Neuronal Cell Segmentation using MASK-RCNN Architecture

1st Tanvir Anjum
Undergraduate Student
Department of EEE
Bangladesh University of Engineering and Technology
Dhaka, Bangladesh

2nd Arafat Asim
Undergraduate Student
Department of EEE
Bangladesh University of Engineering and Technology
Dhaka, Bangladesh

3rd Mehedi Hasan
Undergraduate Student
Department of EEE
Bangladesh University of Engineering and Technology
Dhaka, Bangladesh

4th Walid Hasan
Undergraduate Student
Department of EEE
Bangladesh University of Engineering and Technology
Dhaka, Bangladesh

5th Hasin Ishraq
Undergraduate Student
Department of EEE
Bangladesh University of Engineering and Technology
Dhaka, Bangladesh

***Abstract*—The quantitative study of cell morphology is of great importance as the structure and condition of cells and their structures can be related to conditions of health or disease. In neuro science, instance segmentation plays a very important role to determine neural diseases. Segmentation of neural cell is challenging due to the special shapes of neuron cell. In this work we have used a Deep Neural Network with Mask-RCNN architecture and trained the model with preprocessed images to achieve better performance and accuracy.**

***Index Terms*—segmentation, MASK-RCNN, adaptive histogram, laplacian, gamma correction**

I. Introduction

Cellular mechanism contained the hereditary path from a single neural stem cell remain a mystery in neuroscience. With the help of real time microscopy imaging system the neurons specification from a single stem cell could be recorded . As an important tool to explore the interactions between the cells, neural cell instance segmentation algorithm is in great desire since it locates and segments the cells at the same time. In a definite way a fast and accurate instance segmentation tool is very important when we survey large dataset. However neural instance segmentation is a very difficult due to various factors such as cell mitosis, cell distortion and so on. Besides Medical image segmentation has automatic or semiautomatic detection of the two-dimensional (2D) or three-dimensional (3D) image. Image segmentation is the procedure of dividing a digital image into a multiple set of pixels. In recent years object detection has been significantly advanced following the big success by deep learning. Also we have witnessed a significant improvement in segmentation due to deep neural network(DNN).With the help of these improvements, we propose a deep learning model for neural instance segmentation which takes full advantages of global context information both of detection and segmentation.

II. Proposed Methods

The images in the dataset were first processed using various preprocessing methods. The processed images were then passed to core network which generated the binary mask of the segmentation and detection results.

A. Dataset

For our report we have use two types of dataset - LIVEcell dataset [1] and Sartorius Cell Segmentation Competition dataset [2]

a) LIVEcell dataset

The LIVECell dataset is a large-scale microscopic image dataset for instance-segmentation of individual cells in 2D cell cultures. LIVECell consists of 5,239 manually annotated, expert-validated, Incucyte HD phase-contrast microscopy images with a total of 1,686,352 individual cells annotated from eight different cell types (average 313 cells per image). The LIVECell images have predefined splits into training (3188), validation (539) and test (1512) sets. Each split is also further subdivided into each of the eight cell types.

b) Sartorius dataset

Sartorius - Cell instance segmentation competetion held in kaggle presented a small dataset of only 606 annotated images.The training annotations were provided as run length encoded. Although the number of images is small, but the number of annotated objects is quite high. The images are in PNG format.

B. Pre-processing and Augmentations

Three pre-processing techniques were used on top of our deep learning model to improve the result - Gamma Correction, Adaptive Histogram and Laplacian Filter

a) Gamma Correction

Gamma correction controls the overall brightness of an image. Gamma correction is a nonlinear adaptation applied to each and every pixel value [3]. Generally linear methods like addition, subtraction, and multiplication are applied on all the pixels. Gamma correction is responsible for performing nonlinear methods on the pixels of the input image and thereby remodeling the saturation of the image. Gamma correction is also known as the Power Law Transform [4]. we obtain our output gamma corrected image by applying the following equation:

$$O = I^{-}\gamma \tag{1}$$

Where I is our input image and γ is our gamma value. Gamma values < 1 will shift the image towards the darker end of the spectrum while gamma values > 1 will make the image appear lighter. A gamma value of $\gamma = 1$ will have no affect on the input image.

b) Adaptive Histogram

Histogram equalization is a basic image processing technique that adjusts the global contrast of an image by updating the image histogram's pixel intensity distribution. Doing so enables areas of low contrast to obtain higher contrast in the output image. [5] By applying histogram equalization the input image's contrast improves significantly but at the expense of also boosting the contrast of the noise In the input image. To overcome this problem we used adaptive histogram equalization. With adaptive histogram equalization, we divide an input image into an M x N grid. We then apply equalization to each cell in the grid, resulting in a higher quality output image

c) Laplacian Filter

A Laplacian filter is an edge detector used to compute the second derivatives of an image, measuring the rate at which the first derivatives change [6]. This determines if a change in adjacent pixel values is from an edge or continuous progression. It is a derivative filter used to extract the vertical as well as horizontal edges from an image.

C. Implemented Core Network

The implemented network consists of three parts as shown in 'Fig. 1' : Backbone, Neck and Head. The backbone of the model creates the necessary features for segmentation.The Neck collects features of various scales from the Backbone and process them up to a singular level. The Head finally creates the predictions from the processed features from the Neck.

a) Backbone

As the Backbone (Fig.2), we used ResNet-50 [7]: a 50 layer convolutional network shown in 'Fig.2'. It consists of multiple bottleneck blocks and a stem block. The stem block uses a kernel size of 7 and compresses the image in spatial dimension and extracts the features to channel dimension. The bottleneck block consists of a primary network and a skip connection. If the bottleneck layer is used for down sampling, an extra 1x1 convolution layer is added to the skip connection to match the feature of the other branch in channel size.

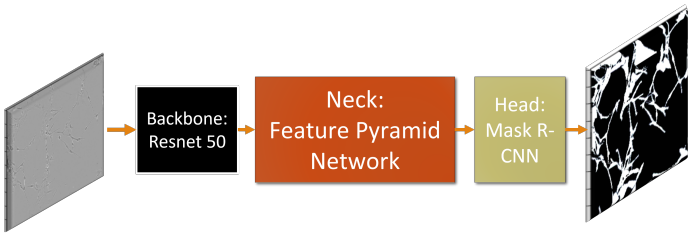

Fig. 1. The Implemented Network

b) Neck

We used Feature Pyramid Network (FPN) [8] as the neck of the network shown in "Fig.3". The features we extracted from the backbone are of different scales. Therefore, they have varying receptive fields in the sense that each of these features went through different degrees of convolution operation and therefore the features they possess occupy different parts of the image. The feature extracted at the deeper end possess more defined features in small scale in comparison to the features extracted from the shallow end which possess more generalized information about the image section. The FPN network tries to combine these features and create features of different scales to detect different sizes of objects from the image.

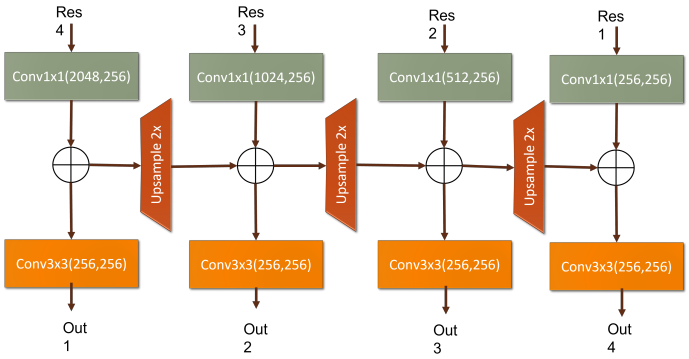

Fig. 3. Neck: Field Pyramid Network

c) Head

The Mask-RCNN architecture [9] was used in our model to finally generate the segmentation mask. The process in shown in "Fig.5" For each feature extracted from the neck, 3 anchors were generated per feature. From the 4 features extracted from the neck, each feature of shape (H,W) generate H*W numbers of boundary box predictions with their objectiveness score using the RPN (Regional Proposal Network) Head. These Region of Interest (ROI) predictions were then filtered in three stages and further processed to keep only the boxes with least overlapping and highest objectiveness scores in the Box Prediction block. Theses processed predictions are then passed to the Mask head where these boundary boxes were used to generate the segmentation mask.

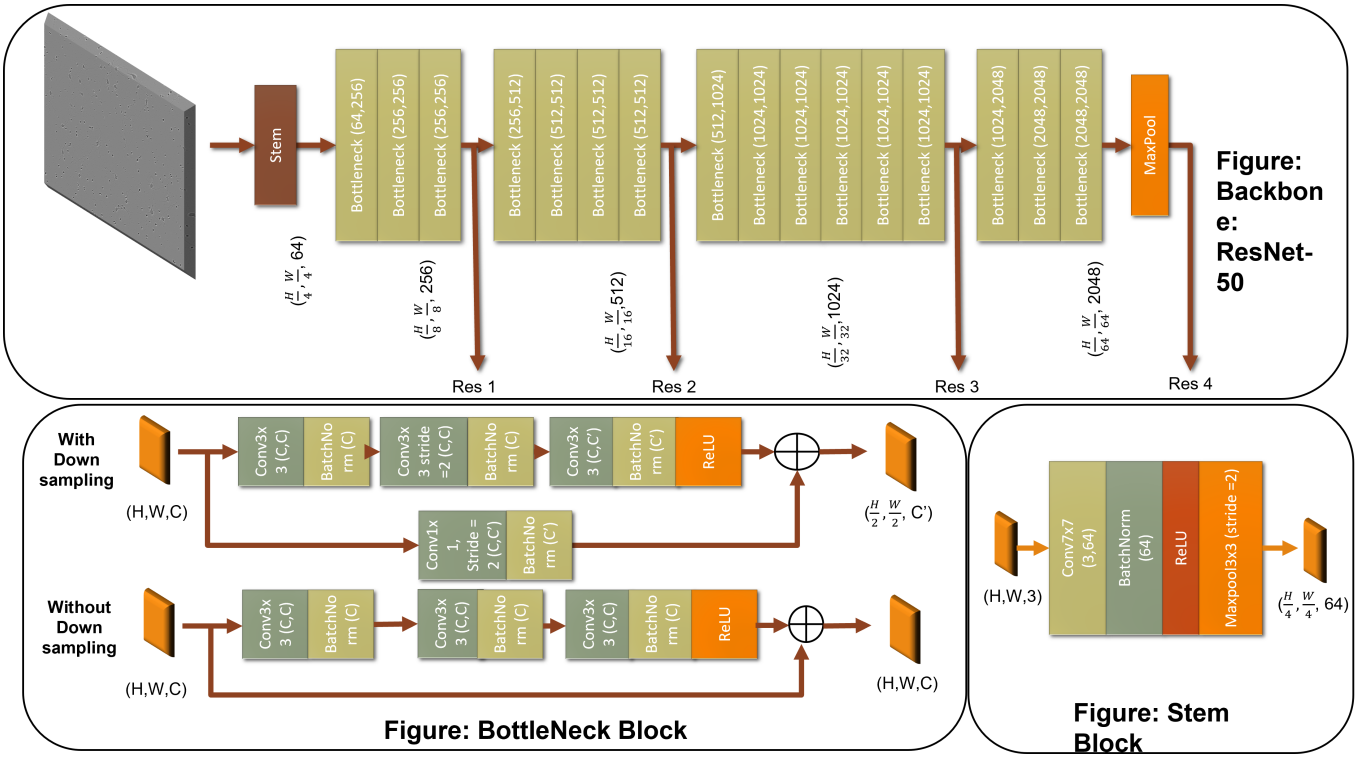

Fig. 2. Backbone of the Implemented Network

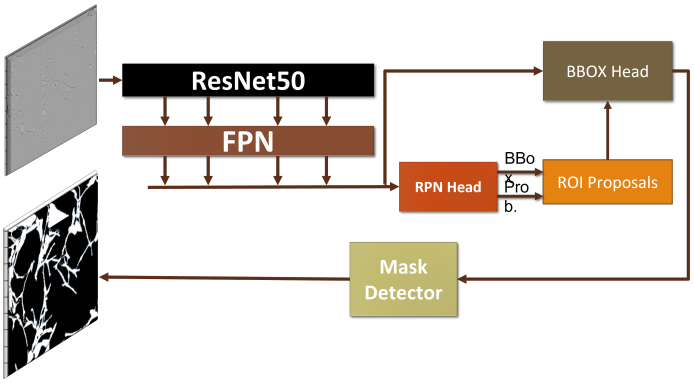

Fig. 4. Mask-RCNN Head

III. Metrics and Results

A. Metrics

As we have implemented segmentation and detection model for our Sartorius dataset, we used mask R-CNN model. We could use deferent matrices Such as: Accuracy, Precision, Recall, F1-score, map, Average Precision, Fallout [**?**]. As our competition demand was IOU, we used mean Average Precision (mAP) at different intersection over union (IoU).

a) Precision

Precision measures how accurate your predictions are [10]. It measures how many of the predictions that your model made were correct.

$$Precision = \frac{True\ Positives}{True\ Positives + False\ Positives} \tag{2}$$

True Positives are cases when predictions are positive and correct False Positives are cases when predictions are positive but incorrect

b) Recall

$$Recall = \frac{True\ Positives}{True\ Positives + False\ Negatives} \tag{3}$$

The recall cares only about how the positive samples are classified. This is independent of how the negative samples are classified, e.g., for the precision. When the model classifies all the positive samples as Positive, then the recall will be 100% even if all the negative samples were incorrectly classified as Positive.

c) Mean Average Precision

mAP is not calculated by taking the average of precision values [11]. Object detection systems make predictions in terms of a bounding box and a class label. For each bounding box, we measured an overlap between the predicted bounding box and the ground truth bounding box. This is measured by IoU (intersection over union).

$$IOU = \frac{Intersection\ Area\ of\ the\ bboxes}{Total\ Area\ of\ bboxes} \tag{4}$$

we calculated Precision using IoU value for a given IoU threshold.

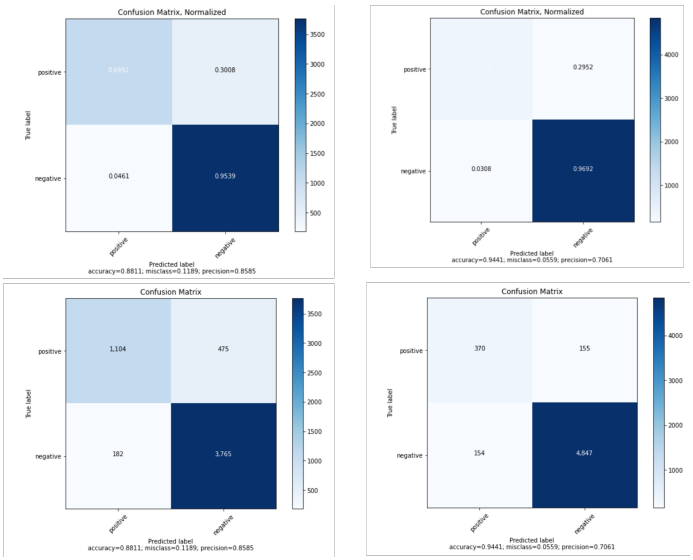

Fig. 5. Confusion Matrices [12]

B. Results

We have tried different type of models for segmentation. We have used Sartorius dataset for training and testing our model. We have splited our dataset in 0.3 ratio where training dataset was 70% and validation dataset was 30% of whole dataset. We ran our model for sufficient epochs, and we avoided overfitting so that our model could not biased. Firstly, we have used Unet++ model which test accuracy, precision, recall was accordingly 0.9024, 0.5050, 0.4998. After some trial-and-error method we found Unet+Attention (EfficientNet b7-5) Was better performer than others and gave about 20% better result than previous models,

Model		Acc	Prec	Rec	F1
Deep Labv3	EfficientNet-b7	0.8946	0.4752	0.6289	0.5414
	EfficientNet-b5	0.8984	0.4988	0.5535	0.5247
	ResNet 50	0.8886	0.5009	0.5398	0.5196
Deep Labv3+	EfficientNet-b7	0.9314	0.6850	0.6119	0.6464
	EfficientNet-b5	0.9223	0.6742	0.6305	0.6516
	ResNet 50	0.9125	0.5654	0.7438	0.6424
LinkNet	EfficientNet-b7	0.9344	0.6667	0.7440	0.7032
	EfficientNet-b5	0.9308	0.6530	0.7490	0.6977
	ResNet 50	0.9225	0.6535	0.7091	0.6802
Swin-V2	Unet	0.9157	0.5827	0.6102	0.5961
Swin-V2	FPN	0.8879	0.4404	0.2665	0.3321
Unet++		0.9024	0.5050	0.4998	0.5024
Unet+ Attention	Efficientnetb7-5	0.9434	0.7040	0.7072	0.7592
	Efficientnetb7-4	0.9397	0.6798	0.7411	0.7091
	Efficientnetb7-3	0.9315	0.6745	0.7306	0.7015
	Efficientnetb5-5	0.9362	0.6578	0.7632	0.7066
	Efficientnetb5-4	0.9308	0.6405	0.7945	0.7092
	Efficientnetb5-3	0.9365	0.6476	0.7582	0.6985
	Efficientnetb3-5	0.9402	0.6695	0.7200	0.6939
	Efficientnetb3-4	0.9360	0.6622	0.7696	0.7119
	Efficientnetb3-3	0.9393	0.6587	0.7406	0.6972

TABLE I
COMPARISON OF MATRICES OF TRIED MODEL IN TABULAR FORMAT

Table.I shows the comparison between all the models we have tried. But when we submit our best model that gives us very poor Kaggle submission score, it is about to 0.02. Then we have tried another model called mask R-CNN with data augmentation. Table II shows the comparison in different matrices:

Model	Accuracy	Precision	Recall	F1-score
Unet+Attention	0.9434	0.7040	0.7072	0.7592
Mask R-CNN	0.8811	0.8585	0.4341	0.7707

TABLE II
COMPARISON BETWEEN UNET+ATTENTION AND MASK-RCNN

We submitted our model in Kaggle competition and it gave public score 0.294 and private score 0.287 shown in Table.III

Comments	Leaderboard score
Different models	0.02
Mask R-CNN 12 epochs and image shape (256*256)	0.057
8 epochs with scheduler	0.197
10 epochs using Adam optimizer	0.135
3 epochs size 25%. 3 epochs size 50%. 6 epochs full sized	0.178
30 epochs	0.203
Using with MIN_SCORE=0.5, use best validation epoch (19)	0.263
Use cell type as class labels, use best validation epoch using IOU score	0.274
Use different thresholds for each class	0.279
Box detection per image =540	0.281
Adding preprocessing steps	0.293

TABLE III
PERFORMANCE IN KAGGLE COMPETITION METRIC

IV. DISCUSSION

In this paper, we have tried to modify and use various deep learning models to achieve a better performance in segmentation of neural cells. Although our model does not have any significant improvement in cell instance segmentation over existing methods, it has a great potential to achieve a better result by modifying it by implementing attention and transformation in the network and using different methods of pre-processing of the images.

ACKNOWLEDGMENT

We are thoroughly indebted to our instructors, Mr. Shahed Ahmed and Mr. Talha Ibn Mahmud for their unlimited guidance and support. Without their help, this project would have been impossible.

REFERENCES

[1] C. Edlund, T. R. Jackson, N. Khalid, N. Bevan, T. Dale, A. Dengel, S. Ahmed, J. Trygg, and R. Sjögren, "Livecell—a large-scale dataset for label-free live cell segmentation," *Nature Methods*, vol. 18, no. 9, p. 1038–1045, 2021.

[2] "Sartorius." [Online]. Available: https://www.kaggle.com/c/sartorius-cell-instance-segmentation/overview

[3] "Gamma correction." [Online]. Available: https://www.sciencedirect.com/topics/engineering/gamma-correction

[4] Abhinav, A. Rosebrock, Brian, Rossi, YuriiChernyshov, Y. Igo, I. Among, Ambika, Kinvert, Esha, and et al., "Opencv gamma correction," Apr 2021. [Online]. Available: https://pyimagesearch.com/2015/10/05/opencv-gamma-correction/

[5] "Opencv histogram equalization and adaptive histogram equalization (clahe)," Apr 2021. [Online]. Available: https://pyimagesearch.com/2021/02/01/opencv-histogram-equalization-and-adaptive-histogram-equalization-clahe/
[6] "Laplacian filter." [Online]. Available: https://www.sciencedirect.com/topics/engineering/laplacian-filter
[7] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," *CoRR*, vol. abs/1512.03385, 2015. [Online]. Available: http://arxiv.org/abs/1512.03385
[8] F. Wu, A. J. Ma, Y. Pan, Y. Gao, and X. Yan, "Receptive field pyramid network for object detection," in *ICASSP 2020 - 2020 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2020, pp. 1873–1877.
[9] K. He, G. Gkioxari, P. Dollár, and R. B. Girshick, "Mask R-CNN," *CoRR*, vol. abs/1703.06870, 2017. [Online]. Available: http://arxiv.org/abs/1703.06870
[10] "Plugins." [Online]. Available: https://supervise.ly/explore/plugins/precision-and-recall-75278/overview
[11] "How to calculate the mean average precision (map) in object detection- an overview," Feb 2022. [Online]. Available: https://hungsblog.de/en/technology/how-to-calculate-mean-average-precision-map/
[12] "Confusion matrix," Feb 2022. [Online]. Available: https://hasty.ai/content-hub/mp-wiki/metrics/confusion-matrix