, USA (Dec 2014)

Multi-organ Nuclei Segmentation using U-net

Rank- 22, Team Name- OnePiece
Yuexiang Li[1], Xinpeng Xie[1] and Linlin Shen[1]
[1]Computer Vision Institute, Shenzhen University

1. Pre-Processing

a. Color normalization.

We used Vahadane et al.'s color normalization model based on sparse nonnegative matrix factorization (SNMF). The SNMF model is in default parameter setting.

b. Data Augmentation.

For the images of training set, we crop 112 x 112 patches with 75% overlapping by sliding window over the whole image, while there is no overlapping area for the patches cropped from validation images.

2. Model details

a. CNN architecture.

U-Net is widely-used in medical image segmentation. In this study, we employ U-Net as the backbone network of proposed framework. To separate each individual cell, the ground truths for network training have three labels, i.e. background, main body of cells and boundaries. *The architecture of U-Net adopted in this study is the same to that of original U-Net.*

b. *Loss Function.*

Since the instance segmentation is transformed to a problem of multi-class classification for each pixel. The cross-entropy loss is used for network training.

c. Hyper-parameter settings.

Adam is used as the network optimizer. The initial learning rate is set to 0.001. The input and output sizes are 112 x 112. The network is trained with a batch of 32.

3. Post-processing

We used masker-watershed transform to separate the adhered cells.

4. Computational Complexity

a. Hardware.

We used one P100 GPU with 12 GB memory to train our deep learning segmentation network.

b. Training time.

The network is observed to converge after 5 epochs of training, which consumes about 3 hours.

c. *Testing time*
For a 1000x1000 image, the model needs about 1 min to process with the GPU resource.

5. **Code Release**
NOT APPLICABLE

Automatic Multiple Organ Nuclear Segmentation by Lightweight CNN and Watershed-based Method

Rank-23, Team Name- Junma
Jun Ma[1]
[1]Department of Mathematics, Nanjing University of Science and Technology

1. Pre-Processing

a. Color normalization, We used Vahadane et. al. 's color normalization model based on sparse nonnegative matrix factorization (SNMF). The value of lambda is set to 0.02.
b. Intensity Transformation, We extract the blue channel from each RGB color image and scale the value to [0,1].
c. Data Augmentation Patches of size 256x256 were sampled from the original tissue images, which were then flipped for data augmentation.
d. Other pre-processing steps- NOT APPLICABLE.

2. Model details

a. CNN architecture- The employed deep learning model has 10 layers, which integrate dilated convolution with different dilated rates and residual connections. No pre-trained CNN is adopted. Following is the configurations.

Layers	Configurations (kernel size, channel number)
Conv	(3,3), 8
Dilated Conv	(3,3), 16, dilated factor = 1
Dilated Conv	(3,3), 32, dilated factor = 2
Dilated Conv	(3,3), 64, dilated factor = 4
Dilated Conv	(3,3), 64, dilated factor = 4
Dilated Conv	(3,3), 64, dilated factor = 2
Dilated Conv	(3,3), 64, dilated factor = 1
Conv	(3,3), 64
Conv	(1,1), 64
Conv	(1,1), 2

A residual connection is added to each "Dilated Conv".

b. Loss Function- We used Dice Coefficient as the loss function.
c. Hyper-parameter settings- We use Adam optimizer with 0.001 learning rate. The batch size is set to 8 and dropout is not used.
d. Other innovations- The main feature of the employed CNN is that it is a lightweight architecture with only 0.176M parameters.

3. **Post-processing**- To separate the overlapping cells, we use watershed-based segmentation method to refine the CNN's outputs 'CNN_BW'. Following is the step-by-step details.

Step 1. Remove isolated points and fill holes less than 10 pixels in CNN_BW.
Step 2. Compute the distance transform of the complement of the binary image in Step 1 and complement the distance transform.
Step 3. Filter out tiny local minima using MATLAB function “imextendedmin” and impose minima using “imimposemin”.
Step 4. Finally, repeat the watershed transform step above and force the transform result’s pixels that equal to zero to be zero in CNN_BW as well.

4. **Computational Complexity**
    a. Hardware We used a computer with one Intel i7-8700 hexa-core processor with 16 GB RAM and NVIDIA 1080Ti graphics card with 11 GB memory.”
    b. Training time- The total training time is 2:17 (Hours:Minutes) for the specified model and hardware.
    c. Testing time- Average processing time is 0:1.2 (Hours:Minutes) for a 1000x1000 image for end-to-end processing (pre-processing and post-processing included).

5. **Code Release**- NOT APPLICABLE.

Nuclei Segmentation using Deep Learning and Convexity Defects

Rank- 24, Team Name- Biosciences R&D, TCS Research
Krishanu Das Baksi[1]
1. Biosciences R&D, TCS Research, TATA Consultancy Services Ltd., Pune, India

1. **Pre-Processing**
    a. Color normalization: NOT APPLICABLE
    b. Intensity Transformation: NOT APPLICABLE
    c. Data Augmentation was done by flipping the images vertically and horizontally, transposing them and rotating all the previous (original, flipped and transposed images) by 90 degrees, thereby yielding 8 augmented images per original image. In total, after augmentation, we ended up with 240 images.
    d. Other pre-processing steps: the images were padded with 12 pixels on every side, so as to come up with 1024 by 1024 pixel images. Additionally, from all the masks, the border (of the nuclei and the background) information was extracted, and thereby all the masks were combined into an array of the following dimensions: 1024 by 1024 by 3, so that each position is one hot encoded, i.e. having one at pixel positions corresponding to the classes (nucleus, border, background) it belongs to and zero otherwise.

2. **Model details**
    a. CNN architecture: Unet architecture without any pre-trained encoder/decoder. Input: Raw Image. Output: Three class prediction map (Nucleus, Background and Border)
    b. Loss Function: (1 - Jaccard Distance Loss (Intersection over Union))
    c. Hyper-parameter settings: Adam Optimizer was used for the optimization step. The learning rate used was 0.01, but it would reduce 0.5 times every time the model training stagnated for more than 50 epochs. The model was trained for 150 epochs.
    d. Other innovations: NOT APPLICABLE

3. **Post-processing**- In order to have more confidence on the predictions, test time augmentation was done in the following way. For every image, the horizontal and vertical flipped and transposed images were used to make the pixel prediction map and then these (after flipping/transposing back) were combined to predict the final segmentation probability maps. From the segmentation probability map, all pixels with nucleus probability value above a certain threshold and connected to others were taken as one mask. However, such masks were inaccurate as there were some masks which contained more than one nucleus and there was no simple way to separate them. The idea that nuclei are mostly convex shaped was used to segment them further. Briefly, convexity defects were used to find out if a predicted mask contains two or more nuclei and the 2 closest convexity defect points were connected by a straight line, thereby, giving two separate masks. This process was done recursively on each mask and newly separated masks, so that every mask with 2 or more nuclei can be separated into its corresponding components. These masks were used as seed points and watershed algorithm for image segmentation was applied to give the final nucleus masks.

4. **Computational Complexity**
    a. Hardware: The model was trained on a NVIDIA Tesla K80 GPU for 150 epochs, and it took approximately 100 seconds per epoch, and approximately 5 hours was required to train the model.
    b. Training time: 5 hours.
    c. Testing time: 4 seconds for segmentation map formation and <10 seconds per image for post processing (convexity detection, watershed segmentation etc.)

5. **Code Release**- NOT APPLICABLE

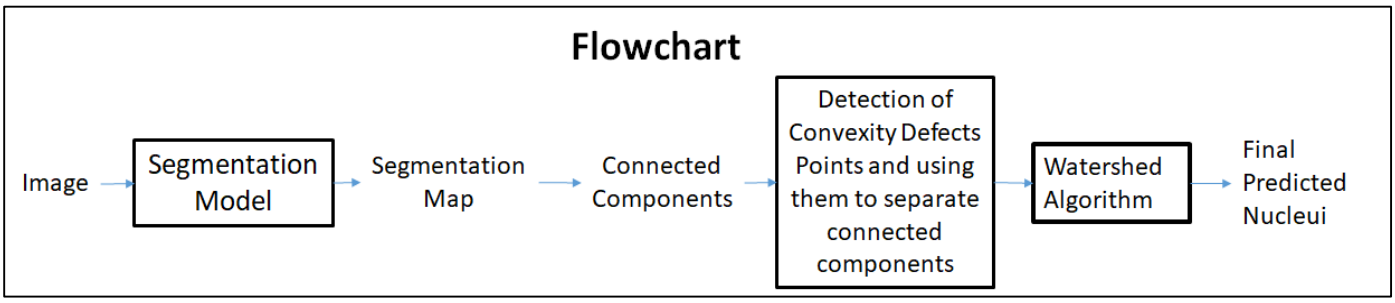


Figure 1: Flowchart of the strategy used.

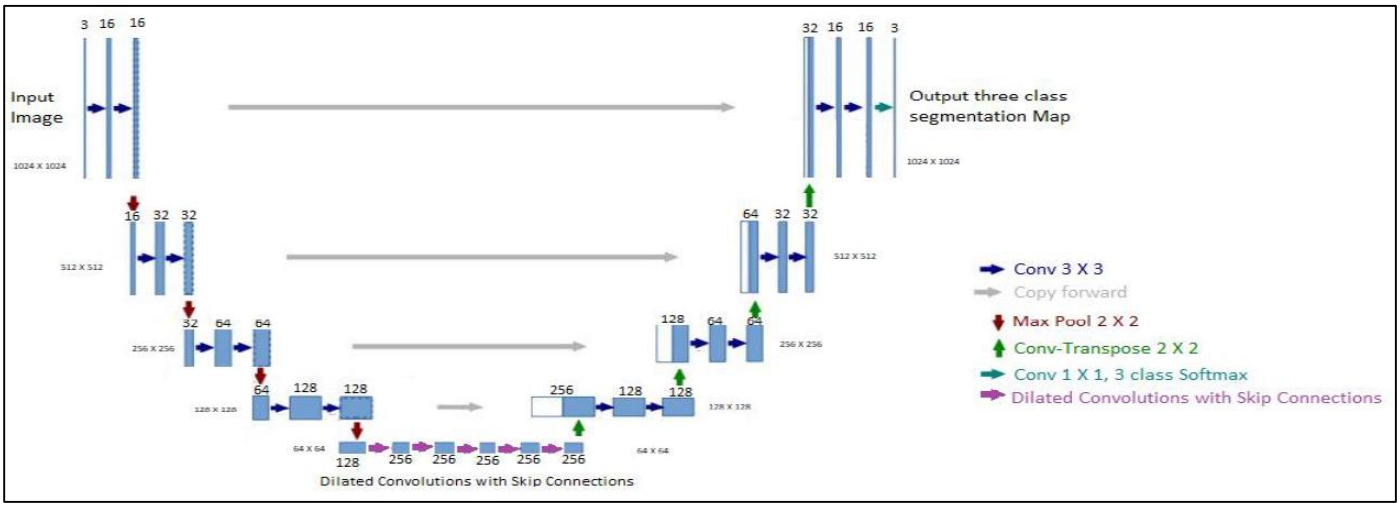


Figure 2: The model architecture showing the number of filters in the convolutions, as well as the dimensions of the intermediate layers

Multi-Organ Nuclei Segmentation using Mask R-CNN

Mohammad Azam Khan
Dept. of Computer Science and Engineering
Korea University
Seoul, South Korea
a_khanss@korea.ac.kr

Jaegul Choo
Dept. of Computer Science and Engineering
Korea University
Seoul, South Korea
jchoo@korea.ac.kr

***Abstract*—Nuclei detection and segmentation is an important task in medical image processing and analysis. However, today's manual procedure for these tasks is prone to misinterpretation and would normally require extensive time by pathologists. This is really a daunting job since the experts need to review the sheer volume of data for the interpretation. Hence, an automated process has a great potential to reduce their workload and help the pathologist for such tasks. In this work-in-progress paper, we demonstrate the efficacy of the Mask R-CNN model, among a variety of deep neural networks, at detecting nuclei in microscopy images. Our initial experiments show that such networks can be used to perform automatic segmentation in diverse set of Hematoxylin-Eosi (H&E) stained histology images obtained from different hospitals spanning multiple patients and organs.**

***Index Terms*—nuclei detection and segmentation, Mask R-CNN, convolutional neural networks (CNNs), microscopic tissue image, medical research**

I. Introduction

In the last few years, significant advancements have been emerged in the computer vision tasks using convolutional neural networks (CNNs) [1]. These tasks include object detection, object localization, semantic segmentation, and object instance segmentation etc. [1]–[4]. The phenomenon has led to increased interest in the applicability of CNN-based methods for problems in medical image analysis as well. Recent work has shown promising results on tasks as diverse as segmentation of liver and tumor 3D volumes, 3D knee cartilage segmentation, automatic nucleus segmentation, Detecting Cancer Metastases [5]–[8].

Mask R-CNN [9] is a recently proposed state-of-the-art deep neural network model that is very effective for object detection, localization, and instance segmentation of natural images. Although the properties of natural images will in general differ significantly from medical images, given the effectiveness of Mask R-CNN at general-purpose object instance segmentation, we can leverage the model for the automatic nuclei segmentation in microscopy images.

II. The Challenge

Multi-Organ Nuclei Segmentation (MoNuSeg) is an official satellite event of MICCAI 2018 [10]. This is a challenge event, which showcases the best nuclei segmentation techniques that will work on a diverse set of H&E stained[1] histology images. The organizer hopes that the best technique will enable training and testing of readily usable (or generalized) nuclear segmentation softwares.

A. Goal of the challenge

Given the diversity of nuclei appearances across multiple organs and patients, and the richness of staining protocols adopted at multiple hospitals, the training datatset will enable the development of robust and generalized nuclei segmentation techniques that will work right out of the box. The organizer strictly prohibited to use any external data, which may violate of the spirit and rules of the challenge.

B. Dataset

Training data containing 30 images and around 22,000 nuclear boundary annotations has been released publicly [11]. A sample image containing multiple nuclei and its associated masks are shown in Fig. 1. A test set of 14 new images has been prepared for the challenge and released only to the participating teams who have submitted their methodology manuscripts before the deadline, upon assigning the agreements. The format of the new cases is exactly the same as the training data (without annotations).

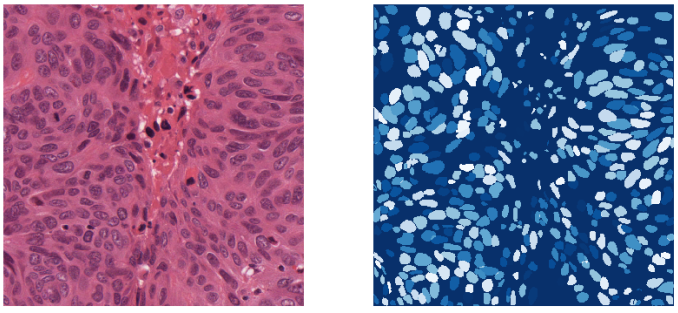


Fig. 1: A sample training image and nuclei masks.

According to the organizer, the dataset for the challenge was obtained by carefully annotating tissue images of several patients with tumors of different organs and who were diagnosed

[1]H&E staining is a routine protocol to enhance the contrast of a tissue section and is commonly used for tumor assessment (grading, staging, etc.)

at multiple hospitals. This dataset was created by downloading H&E stained tissue images captured at 40x magnification from TCGA archive [12].

III. Pre-Processing

The dataset consists of only 30 number of training examples. Of these images, we use 29 images for training and use only one image for validation purposes due to a limited number of images. In addition, the organizer supplied a ground truth and predicted pixel map for segmented nuclei for one image only as an example for calculating average jaccard index (AJI) score.

A. Color normalization

To combat the variety of hematoxylin and eosin (H&E) stained color because of chemical preparation difference per image, color normalization is performed by applying random hue and saturation ([-4, 4]), brightness ([0.55, 1.25]), and contrast ([0.25, 1.75]) and Gaussian Blur with sigma ([0, 5]). We hypothesize that CNN model becomes robust against stain color variety by applying stain color normalization at the training step.

B. Intensity Transformation - NOT APPLICABLE

C. Data Augmentation

We perform some data augmentation techniques at the training step to avoid overfitting. The dataset was augmented using random crops, random rotations(0, 360), scaling (0.9, 1.1) and translation (-8, 8), random horizontal flip (with 50% probability) and vertical flips (with 20% probability).

D. Other pre-processing steps - NOT APPLICABLE

IV. Model details

As mentioned earlier, we hypothesize that Mask R-CNN is a reasonable candidate to use in automated segmentation of medical images. Here, we investigate the efficacy of a Mask-RCNN model for detecting nuclei in microscopy images.

A. CNN architecture

We use a Mask R-CNN model with a feature pyramid network backbone for our experiments. The implementation used is based on an existing implementation by Matterport Inc. released under an MIT License published in github.com [13]. The Matterport Inc. used the open-source framework Keras [14] with Tensorflow [15] back-end in their development. The implementation is well-documented and easy to extend. For these experiments, we tried both a ResNet-50 feature pyramid network model and a ResNet-101 feature pyramid network model as a backbone. In our primary investigation, we find that the model with ResNet-50-FPN backbone has a somewhat lower computational load than that with a ResNet-101 backbone. At the same time, the ResNet-101-FPN gives little improved results with no other changes to the model or training procedure.

B. Loss Function

The multi-task loss function of Mask R-CNN combines the loss of classification, localization and segmentation mask: $L = Lcls + Lbox + Lmask$, where Lcls, Lbox and Lmask are the classification, localization and segmentation loss, respectively.

C. Hyper-parameter settings

As mentioned earlier, we used ResNet50 as a backbone network. We use 64 anchors per image to use for RPN training. In addition, the maximum number of ground truth instances to use in one image is set to 200 while the value was 300 for final detection per image. The head of the network was trained for 10 epochs while the entire network was trained for further 40 epochs.

D. Other innovations - NOT APPLICABLE

V. Post-processing

To select most confidence prediction only, we perform non-max suppression (NMS) technique as a part of post-processing. In this regard, NMS threshold was set to 0.9 to filter RPN proposals. Using this, we could manage to avoid duplicates for the same object.

VI. Computational Complexity

A. Hardware

We used a computer with Intel(R) Xeon(R) CPU E5-2687W v3 @ 3.10GHz with 384 GB RAM and 2 NVIDIA GeForce GTX 1080 graphics cards with 8 GB memory each.

B. Training time

C. Testing time

VII. Code Release - NOT APPLICABLE

VIII. Results

We used ResNet-50 to select feature pyramid network backbone for our proposed Mask R-CNN. With our limited experiments, we achieved around 0.4645 AJI (Average Jaccard Index) score for the validation image. The validation is sampled from organ type 'Liver'. The state-of-the-art AJI score [11] for this type of organ is around 0.5, our model performs almost similar to this performance.

Conclusion

Nuclei detection and segmentation in digital microscopic tissue images can enable extraction of high-quality features for nuclear morphometrics and other analysis in computational pathology. Techniques that accurately segment nuclei in diverse images spanning a range of patients, organs, and disease states, can significantly contribute to the development of clinical and medical research. States-of-the-art CNN models, such as Mask R-CNN, can be used for accelerating this kind of research effectively.

REFERENCES

[1] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," arXiv preprint arXiv: 1512.03385 (2015).

[2] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "Imagenet classification with deep convolutional neural networks," in *Proceedings of the 25th International Conference on Neural Information Processing Systems - Volume 1*, ser. NIPS'12. USA: Curran Associates Inc., 2012, pp. 1097–1105. [Online]. Available: http://dl.acm.org/citation.cfm?id=2999134.2999257

[3] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," arXiv preprint arXiv: 1409.1556 (2014).

[4] S. Zagoruyko, A. Lerer, T.-Y. Lin, P. O. Pinheiro, S. Gross, S. Chintala, and P. Dollr, "A multipath network for object detection," arXiv preprint arXiv: 1604.02135 (2016).

[5] P. F. Christ, F. Ettlinger, F. Grün, M. E. A. Elshaer, J. Lipková, S. Schlecht, F. Ahmaddy, S. Tatavarty, M. Bickel, P. Bilic, M. Rempfler, F. Hofmann, M. D'Anastasi, S.-A. Ahmadi, G. Kaissis, J. W. Holch, W. H. Sommer, R. Braren, V. Heinemann, and B. H. Menze, "Automatic liver and tumor segmentation of ct and mri volumes using cascaded fully convolutional neural networks," *CoRR*, vol. abs/1702.05970, 2017.

[6] A. Prasoon, K. Petersen, C. Igel, F. Lauze, E. Dam, and M. Nielsen, "Deep feature learning for knee cartilage segmentation using a triplanar convolutional neural network," in *Medical Image Computing and Computer-Assisted Intervention – MICCAI 2013*, K. Mori, I. Sakuma, Y. Sato, C. Barillot, and N. Navab, Eds. Berlin, Heidelberg: Springer Berlin Heidelberg, 2013, pp. 246–253.

[7] J. W. Johnson, "Adapting mask-rcnn for automatic nucleus segmentation," arXiv preprint arXiv: 1805.00500 (2018).

[8] Y. Liu, K. K. Gadepalli, M. Norouzi, G. Dahl, T. Kohlberger, S. Venugopalan, A. S. Boyko, A. Timofeev, P. Q. Nelson, G. Corrado, J. Hipp, L. Peng, and M. Stumpe, "Detecting cancer metastases on gigapixel pathology images," 2017, initial publication on arxiv, then submit to MICCAI. [Online]. Available: https://arxiv.org/abs/1703.02442

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

[10] M. 2018, accessed: 2018-07-14. [Online]. Available: https://www.miccai2018.org

[11] N. Kumar, R. Verma, S. Sharma, S. Bhargava, A. Vahadane, and A. Sethi, "A dataset and a technique for generalized nuclear segmentation for computational pathology," *IEEE Transactions on Medical Imaging*, vol. 36, no. 7, pp. 1550–1560, July 2017.

[12] T. Archive, accessed: 2018-07-14. [Online]. Available: https://cancergenome.nih.gov/

[13] Matterport, accessed: 2018-07-14. [Online]. Available: https://github.com/matterport

[14] F. Chollet *et al.*, "Keras," https://github.com/fchollet/keras, 2015.

[15] 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. [Online]. Available: https://www.tensorflow.org/

Multi-organ Nuclei Segmentation Through Fully Convolutional Neural Networks and Marker-controlled Watershed

Rank-26, CVBLab - MoNuSeg
Adrián Colomer and Valery Naranjo
Instituto de Investigación e Innovación en Bioingeniería, I3B,
Universitat Politècnica de València

December 29, 2018

1 Pre-Processing

1. **Color normalization**. Multi-organ Nuclei Segmentation (MoNuSeg) challenge provides a training dataset composed of thirty digital microscopic tissue images from different organs and patients with their corresponding nuclei annotations. This fact propitiates a large inter-image diversity in terms of colours of the tissue components. For this reason, a colour normalisation procedure based on sparse nonnegative matrix factorization (SNMF) [1] is carried out as first step of the proposed algorithm. Note that we used a value of $\lambda = 0.02$ and the image "TCGA-E2-A14V-01Z-00-DX1.tif" was chosen as target image due to its colour stain distribution.

2. **Intensity Transformation**. NOT APPLICABLE. We train the proposed U-Net architecture using the RGB histological images.

3. **Data augmentation**. After the pre-processing step, data conditioning is required to feed the neural network. Due to the low number of available images, a patch-based approach is required, so a random selection of patches is performed. In this process, we make sure that patches containing the three pixel-classes to be segmented are selected in a balanced way. In particular, 4500 pixel locations of each class are randomly chosen from each training image. After that, a 64×64 patch centred in each pixel location is cropped from the color-transformed RGB image and the corresponding annotation mask. This procedure results in 405000 patches containing information about the background (i.e. stroma, cytoplasm and lumen pixels) as well as nuclei boundaries and nuclei content. This information, without performing data augmentation, feeds the encoder-decoder architecture during the learning process.

4. **Other pre-processing steps**. NOT APPLICABLE.

2 Model details

1. **CNN architecture**. The U-Net architecture proposed in [2] for cell segmentation was modified according to Figure 1 to segment nuclei in histological images. The proposed architecture was trained through the 64×64 RGB patches from scratch.

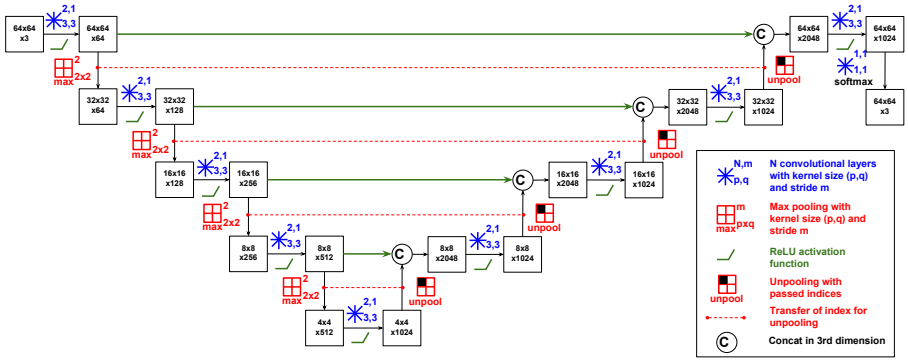


Figure 1: Encoder-decoder network for the semantic segmentation of nuclei in histological images.

2. **Loss function**. The generalised Dice loss function [3] was chosen to solve the multiclass segmentation problem.

$$GDL = 1 - 2\frac{\sum_{l=1}^{2} w_l \sum_n r_{ln} p_{ln}}{\sum_{l=1}^{2} w_l \sum_n r_{ln} + p_{ln}} \tag{1}$$

where r_{ln} is the reference foreground segmentation for class l (i.e. the gold standard), p_{ln} is the predicted probabilistic map for the foreground label and w_l is used to provide invariance to different label set properties correcting the contribution of each label by the inverse of its volume.

3. **Hyper-parameter settings**. The Nesterov Adam optimiser and a learning rate of 1×10^{-5} were used to train the model during 80 epochs. Mini-batches of 64 patches were established to train the model making use of a Titan Xp graphics processing unit (GPU).

4. **Other innovations**. The indexes of each pooling operation in the encoder path are achieved and transferred to the corresponding unpooling layer in the decoder block to preserve spatial consistency. The unpooling layer upsamples the feature maps from the previous decoder block to a double resolution by using the achieved pooling indexes corresponding to the

matched encoder block. After this step, a concatenation of the upsampled feature maps with the corresponding output feature maps of the matched encoder block (skip connections) is performed to enrich the information and avoiding vanishing gradient problems [4].

3 Post-processing

To obtain the segmentation mask of a testing image, we use the predicted probabilities for each label as inputs of a marker-controlled watershed. The thresholded nuclei and background probability maps are used as internal and external markers, respectively. The input image for the watershed algorithm is a combination between the probability map of the nuclei boundaries and the gradient of the hematoxylin component of the colour-transformed image. An overview of the proposed method is reported in Figure 2.

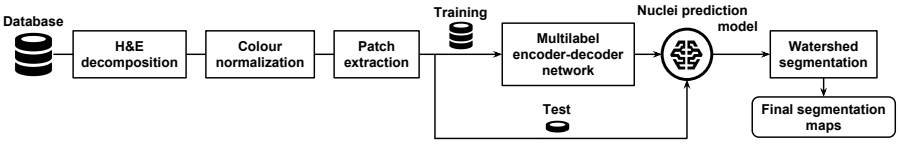
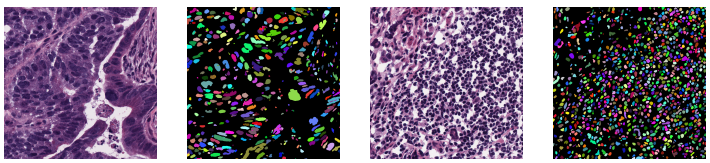


Figure 2: General overview of the proposed method.

4 Computational Complexity

The fact that the annotations of the testing dataset will be released after the celebration of the MoNuSeg challenge makes impossible to provide results of performance over these images. However, some preliminary results (see Figure 3) were obtained by training a model with 28 images and using the remaining two images (randomly selected) to evaluate the performance. Remark that final results (over the 14 testing images) were computed using a model trained with the 30 training images.



(a) (b) (c) (d)

Figure 3: Evaluation of the nuclei segmentation. (a) TCGA-NH original image, (b) TCGA-NH segmentation map ($AJI = 0.5189$ and $Dice = 0.7743$), (c) TCGA-RD original image and (d) TCGA-RD segmentation map ($AJI = 0.7130$ and $Dice = 0.8850$).

1. **Hardware**. This work was performed on an Intel i7@3.10 GHz of 16 GB of RAM with an NVIDIA Titan Xp graphics card with 12 GB memory.

2. **Training time**. Regarding the computational cost of training this model, 11.14 minutes are required for learning during one epoch. As we mentioned above, the model was trained during 80 epochs, i.e. around 14 hours and 51 minutes.

3. **Testing time**. The computational cost of predicting a new test image is 0.9951 sec (average time measured across the fourteen testing images) while the post-processing step to generate the final segmentation map from the probability maps takes 1.5874 sec. Considering both stages, 2.5825 sec is the required time to obtain the nuclei segmentation from a colour normalised image.

5 Code Release

`https://github.com/cvblab/MoNuSeg_Challenge`

References

[1] A. Vahadane, T. Peng, A. Sethi, S. Albarqouni, L. Wang, M. Baust, *et al.*, "Structure-preserving color normalization and sparse stain separation for histological images," *IEEE Transactions on Medical Imaging*, vol. 35, no. 8, pp. 1962–1971, Aug. 2016, ISSN: 0278-0062.

[2] O. Ronneberger, P. Fischer, and T. Brox, "U-net: Convolutional networks for biomedical image segmentation," in *Medical Image Computing and Computer-Assisted Intervention – MICCAI 2015*, N. Navab, J. Hornegger, W. M. Wells, and A. F. Frangi, Eds., Cham: Springer International Publishing, 2015, pp. 234–241, ISBN: 978-3-319-24574-4.

[3] C. H. Sudre, W. Li, T. Vercauteren, S. Ourselin, and M. Jorge Cardoso, "Generalised dice overlap as a deep learning loss function for highly unbalanced segmentations," in *Deep Learning in Medical Image Analysis and Multimodal Learning for Clinical Decision Support*, Cham: Springer International Publishing, 2017, pp. 240–248, ISBN: 978-3-319-67558-9.

[4] E. Shelhamer, J. Long, and T. Darrell, "Fully convolutional networks for semantic segmentation," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 39, no. 4, pp. 640–651, Apr. 2017.

Multi-organ nuclei segmentation in digital pathology data using deep neural network

Rank- 27, Team Name- Linmin Pei

Linmin Pei[1] and Khan M. Iftekharuddin[1]

[1]Vision Lab, Electrical & Computer Engineering, Old Dominion University

1. **Pre-Processing**
    a. Color normalization. We used Vahadane et. al. 's color normalization model based on sparse nonnegative matrix factorization (SNMF) [1]. Color variation exists in hematoxylin and eosin (H&E), which plays an important role for its image analysis. To narrow all images to have same/similar intensity scope, we take one image from training dataset as reference, then apply the SNMF to other images. A given RGB image is first converted to optical density based on Beer-Lambert law. Then, we add a sparseness constraint to optimize the NMF cost function. In the color normalization, we set lambda as 0.02, and stain value as 2.
    b. Intensity Transformation. NOT APPLICABLE.
    c. Data Augmentation. Patches of size 64x64 are sampled from the color normalized tissue images, which are then flipped and rotated for data augmentation. Rotation angles are [0, 90, 180, 270].
    d. Other pre-processing steps. NOT APPLICABLE.
2. **Model details**
    a. CNN architecture- We use U-Net deep learning model in the method [2]. The U-Net architecture consists of an encoding and decoding stages. The encoding has 5 convolutional blocks consisting of two convolutional layers with a filter size of 3*3 and stride of 1 followed by maxpooling with stride 2*2. The decoding stage consists of deconvolution layer with a filter size of 3*3 and stride of 2*2 which doubles the size of the feature maps.
    b. Loss Function. Instead of using cross-entropy loss or dice coefficient as lose function, we use mean squared error (MSE) as the loss function.

    The loss can be described as :

$$\ell(x, y) = \frac{1}{n}\sum_{i=1}^{n} l_i, \; l_i = (x_i - y_i)^2,$$

    where N is the batch size.
    c. Hyper-parameter settings- In training phase, we use ADAM optimizer with an initial learning rate (LR) as 0.01, and then reduce the LR by factor 5 over each 20 epochs [3]. The total epoch is 200, and batch size is 8.
    d. Other innovations- NOT APPLICABLE
3. **Post-processing**- After nuclei segmentation, we have three post-processing steps, small object removal, watershed transform and image erosion for solving nuclei overlapping.
    a. Small object removal. There are some small objects because of segmentation false positive. We remove the small regions that

total pixels are less than 60, empirically. With small object removal, it could improve the segmentation performance.

b. watershed transform. The watershed transform treats an image as a topographic map, with the intensity of each pixel representing the height [4]. Due to nuclei overlapping and segmentation, watershed transform is used to separate large objects.

c. Image erosion. Even though with watershed transform for overlapping issue, there is still some nuclei overlapping. In the case, we apply image erosion for further object separation. If the centroids within one region of watershed image have large enough distance, we consider the region has multiple nuclei. The separation boundary is the orthogonal line at the middle of two centroids.

4. Computational Complexity

a. Hardware. We used a computer with one Intel Xeon(R) CPU E5-2687W v3 processor with 34 GB RAM and NVIDIA Quadro M2000 graphics card with 4 GB memory.

b. Training time- It takes around 13 hours and 20 minutes to complete the training process.

Testing time- For each image, it takes 53 seconds to complete segmentation and also post-processing steps.

c. **Code Release**- NOT APPLICABLE

Important Instructions

1. **Flowchart-** The flowchart is showing in Figure 1.

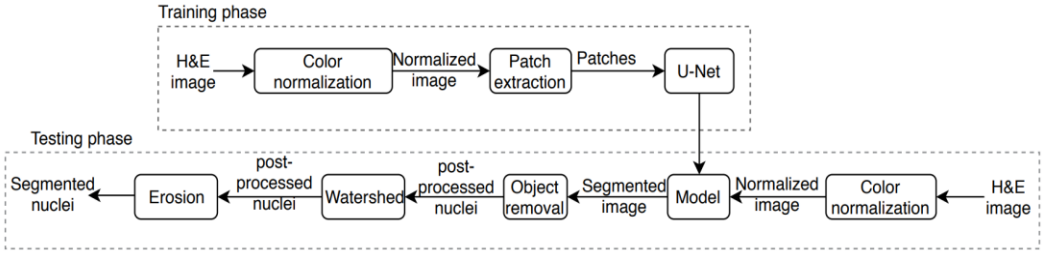


Figure 1. Pipeline of the proposed method for multi-organ nuclei segmentation.

2. **Other figures-** NOT APPLICABLE.

Reference

[1] A. Vahadane *et al.*, "Structure-preserving color normalization and sparse stain separation for histological images," *IEEE transactions on medical imaging,* vol. 35, no. 8, pp. 1962-1971, 2016.

[2] 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*, 2015, pp. 234-241: Springer.

[3] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," *arXiv preprint arXiv:1412.6980,* 2014.

[4] S. Beucher, "The watershed transformation applied to image segmentation," *SCANNING MICROSCOPY-SUPPLEMENT-,* pp. 299-299, 1992.

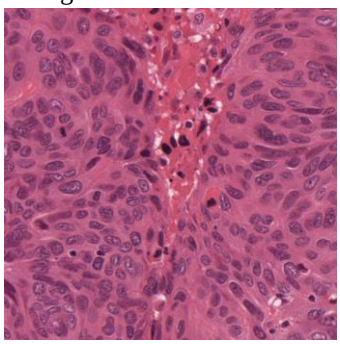
A CNN-based approach for automated segmentation of nuclei in histopathological images

Kaushiki Roy[1*], Debotosh Bhattacharjee[1]
[1]Department of Computer Science and Engineering, Jadavpur University, Kolkata-32, India
[*]kaushiki.cse@gmail.com, debotosh@cse.jdvu.ac.in

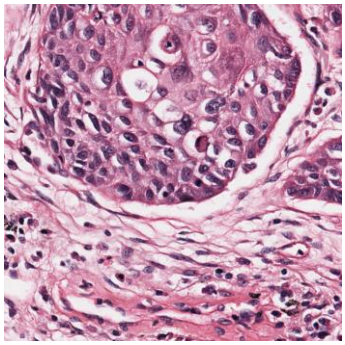
Data Preprocessing

Data preprocessing is necessary to eliminate artifacts which arise due to bad staining, out-of-focus image, variations in colors of H&E images etc. The main preprocessing steps used for this work are listed below-

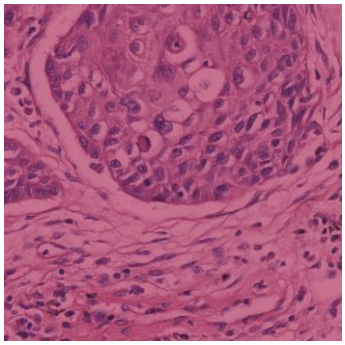
a) Stain Normalization- This is a necessary prior for any H&E stain images because these images are subjected to high variation in appearances due to difference in dye concentrations, lab protocols, source manufacturer, scanners etc. This differences make it difficult for a software which is developed on a particular stain appearance to analyze H&E images. Thus it is necessary to overcome these variations which makes stain normalization an important preprocessing step for any H&E images. In this work, we have performed two normalizations namely Reinhard normalization and Macenko normalization. In Reinhard normalization each image was converted into the same stain space as the source image. Below we show the source image and one of the training image and test image converted to have the same stain appearance as the source.



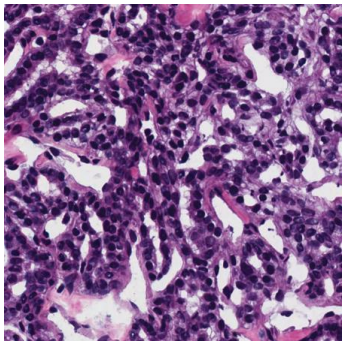
a) Source image



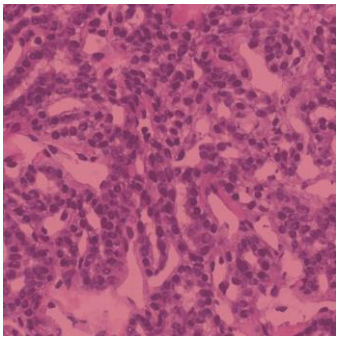
(b)



(c)



(d)



(e)

Figure 1: Train (b) and Test image (d) normalized to have the same stain appearance as (a). (c) and (e) are the results obtained after applying Reinhard normalization on (b) and (c) respectively.

b) Macenko color normalization technique applied on the results obtained after the Reinhard normalization is shown in figure 2. As clearly evident from figure 2, after Macenko normalization the nuclei appears white which are then passed as input to the patch based classifier for nuclei segmentation.

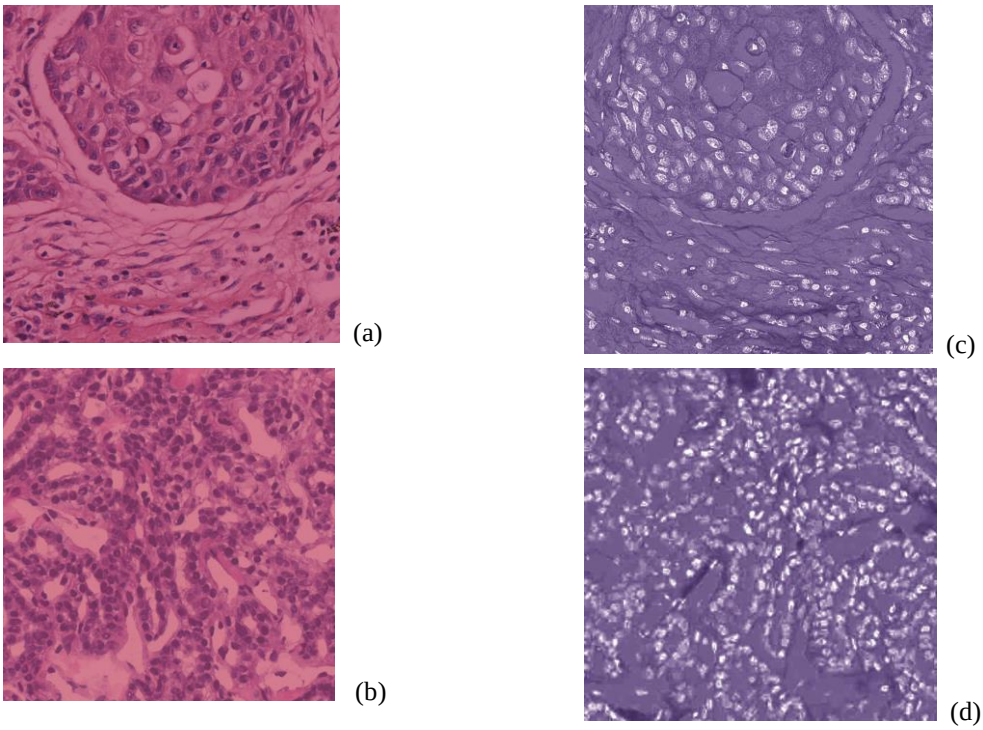
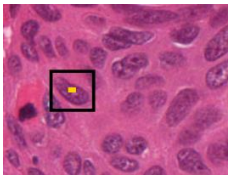


Figure 2: (c) and (d) are the Macenko normalized image of (a) and (b) respectively.

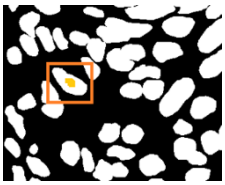
c) Patch extraction and augmentation- The convolutional neural network (CNN) based nuclei segmentation phase consist of a patch extraction and augmentation phase which is necessary to boost up the number of samples in the training dataset. We extracted overlapping patches of size 49x49 from the preprocessed image. Each of these patches were augmented since augmentation presents an effective way to prevent overfitting. The standard augmentation techniques are rotation, cropping, translation. Rotation helps alleviate the rotation variant problem of the input features which allows the convolution layers to extract more discriminating features. Translation on the other hand, shifts the image in X and Y direction but does not extract any new features. Thus we have limited the augmentation to cropping (patch extraction) and rotation only. Each of the extracted patches were rotated with $\frac{Q\pi}{4}$ variations with Q in {0,1,2,3,4,5,6 and 7}. Thus the patches along with their rotations form an augmented patch dataset (APD) which contains much more training samples than the original dataset. The samples in the APD were passed through the proposed CNN for training the model.

Proposed Model for Nuclei segmentation

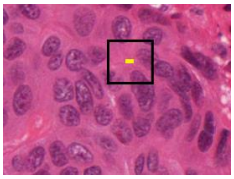
The proposed CNN based framework classifies each of the patches as either nuclei or background pixel based on the class of the center pixel. If the center pixel in a patch is a nuclei pixel then the entire patch is assigned to nuclei class (class 1) else background class (class 0). The class of the center pixel was determined from the annotated ground truth. This idea is illustrated in figure 3. In figure 3a and 3b the patch is centered around a nuclei pixel which makes the entire patch labelled as nuclei class. On the contrary, figure 3c and 3d is centered on a background pixel which makes the entire patch being labelled as class 0 or background class. The model that is used to train the framework is inspired from the ConvNet architecture. Figure 4 gives a diagrammatic representation of the architecture used in this work



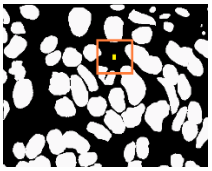
a) Input Patch



b) Patch of the same portion in the Annotated ground truth image



c) Input Patch



d) Patch of the same portion in the Annotated ground truth image

Figure 3: Patch represented by the rectangular regions extracted from the input images and their corresponding ground truth images with dots representing the center pixel. In (a) and (b) we see the patch is centered on a nuclei pixel, thus the entire patch is assigned to nuclei class. On the contrary in (c) and (d) the patch is centered on a background pixel. Thus the entire patch is assigned to the background class.

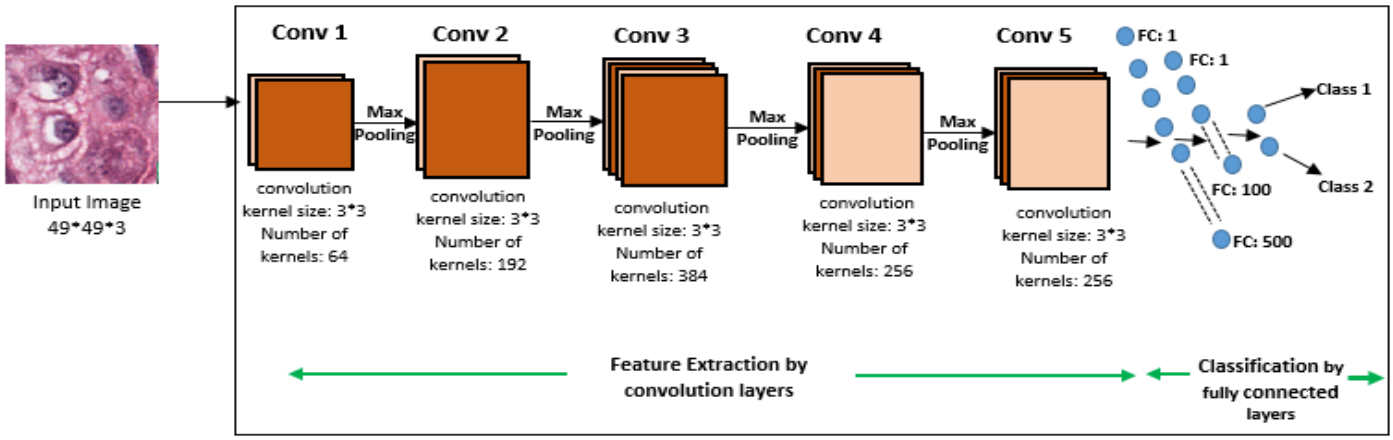
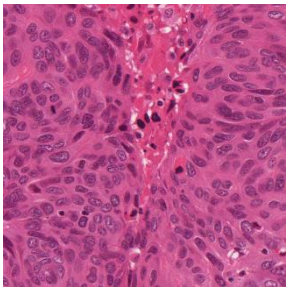


Figure 4: Architecture of the CNN framework used to train the model for nuclei segmentation

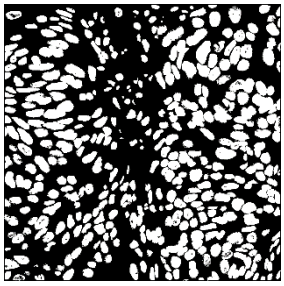
Post Processing

The resultant image obtained by our proposed framework contained some erroneous pixels which were removed by morphological operations and circularity check. We have used marker controlled watershed segmentation for overlapping nuclei separation where binary distance transform has been applied on the output image obtained from the proposed CNN framework to generate the initial seed points. Figure 5a shows an input image, 5b shows the resultant image obtained from our proposed framework after post processing and 5c denotes the color labelling of each nuclei pixel obtained after applying marker controlled watershed algorithm.

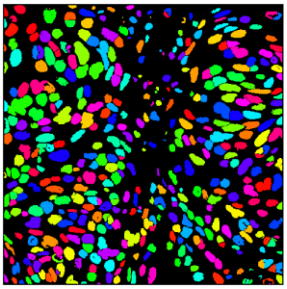
.



a) Input Image



b) Resultant image obtained from our proposed framework after post processing.



c) Labelling of nuclei by marker controlled watershed algorithm

Figure 5: Input image, resultant image and nuclei labelling

Evaluation metrics

The average Aggregated Jaccard index (AJI) obtained for nuclei segmentation is **0.59.** After patch extraction and augmentation 20% of the patches were used for validating the system. The overall training accuracy and validation accuracy of this system are 0.86 and 0.83 respectively. Once the model is trained we passed the test images to it and the results obtained are given in the .mat file. In Figure 6 below we present some of the test images, their segmentation result.

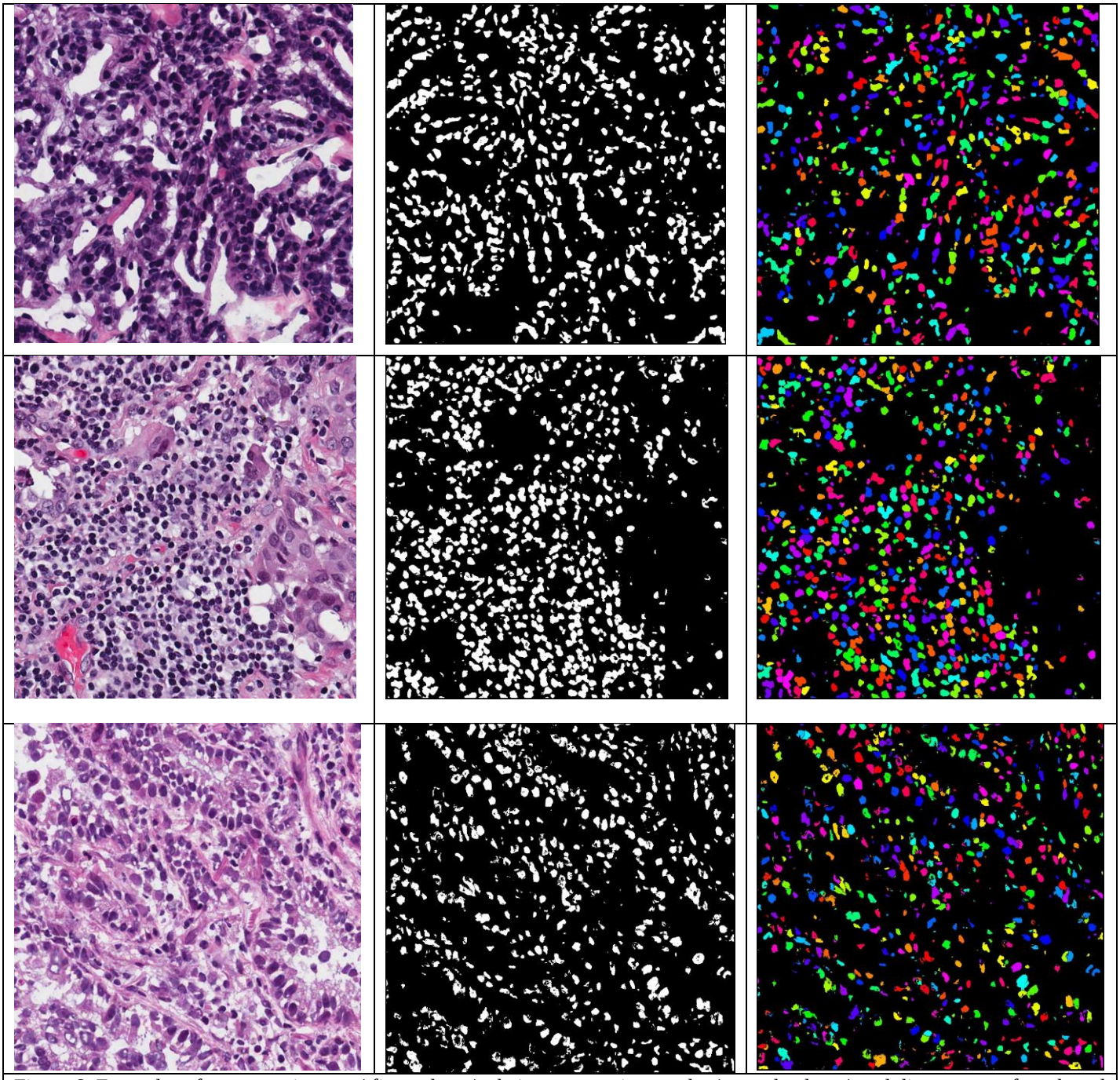


Figure 6: Examples of some test images (first column), their segmentation results (second column) and distance transform based marker controlled watershed segmentation (third column) for overlapping nuclei separation.

1. **Title**:

VISILAB method for MoNuSeg Challenge by Semantic CNN

In this abstract it is explained the processed that have been carried out to approach the MoNuSeg challenge using color transfer and semantic segmentation as the main techniques applied. Finally, some results from a validation dataset are also exposed.

2. Data Pre-processing

The original dataset that was provided was formed by 30 images with the corresponding masks. Each one with 1000x1000 pixels size. The masks format was an XML file, so the first step was to transform these files into binary masks. That was achieved using a script that was also provided along with the data.

In order to augment the available data a preprocessing was done based on

- Rotations of 90º, 180º and 270º
- Mirroring and
- Color standardization based on Macenko's method

After this pre-processing we end with a data based composed of 4650 images

3. Proposed model

With the dataset that was produced in the previous step, a deep learning based computer vision technique was applied to approach the challenge.

This technique makes use of labelled data in which several categories are distinguished. In this case, these were the categories that are used in this problem:

- Tissue: normal region that could be considered as the regular background in the images
- Nucleus: segmented regions that correspond to the nuclei that is wanted to be segmented in the images.

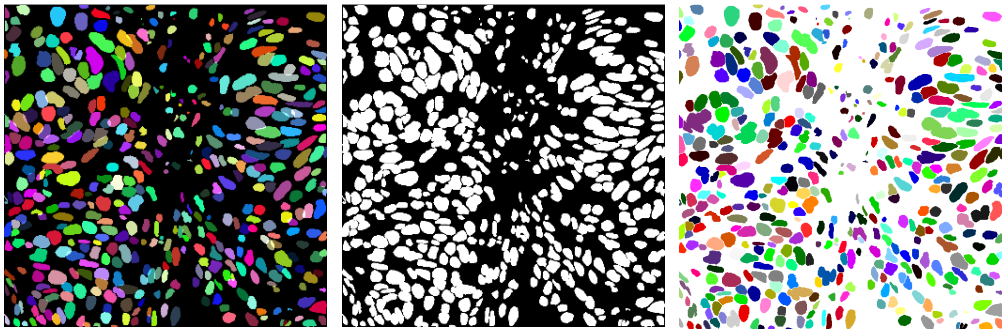
The training process was carried out using a pretrained network from VGG, originally trained for 32 classes, applying transfer learning to retrain the network to the specific classes of this problem.

The network was trained with a NVIDIA Quadro P4000 with 8GB of VRAM, taking 35 minutes to complete the process.

4. Post-processing

The masks that are produced using the semantic segmentation are binary region predictions with these classes, so no information about individual nucleus is provided. This has to be extracted using further post-processing of the results. To achieve this, a watershed segmentation algorithm was applied, to split the global regions segmented as nuclei into individual nucleus, to label each one with a different identifier.

In Figure 1 shows both a groundtruth mask (GM) coloured and binary and the corresponding predictions. Each colour means a different segmented nucleus after applying the proposed processing.



a) Coloured GM b) Binary GM c) Predictions (automatic segmentation)

Figure 1. Groundtruth mask (coloured and binary) and the segmentedimage

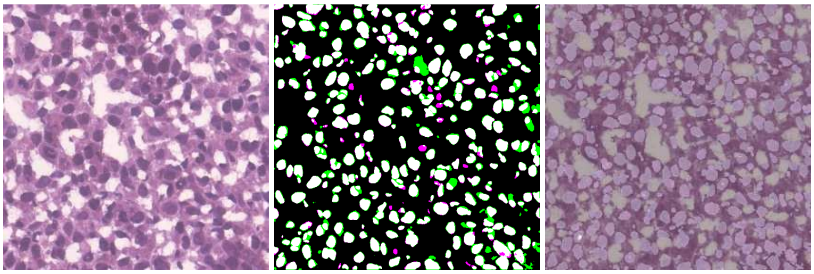
5. Results

Besides the predicted binary masks that are required for result submission, for the validation dataset these masks are also compared with the groundtruth, in order to check the performance of the method. Figure 2 shows samples for these results. 1st column the original images, 2nd column the output where, in green the False Positives (wrong nuclei regions predicted) are highlighted, while False Negatives (missing nuclei regions not predicted) are stated in magenta. And 3rd column the nuclei are highlighted for visualization purpose.

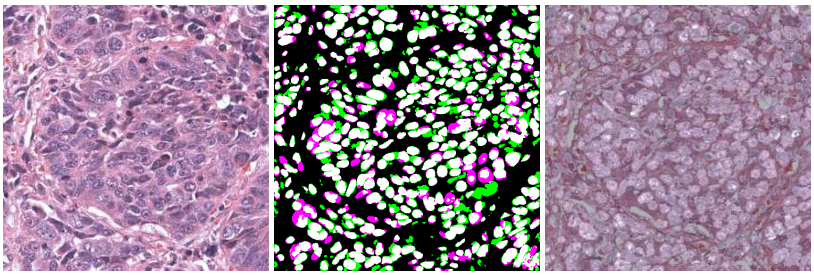
The metrics that are employed for performance evaluation are the following:

- Accuracy: indicates the percentage of correctly identified pixels for each class.
- Intersection over Union: also known as the Jaccard similarity coefficient (AJI), it is a statistical accuracy measurement that penalizes false positives. Abbreviated as IoU, it calculates the ratio of correctly classified pixels to the total number of ground truth and predicted pixels.
- Boundary F1 Score: it is a contour matching score that indicates how well the predicted boundary of each class aligns with the true boundary. It tends to correlate better with human qualitative assessment than the IoU metric.
- Sensitivity: or recall, is the proportion of true positive pixels in comparison with the actual correctly classified pixels in the image

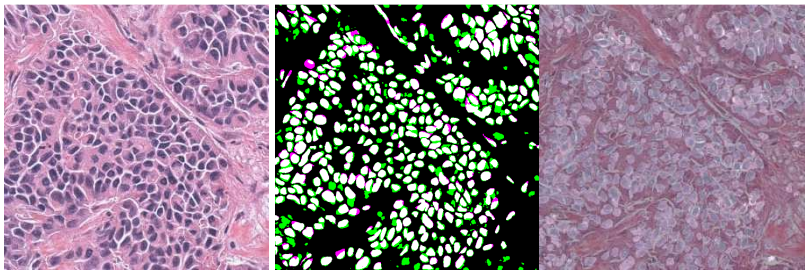
- Specificity: is the proportion of true negative pixels in comparison with the negative classified pixels in the image



a) Example of good case



b) Example of bad case



c) Example of average case

Figure 2 .- Results of the algorithm proposed

The results obtained for the validation test are:

- Mean accuracy: 0.9
- Mean boundary F1 contour matching Score: 0.945
- Sensitivity: 0.92
- Specificity: 0.885
- AJI: 0.896

Deep U-net based Multi-Organ Nuclei Image Segmentation

Rank- 30, Team Name- Sabarinathan
Sabari Nathan D[1], Saravanan R[1], Praveen Koduganty[1]
[1]Cognizant Technology Solutions India Private Ltd, India

1. Pre-Processing

a. Normalized the images dividing them by 255 and bring the pixel values between 0 and 1.
b. Binarized the mask and segregated the intersection between nuclei cells. We created three channels target output Image. Channel 1 contains foreground i.e., Nuclei cells, channel 2 contains the intersecting edges of nuclei cells and channel 3 contains the background
c. Resized all the images as 256X256.
d. Used shearing, rotation, zoom, width and height shift, horizontal and vertical flip augmentation with the below values
    i. shear_range=0.5
    ii. rotation_range=45
    iii. zoom_range=0.2
    iv. width_shift_range=0.2
    v. height_shift_range=0.2
    vi. horizontal_flip=True
    vii. vertical_flip=True
e. Additional **Augmentation Techniques** used to increase the given training dataset

Augmentation Technique	Parameters
Elastic Deformation of Image	Alpha =10, Sigma =1
Quantification of histochemical staining by color deconvolution	Low boundary for augmentation multiplier=0.7 High boundary for augmentation multiplier=1.3
Random Crop	Three random patches were cropped from the image(256x256)
Median Blur	Kernel Size =5
Speckle Noise	Noise Variance is 0.04
Salt and Pepper Noise	Noise density 0.05
Shuffling the RGB Channel	Shuffling the position of RGB (2, 0, 1)
Adding the brightness	Alpha= 0.5
Rotating the Image	90,180,270
Clahe- contrast limited adaptive histogram equalization	Clip Limit=2 , Kernel Size =(8,8)
Shifting the HSV Channel	Hue Sift= 230, Saturation=10, 10
Grayscale	Converting the image into grayscale

2. Model details

a. As given below Fig.1, U-net Architecture is used for Nucleus Segmentation. The encoding path contains (3x3) kernel convolutional followed by "Elu" (Exponential Linear Unit) layers and Keras "he normal" used for kernel initialization. Max pooling (pool size 2x2) layer is added after every two convolutional layers. Also, a dropout layer is added with 0.1 probability between every two convolutional layers. The decoding path follows the similar structure as encoding but up-samples the input. The output is the 256x256x3 layer with Softmax activation.

b. Loss Function: A combination of categorical cross-entropy and dice coefficient is used. 50% categorical cross entropy loss of the image + 30 percentages for channel 1 dice coefficient loss + 20 percentages for channel 2 dice coefficient loss

c. Adam optimizer is used with learning rate as 1e-4, default momentum value (0.9), batch size as 16 and 100 times iteration (epochs) to train the model. 20 percentage dropout is used to improve the model prediction.

d. Validation Loss is used is used to evaluate the model

e. The output is taken from Channel 1 contains the segmented nucleus. Channel 2 helped the model to separate out connected edges between the nuclei cells.

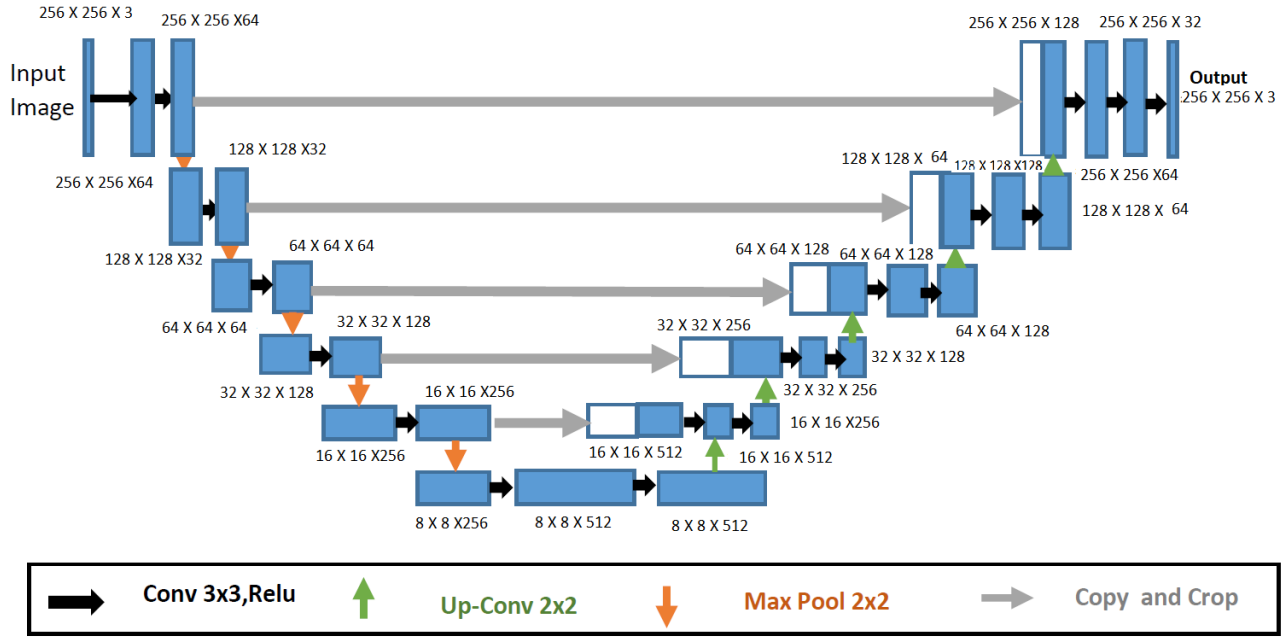


Figure 1. The architecture of U-net network. The size of each layer is shown in height, width and channels.

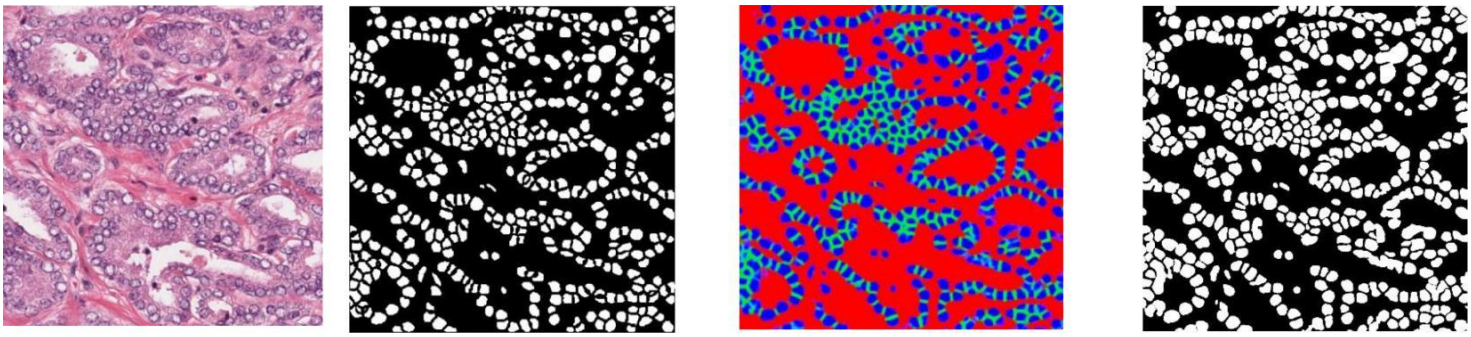
2. **Post-processing**: Not Applicable

3. **Computational Complexity**
    a. Xeon E5-2686 v4 processor with 16 GB RAM and NVidia k80 GPU with 16 GB memory."
    b. Training time- 96Hours for the specified model
    c. Testing time- In 53.415 sec for each image.

4. **Code Release**- "NOT APPLICABLE"

5. **Result –** The dataset is divided into two parts training (70%) and validation (30%). AJI Score, Dice score & F1 score are the accuracy metrics for the Nuclei segmentation.

	Validation Data	**Training Data**
AJI	0.4077	0.4227
Dice	0.75297	0.80147
F1 Score	0.7670988	0.8139417



(a) (b) (c) (d)

Figure 2. Nuclei Segmentation Results.
a) Tissue Image, b) Ground Truth, c) output of Chanel 1+ Chanel 2 , d) Predicted Output taken from Chanel 1

Multi-organ Nuclei Segmentation with Fully Convolutional DenseNet

Zihan Wu

Pre-processing:

1. Creating 3-value map, where 0 represents background pixels, 1 for nucleus pixels and 2 for overlapping part of multiple nuclei.
2. Enhancing contrast to expand the difference between foreground (nuclei or cell tissues) and background.
3. Random horizontal and vertical flip, color jitter and random grayscale transformation for data augmentation.

Proposed Model:

FC-DenseNet[1] which combining DenseNet and U-Net.

Label: 3-value maps

Loss function: NLLLoss2d

Optimizer: RMSprop

Framework: PyTorch

Post-processing:

1. Removing noise from predicted map. E.g. considering 0 pixels surrounded by 1 pixels as mispredicted 0 pixels. With three labels, the actual operation is more complicated.
2. Separating overlapping nuclei according to the overlapping part predicted in predicted map (pixels with predicted label, 2).

3. Morphological transformation to make the boundaries of each individual nuclei more distinguishable.
4. Making final predicted map by number each individual nuclei with a unique positive integer.

[1]S Jégou, M Drozdzal, D Vazquez, A Romero, Y Bengio. The One Hundred Layers Tiramisu: Fully Convolutional DenseNets for Semantic Segmentation[J]. Computer Vision & Pattern Recognition Workshops , 2017: 1175-1183

Monuseg

Guanyu Cai, Xiaojie Liu, Yuqin Wang
Tongji University
caiguanyu@tongji.edu.cn

August 13, 2018

I. Pre-proessing: deformation and color normalization

One of the challenges of this segmentation task is that there are only 30 training data, therefore data augmentation is necessary for the invariance and robustness of the network. Besides regular data augmentation such as rotation, width and height shift, filp and reflection, image deformation technique is employed in these training data. This method first divides an image into 16 equal regions and generates smooth deformations on these intersections using affine, similarity and rigid transformations, which are able to simulate cell diffusion and movement well. Each image is deformed about 80 times, after which color normalization is applied.

II. Methods: End to end compositive model

Our novel idea for the segmentation task is to construct an end-to-end compositive image segmentation model based on CNN that directly classify the foreground and background for each pixel, therefore all the input image are resized to 1024*1024 instead of being cut into patches.The whole model includes 3 end-to-end networks, U-Net, U-Net-v2 and FCN-8s. Therefore, for each pixel, we can get three estimates and the final probability is the maximum of the absolute values of the differences between the three estimates and 0.5.

i. U-Net

The architecture of u-net prototype is symmetric, consisting of a contracting path and an expansive path. The network is composed of 4 downsampling layers and 4 upsampling layers and the numbers of channels of convolutional layers is (64,128,256,512,1024).

ii. FCN-8s

The architecture of FCN-8s is based on VGG-16 network. The fully connected layer of VGG-16 is modified to convolutional layer. To improve localization performance of the model, skip architecture is employed and features of different layers are fused by concatenation and classification by a ą̋score layerą́ consisting of an 1*1 convolution. All the parameters of ą̋score layersą́ is initialized by a method called He_normal initialization instead of the usual way, where the parameters are trained on ImageNet and fine-tuned on the source dataset.

iii. U-Net-v2

U-net-v2 is a variety of u-net, adding a downsampling layer and an upsampling layer and the numbers of channels of convolutional layers is modified to (32,64,128,256,512,1024).

iv. Training

For training and verification, we divide the dataset. In order to verify the images of all organs, we selected one image from each kind of organs respectively to form a validation set,

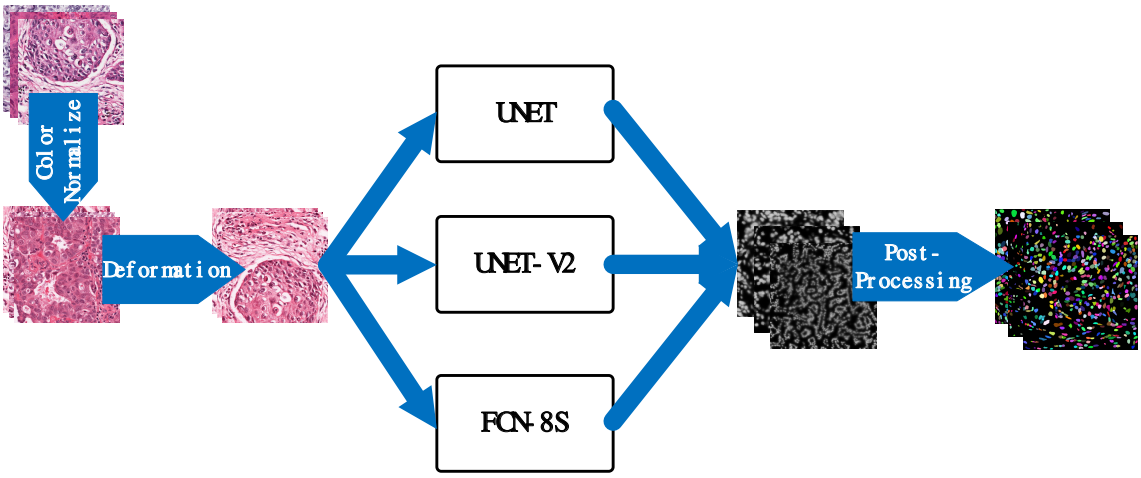


Figure 1: *Flow chart of the method we use*

while the remaining 23 images serve as training sets. The dropout technique is applied in some convolutional layers of u-net and u-net-v2 to prevent overfitting.
The entire model is based on keras, trained on two Nvidia GeForce GTX 1080ti graphics cards. The initial learning rate of U-Net and FCN-8s is set 1e-4 while that of U-Net-v2 is set 1e-5. When the loss of networks doesn't descend any more during 5 epochs, the learning rate is reduced by 10 times. Each model is trained 100 epochs, taking a total of 13.25 hours. It takes about 13 seconds to predict 30 images in the entire data set.

III. Post-processing: Region growing

First, 1024*1024 predicted images are converted to 1000*1000 by using method called bicubic interpolation. Then, nuclei were seeded by thresholding the inside class probability map at 0.7. Each seeded nuclei grows iteratively to all directions and is stopped growing further when the probability difference of adjacent pixels is less than 0.05.

IV. Results

Preliminary AJI results are given in Table 1.

Table 1: *AJI of the model*

	AJI(training data)		AJI(testing data)
0	0.2457218	16	0.384301
1	0.2929523	17	0.3847792
2	0.1624888	18	0.4072807
3	0.2418734	19	0.3269733
4	0.2618208	24	0.2020056
5	0.2320296	26	0.4948766
6	0.2559795	28	0.2600952
7	0.3697756		
8	0.2910459		
9	0.2608115		
10	0.0862869		
11	0.4342301		
12	0.3397802		
13	0.2857899		
14	0.2371869		
15	0.3793569		
20	0.264521		
21	0.2936785		
22	0.3300699		
23	0.4115399		
25	0.3262508		
27	0.304777		
29	0.252083		
max	0.4342301		0.4948766
ave	0.2852195		0.3514730