WWW.IJRPR.COM

International Journal of Research Publication and Reviews

Journal homepage: www.ijrpr.com ISSN 2582-7421

SEMANTIC IMAGE SEGMENTATION OF KIDNEY HISTOLOGY IMAGES USING UNET ARCHITECTURE AND SLIDING WINDOW LIKE ALGORITHM

Atharva Hude[a], Aditya Pawase[b], Abhishek Jadhav[c], Atharva Wadkar[d]

[a,b,c,d]Bachelor of Engineering in Computer Engineering, Pune 411045, India

A B S T R A C T

The research in biomedical image segmentation blossomed after the famous unet architecture which was published in 2015. In 2020-21 the Human BioMolecular Atlas Program (HuBMAP), a major endeavour, sponsored by the National Institutes of Health (NIH) published an open-source dataset on Kaggle for semantic image segmentation of glomeruli in histology images. The proposed method uses deep learning models and a sliding window-like algorithm that classifies individual pixels between two classes: Glomerulus and Non-Glomerulus, thus generating an output binary mask that is of the same dimensions as the input image. The output generated when evaluated, with the test label masks gives us a fairly good result. There is some noise in the output which can be tacked by experimentation in the future.

Keywords: CNN, Deep Learning, Neural Networks, Image Segmentation, Glomerulus Classification, Unet

1. Introduction

Our challenge is to detect Glomeruli (Functional Tissue Unit) in Histology images automatically with the help of a deep learning model. This paper proposes a method using the Unet model and a sliding window-like algorithm to generate binary masks of the input image.
Semantic segmentation in Deep learning is an advanced computer vision task. The output of an image segmentation model such as Unet is a mask image where each pixel in the mask is associated with a class. In our case, we have two classes i.e., FTU class and Background class. The challenge with the dataset is such that the input image RGB has large dimensions. Today's GPUs have limited memory, training a huge deep learning model is next to impossible, so we use a sliding window like algorithm to generate 256*256 masks and then stitch these slices to generate a mask image with the same dimensions as that of the input image.

2. Objective

Using Deep learning model (UNet) architecture to develop a method to automatically identify Glomeruli in histology images. The goal of this project is to implement a successful and robust glomeruli FTU detector.

E-mail address: *atharvahude005@gmail.com*, *adityapawase@gmail.com, abhishekbuzz4@gmail.com, atharvawadkar@gmail.com*

International Journal of Research Publication and Reviews Vol. 2, Issue (7) Page 2050-2055 , July 2021

3. Dataset Analysis

The dataset sourced from the Kaggle competition named Human BioMolecular Atlas Program (HuBMAP), which is organized by the National Institutes of Health (NIH). The dataset consists of very large (>500MB-5GB) TIFF files with corresponding csv files which contain the mask in RLE encoded format representing segmentations of glomeruli. The output mask image can be generated by decoding this RLE encoded strings. Below table describes the nature of the images.

	Mean	Standard Deviation
Height Pixels	30300	7820
Width Pixels	36800	10700

As we can infer from the table that the input dimensions of the images are large by today's standards. We cannot process this image in its entirety. One solution is to divide the input image and corresponding mask image into tiles of dimension 256*256*3 and 256*256 respectively thus creating a custom dataset for training.

4. Network Architecture

The Unet-like model is implemented using the Keras functional API in TensorFlow. The model takes an input image of size 256*256*3 and outputs a mask of dimension 256*256*1. Here is the model summary:

Layer (type)	Output Shape	Param #	Connected to
input_1 (InputLayer)	[(None, 256, 256, 3)	0	
conv2d (Conv2D)	(None, 256, 256, 16)	448	input_1[0][0]
dropout (Dropout)	(None, 256, 256, 16)	0	conv2d[0][0]
conv2d_1 (Conv2D)	(None, 256, 256, 16)	2320	dropout[0][0]
max_pooling2d (MaxPooling2D)	(None, 128, 128, 16)	0	conv2d_1[0][0]
conv2d_2 (Conv2D)	(None, 128, 128, 32)	4640	max_pooling2d[0][0]
dropout_1 (Dropout)	(None, 128, 128, 32)	0	conv2d_2[0][0]
conv2d_3 (Conv2D)	(None, 128, 128, 32)	9248	dropout_1[0][0]
max_pooling2d_1 (MaxPooling2D)	(None, 64, 64, 32)	0	conv2d_3[0][0]
conv2d_4 (Conv2D)	(None, 64, 64, 64)	18496	max_pooling2d_1[0][0]
dropout_2 (Dropout)	(None, 64, 64, 64)	0	conv2d_4[0][0]
conv2d_5 (Conv2D)	(None, 64, 64, 64)	36928	dropout_2[0][0]
max_pooling2d_2 (MaxPooling2D)	(None, 32, 32, 64)	0	conv2d_5[0][0]
conv2d_6 (Conv2D)	(None, 32, 32, 128)	73856	max_pooling2d_2[0][0]
dropout_3 (Dropout)	(None, 32, 32, 128)	0	conv2d_6[0][0]

conv2d_7 (Conv2D)	(None, 32, 32, 128)	147584	dropout_3[0][0]
max_pooling2d_3 (MaxPooling2D)	(None, 16, 16, 128)	0	conv2d_7[0][0]
conv2d_8 (Conv2D)	(None, 16, 16, 256)	295168	max_pooling2d_3[0][0]
dropout_4 (Dropout)	(None, 16, 16, 256)	0	conv2d_8[0][0]
conv2d_9 (Conv2D)	(None, 16, 16, 256)	590080	dropout_4[0][0]
conv2d_transpose (Conv2DTranspo	(None, 32, 32, 128)	131200	conv2d_9[0][0]
concatenate (Concatenate)	(None, 32, 32, 256)	0	conv2d_transpose[0][0] conv2d_7[0][0]
conv2d_10 (Conv2D)	(None, 32, 32, 128)	295040	concatenate[0][0]
dropout_5 (Dropout)	(None, 32, 32, 128)	0	conv2d_10[0][0]
conv2d_11 (Conv2D)	(None, 32, 32, 128)	147584	dropout_5[0][0]
conv2d_transpose_1 (Conv2DTrans	(None, 64, 64, 64)	32832	conv2d_11[0][0]
concatenate_1 (Concatenate)	(None, 64, 64, 128)	0	conv2d_transpose_1[0][0 conv2d_5[0][0]
conv2d_12 (Conv2D)	(None, 64, 64, 64)	73792	concatenate_1[0][0]
dropout_6 (Dropout)	(None, 64, 64, 64)	0	conv2d_12[0][0]
conv2d_13 (Conv2D)	(None, 64, 64, 64)	36928	dropout_6[0][0]
conv2d_transpose_2 (Conv2DTrans	(None, 128, 128, 32)	8224	conv2d_13[0][0]
concatenate_2 (Concatenate)	(None, 128, 128, 64)	0	conv2d_transpose_2[0][0 conv2d_3[0][0]
conv2d_14 (Conv2D)	(None, 128, 128, 32)	18464	concatenate_2[0][0]
dropout_7 (Dropout)	(None, 128, 128, 32)	0	conv2d_14[0][0]
conv2d_15 (Conv2D)	(None, 128, 128, 32)	9248	dropout_7[0][0]
conv2d_transpose_3 (Conv2DTrans	(None, 256, 256, 16)	2064	conv2d_15[0][0]
concatenate_3 (Concatenate)	(None, 256, 256, 32)	0	conv2d_transpose_3[0][0 conv2d_1[0][0]
conv2d_16 (Conv2D)	(None, 256, 256, 16)	4624	concatenate_3[0][0]
dropout_8 (Dropout)	(None, 256, 256, 16)	0	conv2d_16[0][0]
conv2d_17 (Conv2D)	(None, 256, 256, 16)	2320	dropout_8[0][0]
conv2d_18 (Conv2D)	(None, 256, 256, 1)	17	conv2d_17[0][0]

Total parameters: 1,941,105
Trainable parameters: 1,941,105
Non-trainable parameters: 0

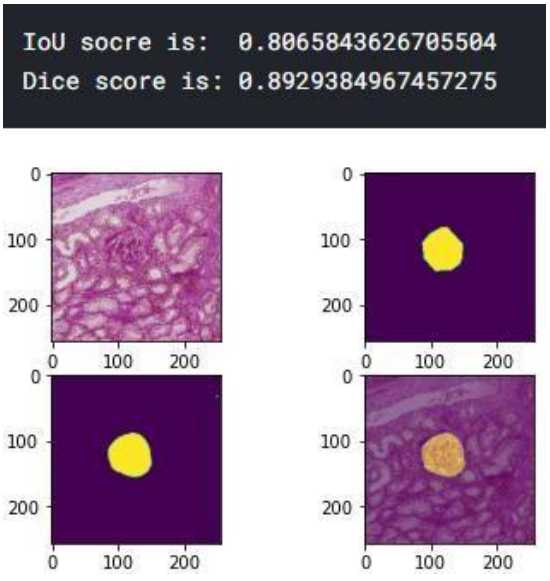
International Journal of Research Publication and Reviews Vol. 2, Issue (7) Page 2050-2055 , July 2021

5. Training & Inferencing

The experiment was conducted in two phases. In the first phase, we trained the custom Unet model on 256*256*3 tiled images and in the second phase, we used a sliding window type algorithm on the original image and generated the output mask.

Phase 1: The training dataset consisted of 3676 jpeg images and their corresponding masks. The input data pipeline is created using TensorFlow's TensorFlow data services (TFDS) and split into training, validation and testing sets. Normalization, Shuffling and generating batches for training is done on the fly by TensorFlow. The model is compiled using the Adam optimizer with default parameters and binary cross-entropy.

Training results after 50 epochs: - loss: 0.0099 - accuracy: 0.9960 - val_loss: 0.0432 - val_accuracy: 0.9918



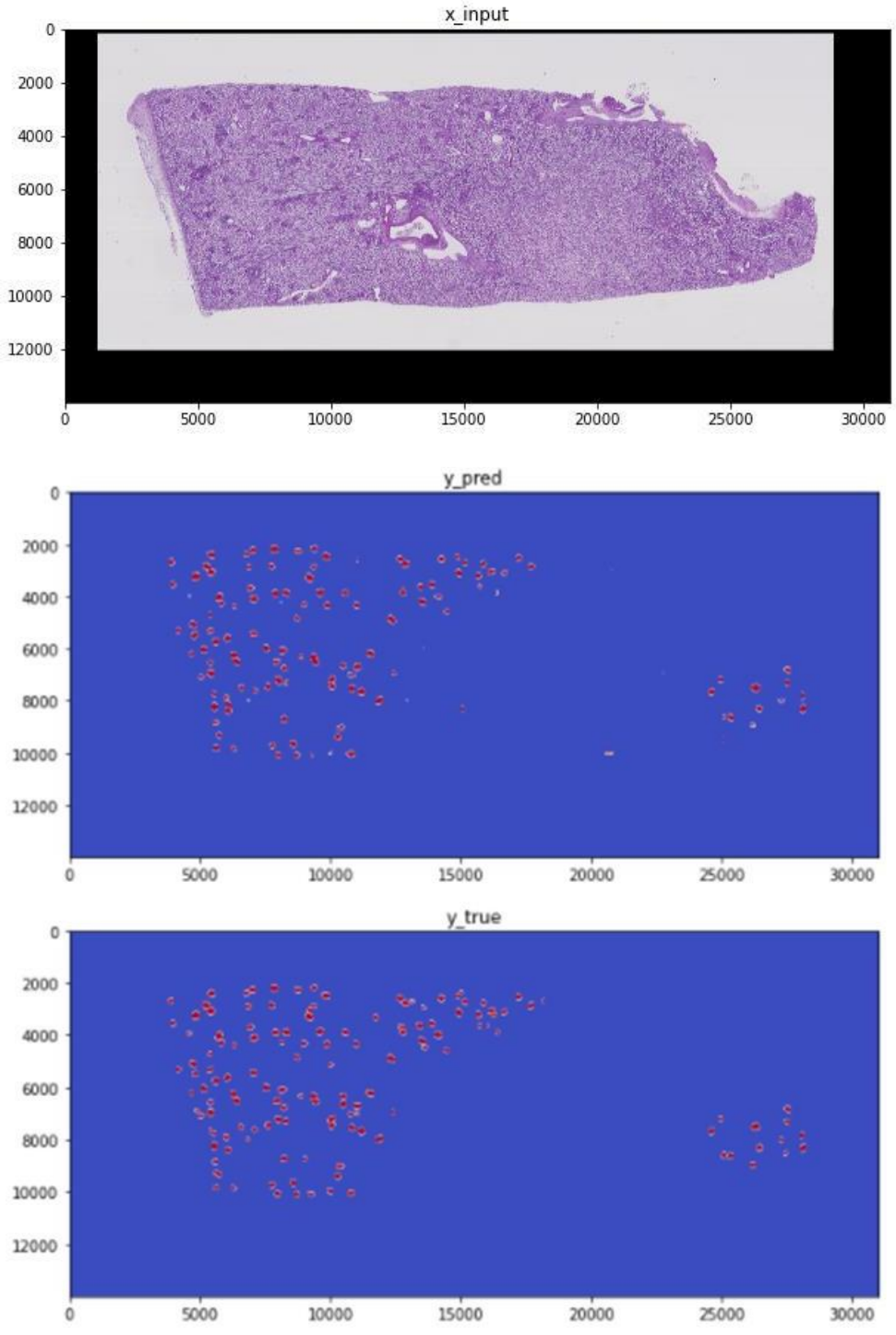
https://www.kaggle.com/competitions/hubmap-kidney-segmentation/data

(Fig Description: The top right corner is the input image (X) Dimension = 256*256*3. The top left image is the corresponding mask (Y_true) Dimension = 256*256. Bottom left is the output predicted mask (Y_Pred) DImension = 256*256. The bottom right is a superimposed image of Y_pred and X.) Segmentation models are evaluated using two metrics Iou and Dice Score.

Phase 2: In the second phase of the experiment, we generated the final required output mask. As we are sliding a window over the input image, we need to resize the image to the nearest thousand. The dimensions of the image are very high; resizing does not affect the quality. Then using NumPy's array slicing we slide across the image and take each slice of 1000*1000*3 and resize it to 256*256*3 to feed it to the custom unet model to generate the mask which is of dimension 256*256*1. Again, we upscale the output mask to 1000*1000*1 and repeat this loop until the whole image is done. Below are the results from a test image from the dataset.

The input image(x_input) is a TIFF file of dimensions 14844 * 31262*3 before slicing we need to resize the dimensions to the nearest thousand so that we could slice it in a window size of 1000*1000*3. Hence the dimension of the input image after resizing is 14000*31000*3. After running the sliding window-like algorithm on this input image(x_input) we got an output mask(y_pred) that had the same dimensions. Now to evaluate the output mask, we need to compare it with the annotated mask(y_true). After computing the scores, we obtained the IOU of 0.7551247129948839 and the Dice Score of 0.8604798364522639.

After looking at the generated masks we can observe that some pixels are falsely classified. There is also some noise in the output. These drawbacks can be tackled by using a larger model and hyperparameter tuning methods.



https://www.kaggle.com/competitions/hubmap-kidney-segmentation/data

6. Conclusion

Before Deep Learning, many things that could have taken considerable time and expert supervision have become simple in medical fields, such as classifying and detecting glomeruli, which was previously required only by experts, now it is possible with Deep Learning. With the U-net model, we have successfully implemented the FTU detector on a very large image and produced the mask image as an output.

ACKNOWLEDGEMENT

This study was conducted with the support of Kaggle's deep learning Community and the authors of the HUBMAP dataset.

REFERENCES

[1] Bukowy, John D., Alex Dayton, Dustin Cloutier, Anna D. Manis, Alexander Staruschenko, Julian H. Lombard, Leah C. Solberg Woods, Daniel A. Beard, and Allen W. Cowley. 2018. "Region-Based Convolutional Neural Nets for Localization of Glomeruli in Trichrome-Stained Whole Kidney Sections." Journal of the American Society of Nephrology 29 (8): 2081– 88.
https://doi.org/10.1681/ASN.2017111210

[2] Govind, Darshana, Brandon Ginley, Brendon Lutnick, John E. Tomaszewski, and Pinaki Sarder. 2018. "Glomerular Detection and Segmentation from Multimodal Microscopy Images Using a Butterworth Band-Pass Filter." In Medical Imaging 2018: Digital Pathology, 10581:1058114. International Society for Optics and Photonics. https://doi.org/10.1117/12.2295446

[3] U-Net: Convolutional Networks for Biomedical Image Segmentation.
https://arxiv.org/pdf/1505.04597.pdf

[4] Glomerulus Classification and Detection Based on Convolutional Neural Networks (Gallego et al. 2018).
https://doi.org/10.3390/jimaging4010020

[5] Kannan, Shruti, Laura A. Morgan, Benjamin Liang, McKenzie G. Cheung, Christopher Q. Lin, Dan Mun, Ralph G. Nader, et al. 2019. "Segmentation of Glomeruli Within Trichrome Images Using Deep Learning." Kidney International Reports 4 (7): 955–62.
https://doi.org/10.1016/j.ekir.2019.04.008