Nuclei detection based on Fully Convolutional Networks

Eduardo Rocha de Andrade
UNICAMP

1 ABSTRACT

The work hereby presented intends to solve Kaggle's 2018 Data Science Bowl competition, which consists of segmenting and instantiating cell's nuclei on microscope images. Accurate nuclei segmentation is paramount to drug and disease researches and diagnostics. In this paper, the main obstacles to precise automatized nuclei segmentation are studied and solutions are proposed. This work relies on Fully Convolutional Networks based on U-Net architecture to predict the nuclei, borderless nuclei and overlapping borders triad, which are the inputs to post-processing algorithms, such as watershed and random walker. Furthermore, the usage of transfer-learning is exploited by using competition-winning networks, e.g. Xception, as U-Net's encoder. Finally, an U-Net's decoder based on the Squeeze-and-Excitation and Inception concepts was proposed, which consistently improved performance and decreased training time.

2 INTRODUCTION

Nuclei identification is one of the first steps in many biological and medical researches. Accurate segmentation is very crucial to obtain morphological statistics about tissues, pathologies and other biological structures, which are required for researches, medical diagnostics and treatments. Normally, segmentation is done manually by a pathologist or experienced biologist, which is a time-consuming task, costly and does not guarantee reproducibility since it can differs significantly depending on the imaging technique used [1]. By automating nuclei segmentation, it is expected to reduce researches' costs and enable professionals to better allocate their time into more meaningful tasks, such as analyzing results, performing tests and finding potential cures. Consequently, leading to improved drug testing, reducing their time-to-market and allowing better treatments.

Medical imaging has been considerably beneficed with the ever increasing advantages of Deep Learning, particularly Fully Convolutional Networks (FCN) [2], which has consistently replacing traditional methods that usually rely on handcrafted features and morphological characteristics. One of the main breakthroughs was U-Net, proposed by Ronneberger et. al. [3], which consists of a FCN comprised of a down-sampling or encoder part and a up-sampling or decoder part forming an "U" shape . The encoder is a common convolutional network architecture where the number of feature maps increase as their size reduces with pooling operations. Conversely, in the decoder the number of feature maps decreases while their sizes are increased by using transposed convolutions. Furthermore, U-Net uses a weighted cross-entropy loss to deal with cluttered cells and skip connections from encoder to decoder to improve the up-sampling's precision.

Following the success of U-Net, many other solution based on similar schemes were proposed; some leading to important improvements. The main contributions were an improved weighted loss function proposed by F. Guerrero-Penna et. al. [4]; contour aware network DCAN [5]; introduction of watershed post-processing using predictions (nuclei – contours) as markers [6] and deep watershed transform network [7] that predicts an energy function, which is thresholded to obtain the individual masks.

The next sections introduce the dataset, evaluation metrics, the methodology proposed to solve the given task , the outcomes obtained and, finally, concludes analyzing and benchmarking the results.

3 DATASET

The dataset contains a large number of microscope images of cells under a different set of conditions varying in cell type, magnification, and imaging modality (bright-field vs. Fluorescence). For each image, there are N segmented binary masks where N is the total number of cells in the image. Each mask can only contain one nucleus and masks are not allowed to overlap. Figure 1 shows an example from the dataset.

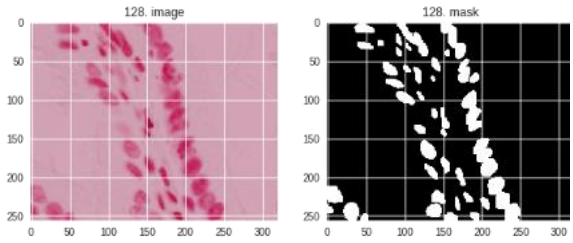

Figure 1: An example from the dataset. The left picture shows a typical training image, while the annotated mask is displayed on the right.

The training set contains approximately 700 RGB images with different dimensions ranging from 256x256 to 1040x1388 pixels, while the test set contains 3019 images with dimensions ranging from 205x231 to 1040x1388. Due to the small training set size, an external dataset referred to as V9, which consists of approximately 3000 256x256 crops from images of three datasets (Kaggle + TCGA + Celltracking) was also used in some experiments.

4 EVALUATION METRICS

The competition uses the mean average precision at different intersection over union (IoU) thresholds. Where thresholds vary in steps of 0.05 from 0.5 to 0.9.

$$IoU(A,B) = \frac{A \cap B}{A \cup B} \qquad Eq.1$$

This means that if a prediction has IoU (Eq. 1) with the ground-truth mask greater than the threshold, it is considered a match. If the prediction is correct, a true positive is computed, otherwise a false positive is incremented. False negatives are accounted for all mask that have not been predicted. Then, a precision value is computed for each image as follows:

$$\frac{1}{|thresholds|}\sum_{t}\frac{TP(t)}{TP(t)+FP(t)+FN(t)} \qquad Eq.2$$

Where TP, FP and FN are the True positive, false positive and false negative, respectively, and are computed for each threshold. Finally, the competition metric is the mean taken over the individual average precisions of each image in the test dataset.

5 METHODOLOGY

In this sections the methodology utilized is explained in details, describing the principal networks and algorithms used and the tests performed.

The implementation was made in Python and native Tensorflow, except for the pre-trained encoder versions that were implemented using an hybrid Keras-Tensorflow approach (encoder from *tf.keras.applications*, while the rest implemented in Tensorflow native/slim).

5.1 Data Pipeline

The data pipeline was implemented differently depending on the dataset that was used. For V1 (Kaggle's default training set) the images and masks were resized to 256x256 pixels, and saved to disk. During training, the data was dynamically loaded into memory, in order to overcome memory constraints. For the V9 training set, the images were already cropped to 256x256 pixels, therefore no resizing was needed.

Data augmentation was also crucial during training since nuclei images can vary a lot, specially under different medical imaging modalities. Moreover, the training set is considerably small, increasing even more the role of data augmentation in this task. Several techniques were used consisting of rotations, horizontal and vertical flips, channel shuffle, zoom and shear.

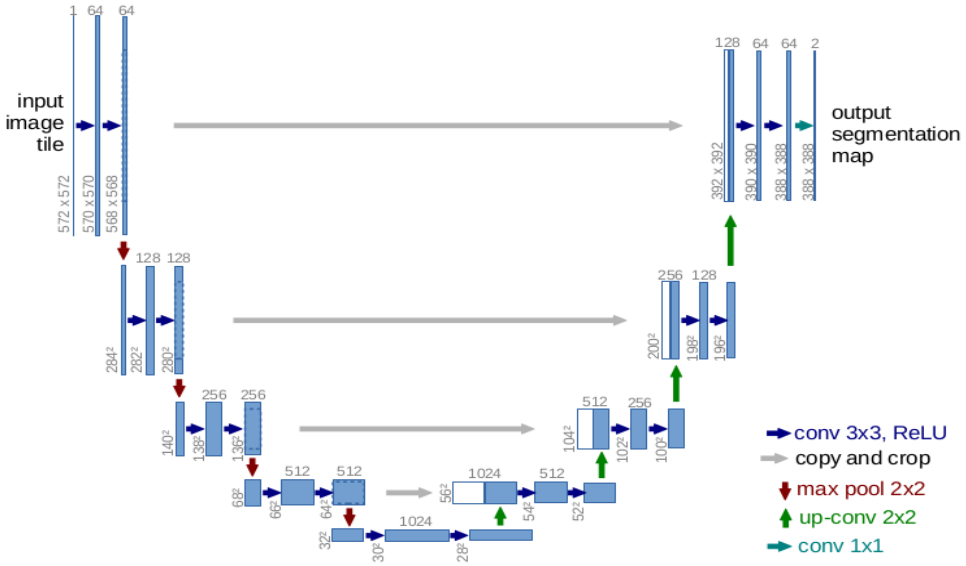

Figure 2 - Vanilla U-Net. The blue boxes represent the feature maps at each stage. It should be noticed the usage of 'valid' padding convolutions and cropping of the skip connections, which is believed to be a weakness of the model. Image from [3].

5.2 Network Architecture

Although the U-Net model has been a breakthrough in image segmentation, there are some key aspects that could be improved. One particularity of the U-Net architecture is the usage of 'valid' padding convolutions, which consequently, reduces the feature maps' size and loses information at the borders too quickly. Furthermore, the lack of padding makes the model asymmetric, requiring an additional crop operation at the skip connection, as can be seen in Figure 2.

Another possible limitation of the vanilla implementation could be its depth. Since the network is considerably shallow, it may lacks entropic capacity to fit the data properly, leading to a biased model. Thus, using a more modern and deeper model architecture, such as *Xception* [8], *ResNet* [9], etc.., could lead to improvements. Therefore, three modifications on the regular architecture are proposed and are described next.

5.2.1 Plain U-Net

In the U-Net implementation proposed in this paper, the 'valid' paddings are changed to symmetric, with hope to retain even more information at the borders. Consequently, the model becomes symmetric and the encoder-decoder pair share the same spatial dimensions at a given stage, dispensing the need for cropping at the skip connection. Another modification from the vanilla model was the addition of Batch Normalization layers after every convolution, which help the gradients flow through the network, improving convergence and reducing training time [14]. The network weights are initialized using a Xavier-Glorot scheme proposed in [10].

5.2.2 Pre-trained Encoder

Deep learning, in particular convolutional neural networks, has not only been a disruptive technology in computer vision due to its outstanding results, but there are also many other convenient traits that comes along. One of which is transfer learning, which consists of re-using the weights of a network, which can be trained in a different task entirely since low-level features are usually shared across many applications. Hence, this technique enables the work of intensive computational power and big datasets to be exploited by others, which is particularly important for segmentation tasks since pixel-wise labeling is very costly and, therefore, big datasets are hard to come by.

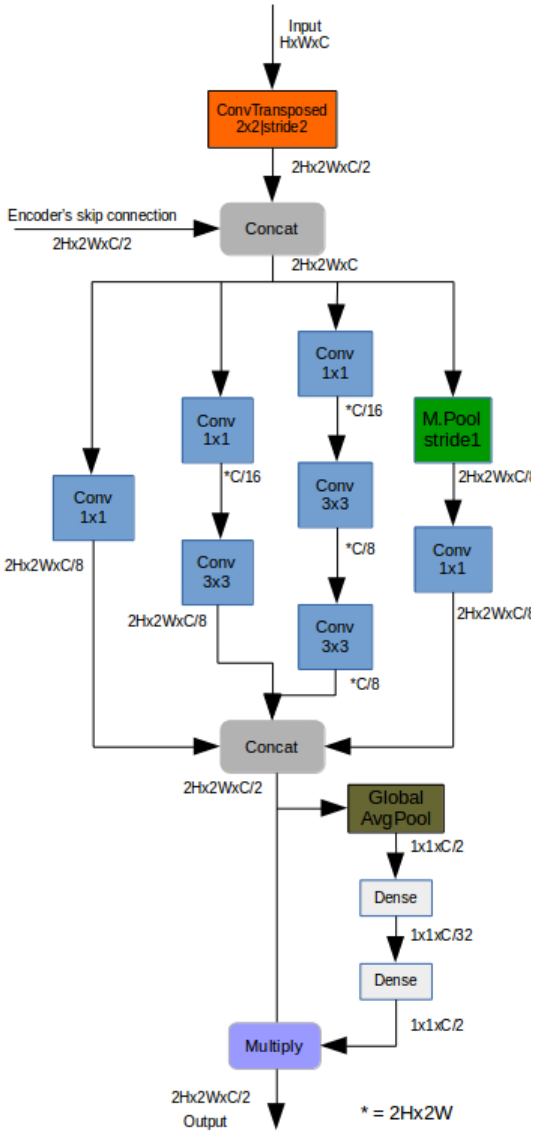

Figure 3 - Up-sampling block of proposed U-Net's decoder. The up-sampling itself is performed by the transposed convolution, which is followed by an Inception-like module that performs multi-scale convolutions. Finally, a Squeeze-and-Excitation block is applied to the output. Note: Batch Normalization applied after every convolution, omitted for the sake of clarity.

The idea of using an U-Net's pre-trained encoder has already been explored by Iglovikov et. al. in [11], where a VGG11 network pre-trained on ImageNet was used as encoder, leading the authors to win the Kaggle: Carvana Image Masking Challenge. In this paper, however, the idea is to exploit even further the advantages of transfer learning by using a more modern architecture than VGG. Thus, the *Xception* model proposed by F. Chollet in [8], which extends the idea of the Inception module [12] by using depth-wise separable convolutions is chosen as encoder due to its good performance and reduced model size.

There is only one caveat that must be taken into account before using any model of the Inception family as encoder. These networks use 'valid' padding in the *stem* block (refer to [12]), which breaks the encoder-decoder symmetry. This issue is

addressed by changing to 'same' padding convolutions. Note that altering the padding scheme only changes the output size of each stage, leaving the weights unaltered and permitting the transfer learning usage.

5.2.3 Augmented Decoder

Inspired by the recent improvements presented by Hu et. al. in [13], which won the ILSVRC 2017, the Squeeze-and-Excitation (SE) block is applied to the decoder as a matter of study. This block is added in parallel after every convolutional block and consists of a global average pooling (squeeze) followed by two fully-connected layers (excitation), whose activation functions and outputs shapes are *ReLu (1x1xC/r)* and *Sigmoid (1x1xC)*, respectively. The output is then multiplied by the convolutional block's output, whose shape is HxWxC, where H, W, C and *r* denotes height, width, channels and dimensionality reduction ratio, respectively. The SE block can be at the bottom of Figure 3.

The goal of the squeeze operation is to retain global information, which is usually disregarded in network's lower levels due to small convolution's receptive fields. The fully-connected layers map the inter-channel dependencies, which adaptively rescales the convolution's output [13].

The proposed decoder is comprised of several up-sampling blocks of Figure 3. This block is comprised of an Inception-like module, to investigate the effects of multi-scale convolutions on the decoder, followed by the SE block. Although not shown in Figure 3, Batch Normalization layers are applied after every convolution.

5.3 Multi-head Architecture

Although, in general, U-Net performs well for nuclei segmentation, it have a well known difficult to predict individual nuclei in conglomerate cell blocks. This occurs since, many times, the binary masks used as labels are touching or very close to, which fools the network into thinking the whole block is a single cell.

This issue is mitigated in [5] by extending the traditional U-Net to also predict the cell's contour, which is then subtracted from the nuclei prediction yielding borderless masks. The problem with this approach is that the prediction overlap with the ground-truth is reduced, which penalizes the model specially at high IoU thresholds. Furthermore, this approach is particularly harmful to small nuclei, whose border to interior ratio is similar in terms of area. In order to overcome this, Cui et. al.[6] uses the borderless nuclei as markers for a watershed post-processing algorithm, retaining the predicted nuclei's full size. However, from Kaggle's discussions forum, it has been observed that predicting contours may not be the best option, since the network usually fails to predict contours in difficult areas, such as conglomerate cells, which is where contours have utmost importance.

In this implementation, we neglect the easy contours where the nuclei are isolated from others and focus on predicting only the touching borders between nuclei, with hope to improve the model's performance on conglomerate cell blocks. Furthermore, a prediction for shrunk nuclei is also implemented; albeit theoretically redundant, it is believed that this prediction can be useful when the model fails to predict the touching borders.

5.3.1 Pipeline changes and Post-processing

In order for the U-Net to predict multiple channels, modified masks must be fed to the network. The touching borders and shrunk nuclei masks are generated using morphological operations on the original masks. The original image and the resulting multi-channel mask, which is used as label, are displayed in Figure 4. Once the prediction is made, cell instantiation is obtained by applying the watershed algorithm on the original mask channel, using the result from the subtraction of the other two channels (borderless nuclei – touching borders) as markers.

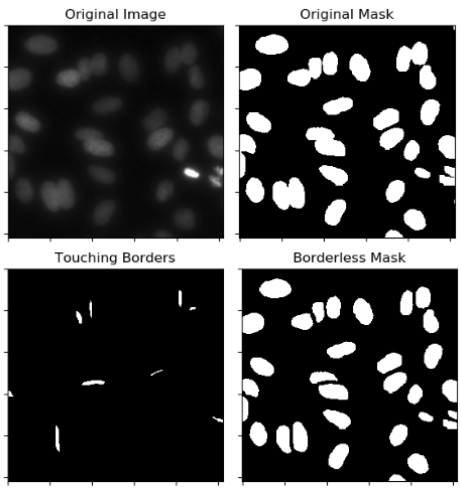

Figure 4 - Final mask used as label of shape (H,W,3). Where the channels are respectively the original mask, borderless mask and the touching borders.

5.4 Activation Function

Three different activation functions were experimented in this work: ReLU, ELU [15] and SELU [16]. From preliminary tests , its was observed

that even though the final performance of three was similar, SELU improved training speed and allowed the network to be more insensitive to the learning rate choice. Whereas, the other two functions required a more carefully chosen learning rate determined through a grid search scheme. This observation seems to confirm what is claimed by Klambauer et. al. in [6].

5.5 Loss Function

In the original U-Net paper [3] a weighted cross-entropy function was used as loss. From what has been observed in this tasks, this loss, even in its weighted form, suffers from the huge data unbalancing presented in our task, specially when predicting touching borders. In order to tackle this issue, a soft-dice loss, defined in Equation 3, can used. The problem, however, is that this loss tends to generate overconfident predictions, which may lead to overfit.

$$dice = 1 - 2\frac{A \cap B}{A \cup B} \qquad Eq.3$$

Finally, in order to exploit the benefits of both losses, a third implementation is considered, consisting of a simple sum of both functions, allowing the model to deal with unbalanced data, while refrained to make overconfident predictions.

5.6 Inference

As previously mentioned, there are 3019 images in the test set with different sizes. Since the proposed networks are fully convolutional, it was possible to infer each image in its original size. The only care needed was to guarantee that the image size was multiple of 32, which is the number of pooling step in the encoder. Otherwise, the image was symmetrically padded for inference and then cropped back to the original size.

6 RESULTS

In this section, the results of several experiments and studies are presented. It should be noted that all experiments were run on the V1 dataset using the Adam optimization algorithm [17] with a learning rate decay schedule of 25% every 3 epochs and SELU activation functions. Furthermore, all experiments, except for the final model, were evaluated on Kaggle using resized images during inference in order to save time, even though it is known to degrade performance.

6.1 Loss function analysis

The first experiment performed intended to verify how the loss function choice affects the model performance. Three *Xception* U-Nets of section 5.2.2 with different loss functions: cross-entropy, soft-dice and the combination of both were trained for 20 epochs with learning rate of 0.001. The results are displayed in Figure 5, confirming the previous assumption that a combination of both losses could improve the performance. The enhancement is also present when evaluated with the mAP score on Kaggle, which is displayed in Table 1.

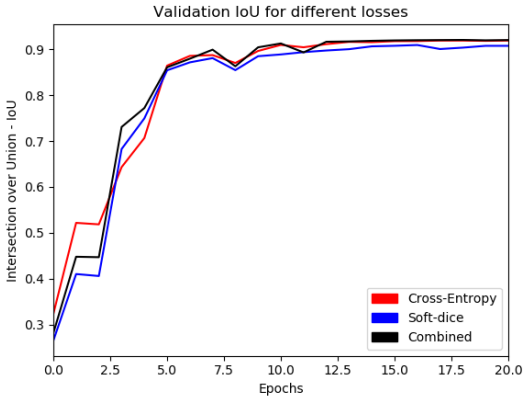

Figure 5 - IoU performance for different loss functions.

Table 1 - mAP score evaluated on Kaggle for different loss functions.

Loss function	mAP score
Cross-Entropy	0.358
Soft-dice	0.320
Combined	0.361

6.2 Network architecture impact and multi-head predictions

Another important study realized in this work was assessing the impacts of transfer learning and a modern encoder architectures on the given task. Furthermore, the effects of recent published concepts, such as the Squeeze-and-Excitation block, were investigated in the network's decoder.

Figure 6 shows the results for the three networks proposed in Section 5.2: plain U-Net, *Xception* encoder and *Xception/InceptionSE* encoder/decoder. The first was trained for 20 epochs with a learning rate of 0.001. Whereas the other two had a two stages training, first freezing the encoder weights and training for 5 epochs with a learning rate of 0.001, followed by a fine-tuning of all layers for 15 epochs with a learning rate of 0.0001. All networks were implemented for multi-head predictions in order to

assess their capabilities on both nuclei and touching borders masks. Additionally, all models used watershed post-processing and the sum of cross-entropy and soft-dice as loss function.

Although the validation IoU looks similar for all the networks, the improvements of using a pre-trained encoder are evident, specially for the touching borders predictions. Furthermore, it seems that the augmented decoder architecture led to performance gains and reduced training time. The Kaggle evaluation for the models is displayed in Table 2, where the differences are more accentuate.

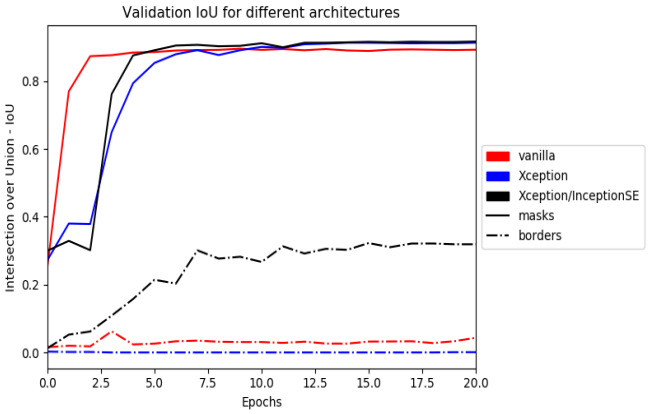

Figure 6- IoU performance for different network architectures.

Table 2 - mAP score evaluated on Kaggle for different network architecture.

Architecture	mAP score
Plain U-Net	0.273
Xception	0.322
Xception-InceptionSE	0.420

7 FINAL SUBMISSION

For the final submission, the ***Xception-InceptionSE*** model was retrained in the V9 dataset using the same scheme of section 6.2 and the proposed combined loss function. Unfortunately, due to an implementation error, it was not possible to use original image sizes during test inference. This issue significantly harmed the final submission, which scored **0.469** on Kaggle's mAP evaluation. Figure 7 and 8 shows some challenging, yet interesting predictions made by the model.

8 CONCLUSION AND FUTURE WORK

It was presented a fully convolutional model that can be used to solve complex image and instance segmentations problems. Furthermore, it was shown how transfer learning, modern encoder/decoder architectures and a hybrid loss function can exploited to boost performance even in a small training set with high unbalancing. Finally, it was demonstrated how flexible the U-Net architecture is and how it can be used to predict the touching borders between nuclei, which is later used for watershed post-processing.

It was stated by the competition organizers that the human mAP is around 0.550 for a biologist; considering this benchmark , the 0.469 score obtained in this work is somewhat satisfactory. However, there were many strategies and ideas that could not be explored due to time, some of which are: more aggressive data augmentations with blurs, gaussian noise, pixel dropout, emboss, piecewise affine, perspective transform and brightness and contrast changes; generation of synthetic conglomerate cell blocks to improve touching border predictions, ensembles of networks, random-walker as post-processing algorithm and different networks as encoders. Consequently, there is still much room from improvement.

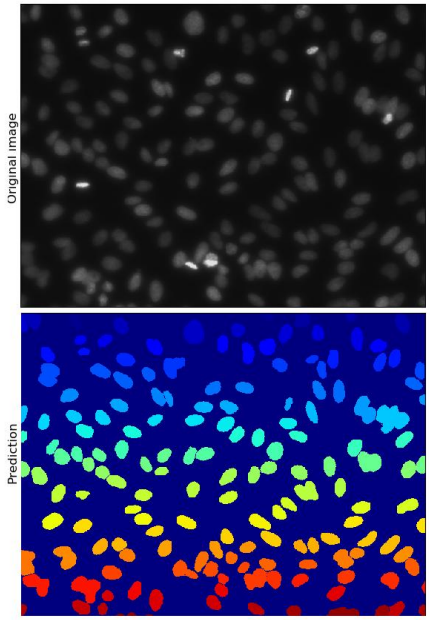

Figure 7 - Example of a prediction made by the final model. Note how some touching cells are correctly separated while others are unified in a single block.

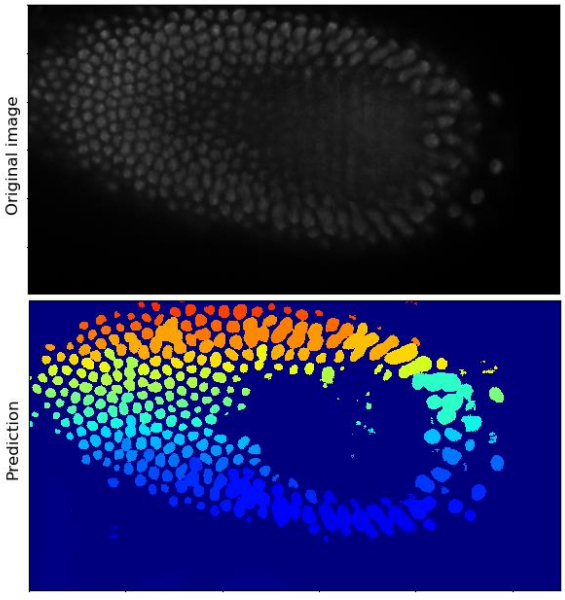

Figure 8 - Another prediction example with many nuclei. Note how the interior of the circle is ambiguous.

9 REFERENCES

[1] X. Xu, Q. Lu, Y. Hu, L. Yang, S. Hu, D. Chen, Y. Shi, "Quantization of Fully Convolutional Networks for Accurate Biomedical Image Segmentation" , *arXiv preprint arXiv:1803.04907* [cs.CV], Mar. 2018.

[2] E.Shelhamer, J. Long, T. Darrel, "Fully Convolutional Networks for Semantic Segmentation". *arXiv preprint arXiv:1605.06211* [cs.CV], May 2016.

[3] O. Ronneberger, P. Fischer, T. Brox, "U-Net: Convolutional Networks for Biomedical Image Segmentation". *arXiv preprint arXiv:1505.04597* [cs.CV], May 2015.

[4] F.A.Guerrero-Pena, P.D. Marrero Fernandes, T.I.Ren, M.Yui, E.Rothenberg, A.Cunha., "Multiclass Weighted Loss for Instance Segmentation of Cluttered Cells". *arXiv preprint arXiv:1802.07465* [cs.CV], Feb 2018.

[5] H. Chen, X. Qi, L. Yu, P. A. Heng, "DCAN: Deep Contour-Aware Networks for Accurate Gland Segmentation". *arXiv preprint arXiv: 1604.02677* [cs.CV], Apr 2016.

[6] Y. Cui, G. Zhang, Z. Liu, Z. Xiong, J. Hu, "A Deep Learning Algorithm for One-step Contour Aware Nuclei Segmentation of Histopathological Images". *arXiv preprint arXiv: 1803.02786* [cs.CV], Mar 2018.

[7] M. Bai, R. Urtasun, "Deep Watershed Transform for Instance Segmentation". *arXiv preprint arXiv:1611.08303* [cs.CV], May 2017.

[8] F. Chollet, "Xception: Deep Learning with Depthwise Separable Convolutions". *arXiv preprint arXiv: 1610.02357* [cs.CV], Aprl 2017.

[9] K. He, X. Zhang, S. Ren, J. Sun, "Deep Residual Learning for Image Recognition". *arXiv preprint arXiv: 1512.03385* [cs.CV], Dec 2015.

[10] X. Glorot, Y. Bengio, "Understanding the difficulty of training deep feedforward neural networks," in Pro-ceedings of the Thirteenth International Conference on Artifi-cial Intelligence and Statistics, 2010, pp. 249–256.

[11] V. Iglovikov, A. Shvets, "TernausNet: U-Net with VGG11 Encoder Pre-Trained on ImageNet for Image Segmentation". *arXiv preprint arXiv: 1801.05746* [cs.CV], Jan 2018.

[12] C. Szegedy, V. Vanhoucke, S. Loffe, J. Shlens, Z. Wojna, "Rethinking the Inception Architecture for Computer Vision". *arXiv preprint arXiv: 1512.00567* [cs.CV], Dec 2015.

[13] J. Hu, L. Shen, G. Sun, "Squeeze-and-Excitation Networks". *arXiv preprint arXiv: 1709.01507* [cs.CV], Apr 2018.

[14] S.Loffe, C. Szegedy, "Batch Normalization: Accelerating Deep Network Training by Reducing Internal Covariate Shift". *arXiv preprint arXiv: 1502.03167* [cs.CV], Mar 2015.

[15] D. A. Clevert, T. Unterthiner, S. Hochreiter, "Fast and Accurate Deep Network Learning by Exponential Linear Units (ELUs)". *arXiv preprint arXiv: 1511.07289* [cs.CV], Feb 2016.

[16] G. Klambauer, T. Unterthiner, A. Mayr, S. Hochreiter, "Self-Normalizing Neural Networks". *arXiv preprint arXiv: 1706.02515* [cs.CV], Sep 2017.

[17] D. P. Kingma, J. Ba , "Adam: A Method for Stochastic Optimization". *arXiv preprint arXiv: 1412.6980* [cs.CV], Jan 2017.