
VIT®

Vellore Institute of Technology

(Deemed to be University under section 3 of UGC Act, 1956)

# Cell Image Segmentation using Deep Learning

Submitted by

Name: Swarup Tripathy

Registration No.: 19BEE0167

Under the guidance of

Professor Mohan C.G

# Abstract

As an emerging biomedical image processing technology, medical image segmentation has made great contributions to sustainable medical care. In this paper, the motive is to present the implementation of UNet++ architecture, a very powerful architecture for medical image segmentation and extraction. This robust cellular segmentation model is essentially a deeply supervised encoder-decoder network where the encoder and decoder sub-networks are connected through a series of nested, dense skip pathways. The current basic image processing algorithms have been developed to solve the general computer vision/segmentation problems and all of them have inherent insufficiencies. The facing challenges include the ever-increasing complexity, the great variety of cell types, the low image contrast, the poor image quality, the connection or overlapping of neighbouring cells, the nonuniform pixel intensity and the influence of noise or clutter.


# 1 Introduction

The first uses of computers for the analysis of cells date back more than half a century. Already in the mid-1950s, systems were developed to automate the classification of smears of exfoliated cells, with the ultimate aim to enable mass screening for cervical cancer. These systems applied thresholding-based decision rules to serial one-dimensional (1-D) microscopic line scans of a specimen. Digital pathology and microscopy image analysis is widely used for comprehensive studies of cell morphology or tissue structure. Manual assessment is labour intensive and prone to inter-observer variations. Computer-aided methods, which can significantly improve the objectivity and reproducibility, have attracted a great deal of interest in recent literatures. In the recent times, optical microscopy is widely used to quantify single cell features, such as cell size or intracellular densities of fluorescent markers. Although there is a rapid development of imaging hardware and image analysis software platforms, the development of cell segmentation algorithms is lagging behind.

Cell Segmentation is a technique or task of splitting a microscopic image domain into segments, which represent individual instances of cells. This is considered to be a fundamental step in many biomedical studies and is regarded as a stepping stone of image-based cellular research.

Cell Segmentation is challenging for many reasons. First, one of the most non-trivial tasks in image processing is segmentation which requires the identification of multiple objects in the image. Cell nuclei's have heterogeneous shapes that are

typically subject to dynamic changes. Second, growing cell populations usually result in dense cell regions; this makes it hard to assign image features to the correct cell, especially among sets of spatially close cells.

Deep learning is one alternative technique to handcrafted feature-based approaches because it can learn visual features automatically to solve a specific task, such as object classification and target segmentation. In many research fields, a convolutional neural network (CNN)-based model, which is a deep learning technique, has been reported to be extremely effective compared with handcraft-based approaches.

Arguably, image segmentation in natural images has reached a satisfactory level of performance, but do these models meet the strict segmentation requirements of medical images?

Segmenting lesions or abnormalities in medical images demand a higher level of accuracy that what is desired in natural images. While a precise segmentation mask may not be critical in natural images, even marginal segmentation errors in medical images can lead to poor user experience in clinical settings.

# 2 Related Work

One of the most common and important features of extraction is the identification of membrane pattern with the microscopy images. Based on the input image, we specify few biologically intuitive parameters. Then, we detect the potential features especially margins of the cells. All individual segmentation results are processed to obtain the final segmentation of the image.

## 2.1 Intensity Thresholding

One of the most predominant approaches towards cell segmentation is intensity thresholding where the underlying assumption is that cells have significantly and consistently different intensities that background, either globally, in which case a fixed, or locally, which would require adaptive thresholding. Most cell segmentation methods apply thresholding only as a first step in the pipeline.

## 2.2 Region Accumulation

An alternative approach to cell segmentation is to start from selected seed points in the image and to iteratively add connected points to form labelled regions. The most straightforward implementation of this idea is ordinary region growing, which works per neighbourhood layer of connected points and, when applied directly to the image, assumes (and suffers from) a similar image model as in the

case of intensity thresholding. Though by far the most popular region accumulation approach, the watershed transform is infamous for producing oversegmentation and usually requires further processing.

### 2.3 Cell Segmentation

Automatic cell detection and counting in microscopy images has been earlier studied with the support of image processing and computer vision techniques. In particular, fully convolutional neural networks (FCNs) allow to make predictions on the same or similar spatial resolution of the input image. FCN approaches have been widely adapted in medical imaging with applications to nuclei segmentation, brain tumor segmentation from magnetic resonance imaging (MRI) and of course segmentation in microscopy images. Among the Fully Convolution Neural Network UNet has been a very popular architecture. In FCN, up-sampled feature maps are summed with feature maps skipped from the encoder, while U-Net concatenates them and add convolutions and non-linearities between each up-sampling step. The skip connections have shown to help recover the full spatial resolution at the network output, making fully convolutional methods suitable for semantic segmentation.

The other two recent related works are GridNet and Mask-RCNN. GridNet is an encoder-decoder architecture wherein the feature maps are wired in a grid fashion, generalizing several classical segmentation architectures. GridNet, however, lacks up-sampling layers between skip connections; and thus, it does not represent UNet++. Mask-RCNN is perhaps the most important meta framework for object detection, classification and segmentation.

## 3 Limitation for the State-of-the-Art-Methods

Most of the existing state-of-the-art-methods rely heavily on the existing basic image processing algorithms proposed several decades ago. All these image processing algorithms have inherent insufficiencies. However, fully automatic and unsupervised segmentation by iterative erosion is not achieved yet because of the following challenges. The great variety of cells appears with different shapes, sizes, and characteristics. There might be clutters that are indistinguishable from cells.

Classical thresholding algorithms tend to find the threshold between two-pixel classes with larger intensity interval. However, the intensity interval between the background and the darker cells might be smaller than the intensity interval between the darker cells and bright cells in the same cell image. In such situations, classical thresholding algorithms will find the wrong threshold. Overall,

thresholding is better than clustering for cell segmentation because the Gray-scales between different parts of the same cell may vary significantly in some types of cell images. The required thresholding algorithms should be robust and flexible in order to adjust to several type of cell images.

# 4 Architecture Overview: UNet++

Fig. 1a shows a high-level overview of the suggested architecture. As seen, UNet++ starts with an encoder sub-network or backbone followed by a decoder sub-network. What distinguishes UNet++ from U-Net (the black components in Fig. 1a) is the re-designed skip pathways (shown in green and blue) that connect the two sub-networks and the use of deep supervision (shown red).

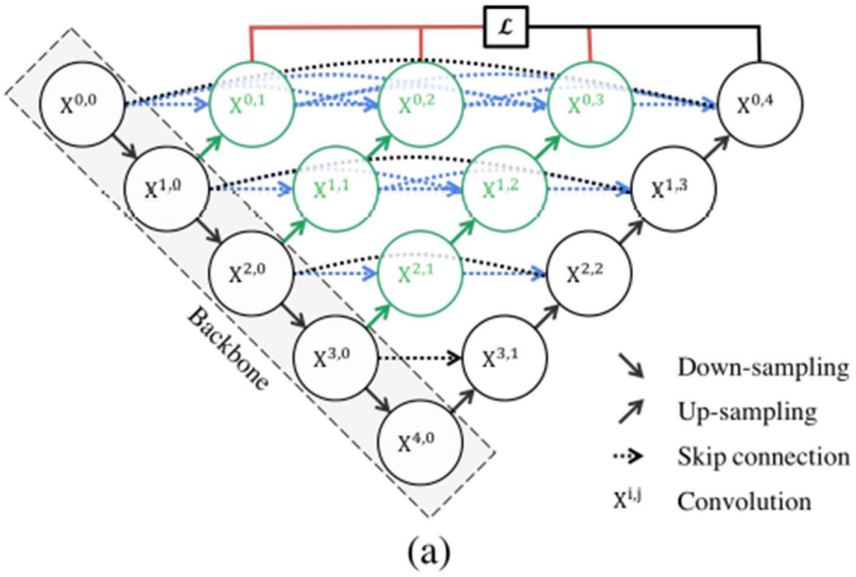



Fig. 1: (a) UNet++ consists of an encoder and decoder that are connected through a series of nested dense convolutional blocks. The main idea behind UNet++ is to bridge the semantic gap between the feature maps of the encoder and decoder prior to fusion. For example, the semantic gap between ($X^{0,0}$ ,$X^{1,3}$) is bridged using a dense convolution block with three convolution layers. In the graphical abstract, black indicates the original U-Net, green and blue show dense convolution blocks on the skip pathways, and red indicates deep supervision. Red, green, and blue components distinguish UNet++ from U-Net.

## 4.1 Redesigned Skip Pathways

In UNet++, the redesigned skip pathways (shown in green) have been added to bridge the semantic gap between the encoder and decoder subpaths.

The purpose of these convolutions layers is aimed at reducing the semantic gap between the feature maps of the encoder and decoder subnetworks. As a result, it is possibly a more straightforward optimisation problem for the optimiser to solve. Skip connections used in U-Net directly connects the feature maps between encoder and decoder, which results in fusing semantically dissimilar feature maps. However, with UNet++, the output from the previous convolution layer of the same dense block is fused with the corresponding up-sampled output of the lower dense block. This brings the semantic level of the encoded feature closer to that of the feature maps waiting in the decoder; thus optimisation is easier when semantically similar feature maps are received.

All convolutional layers on the skip pathway use kernels of size 3×3.

### 4.2 Dense Skip Connections

In UNet++, Dense skip connections (shown in blue) has implemented skip pathways between the encoder and decoder. These Dense blocks are inspired by DenseNet with the purpose to improve segmentation accuracy and improves gradient flow.

Dense skip connections ensure that all prior feature maps are accumulated and arrive at the current node because of the dense convolution block along each skip pathway. This generates full resolution feature maps at multiple semantic levels.

## 5 Materials and Methods

The study aims to determine a solution for the automatic segmentation and localization of cells. I have tried to utilise UNet++ architecture over UNet to detect cell nuclei and perform segmentation into individual objects. I believe the state-of-the-art-methods described above are not sufficient. Therefore, I propose a solution based on the UNet++ architecture for cell segmentation and extraction.

### 5.1 Image sets and experiment description

For this study, I have used an image dataset from the 2018 Kaggle Data Science Bowl which contains a large number of segmented nuclei images. The images were acquired under a variety of conditions and vary in the cell type, magnification, and imaging modality (brightfield vs. fluorescence). The dataset is designed to challenge an algorithm's ability to generalize across these variations.

*The Image segmentation dataset used in our experiment*

| Dataset | Images | Input Size | Modality | Provider |
|---|---|---|---|---|
| Cell Nuclei | 670 | 96x96 | microscopy | Data Science Bowl 2018 |

Each image is represented by an associated ImageId. Files belonging to an image are contained in a folder with this ImageId. Within this folder are two subfolders:

- images contain the image file.
- masks contain the segmented masks of each nucleus. This folder is only included in the training set. Each mask contains one nucleus. Masks are not allowed to overlap (no pixel belongs to two masks).

The second stage dataset will contain images from unseen experimental conditions. To deter hand labelling, it will also contain images that are ignored in scoring. The metric used to score this competition requires that your submissions are in run-length encoded format. Please see the evaluation page for details.

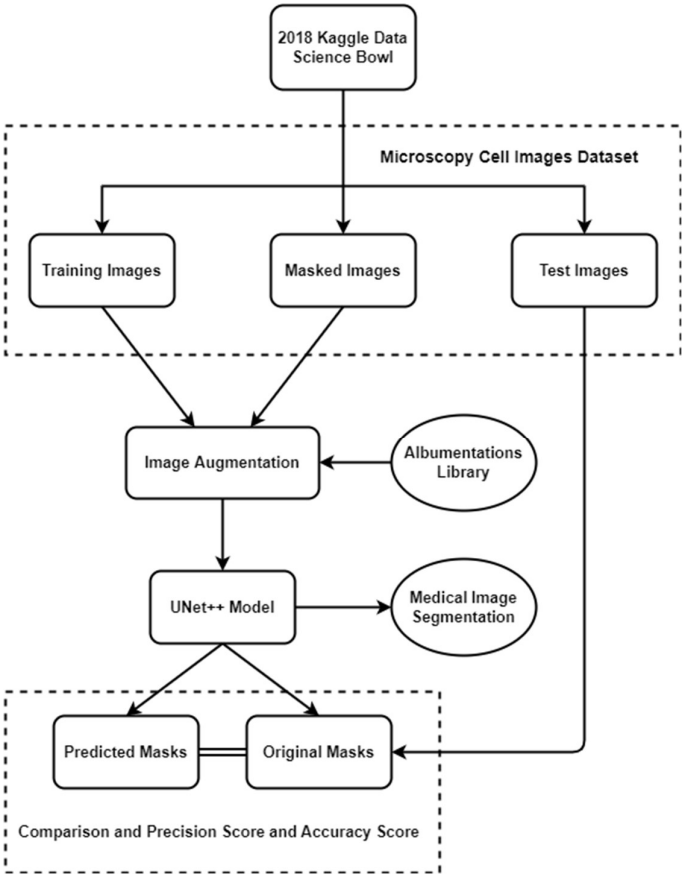
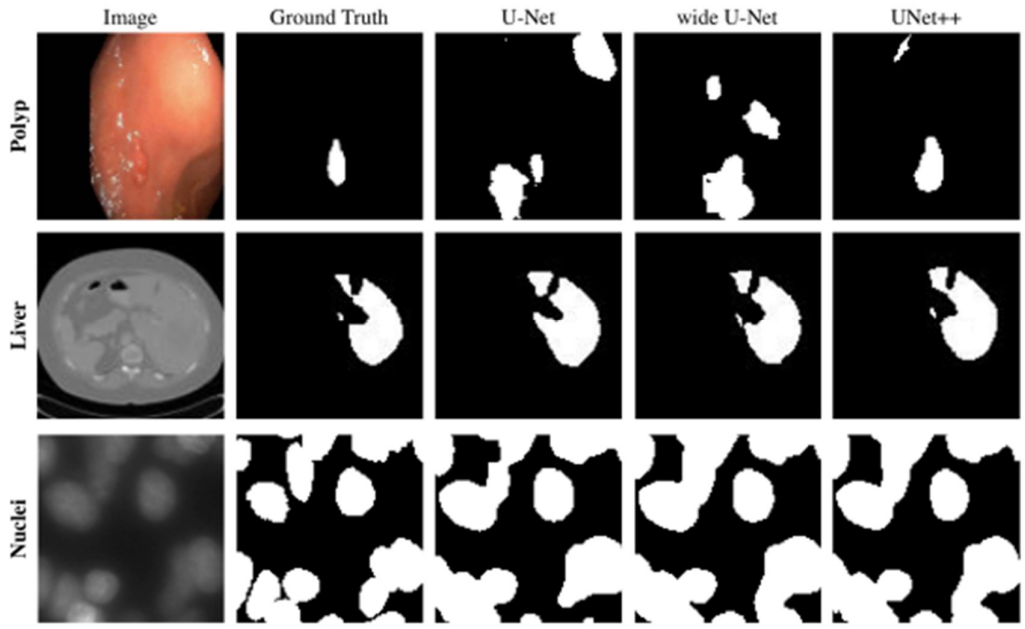



Fig. 2. Block Diagram of the overall process

As with any human-annotated dataset, you may find various forms of errors in the data. You may manually correct errors you find in the training set. The dataset will not be updated/re-released unless it is determined that there are a large number of systematic errors. The masks of the stage 1 test set will be released with the release of the stage 2 test set.




*Qualitative comparison between U-Net, wide U-Net, and UNet++, show ing segmentation results for polyp, liver, and cell nuclei datasets (2D-only for a distinct visualization).*

## 5.2 Algorithm Working

The following will discuss about the various important techniques that were utilised for the function of UNet++ architecture and to get the desired results with the performance matrix discussed in output. In many machine learning tasks, a dataset consists of three parts: a training set that is used to train the model, a validation set for choosing the best model during training, and a test set that aims to evaluate the performance of the trained model

### 5.2.1 Augmentation and Pre-processing

Data augmentation comprises two terms data and augmentation. It is a popular technique that is used to increase the generalizability of an overfitted data model. Data augmentation is a part of data analysis. It is the set of techniques that is used to increase the amount of data by adding modified copies of already existing data. Sometimes, it creates newly synthetic data from the existing data.

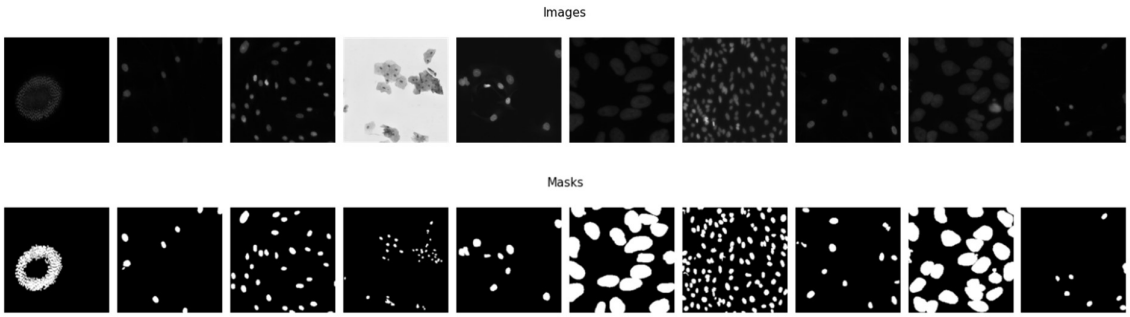
Data augmentation acts as a regularizer and assists in managing the overfitting of data. Data augmentation increases the possibility of overfitting the model by generating additional training data and exposing the model to different versions of data. Data augmentation is an element of machine learning. Machine learning is filled with diversifications and its domain increases rapidly. And data augmentation acts as a tool against these challenges. It is useful in improving

performances and outcomes of machine learning models. The data augmentation tools make the data rich and sufficient and thus makes the model perform better and accurately. Data augmentation techniques reduce the operational costs by introducing transformation in the datasets. Data augmentation assists in data cleaning, which is essential for high accuracy models. Data augmentation makes machine learning more robust by creating variations in the model.

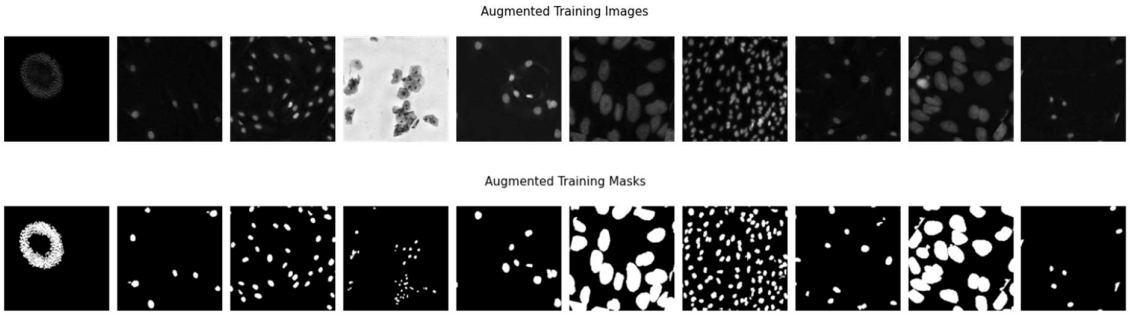
Training data was augmented using Albumentations Library. A strong combination of different types of image augmentations were applied with varied probabilities.

- CLAHE → Contrast limited adaptive histogram equalization: Takes care of over-amplification of the contrast. Operates on small regions in the image, rather than the entire image.
- Rotate
- Flip
- Gauss Noise → Adding gaussian noise to the image
- Horizontal Flip
- Vertical Flip
- Hue Saturation Value
- Random Gamma
- Random Brightness and Contrast

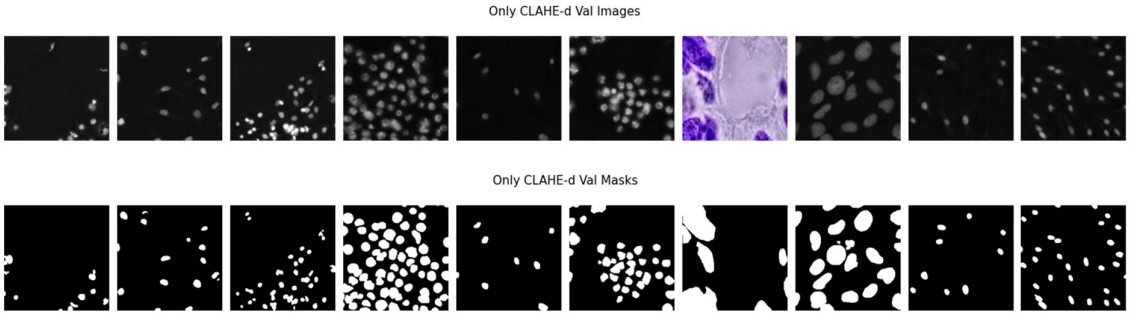

### 5.2.2 Sample Images and their Masking



### 5.2.3 Augmented Training Set

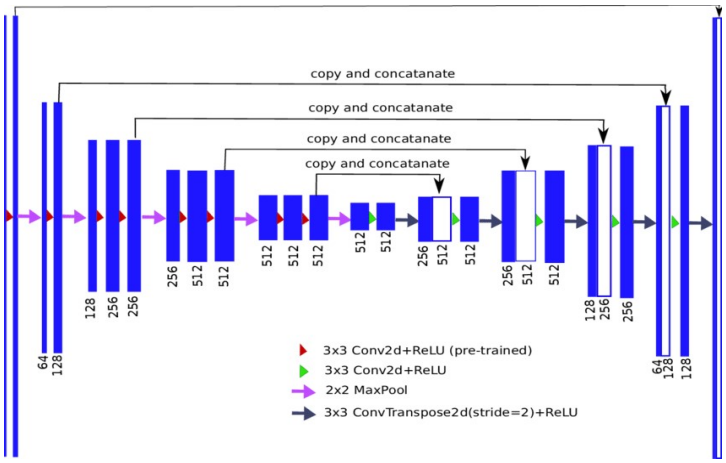



### 5.2.4 Augmented Validation Set



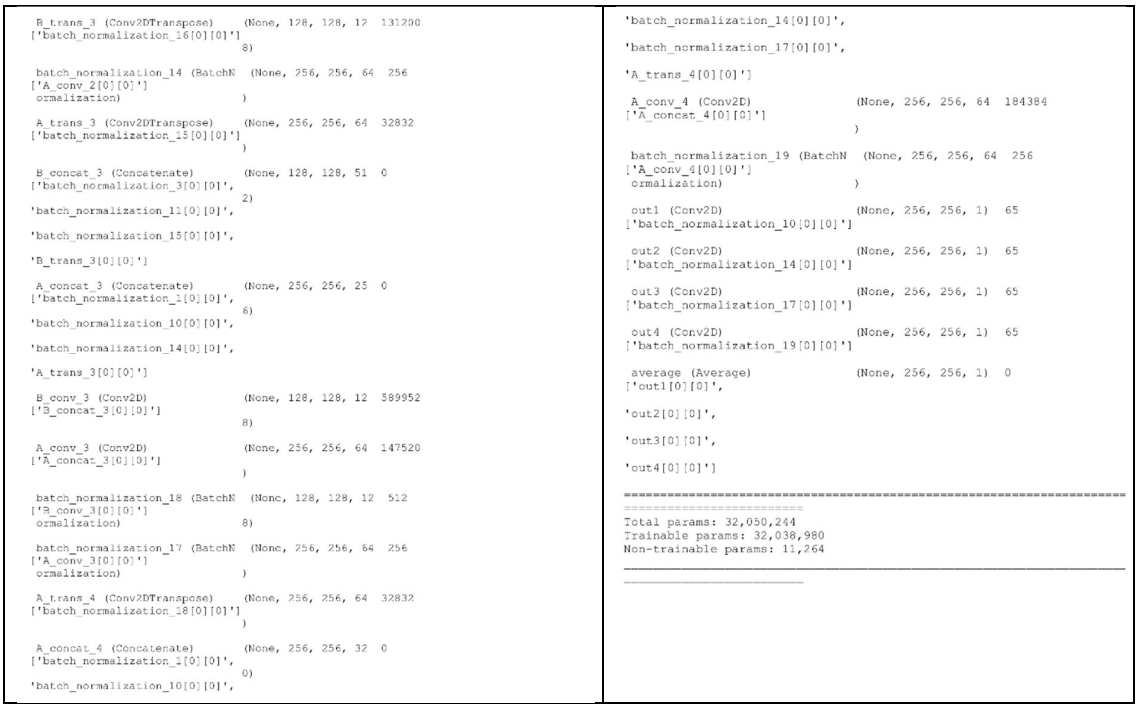

### 5.2.5 Encoder Decoder Network

The underlying hypothesis behind the architecture is that the model can more effectively capture fine-grained details of the foreground objects when high-resolution feature maps from the encoder network are gradually enriched prior to fusion with the corresponding semantically rich feature maps from the decoder network. We argue that the network would deal with an easier learning task when the feature maps from the decoder and encoder networks are semantically similar. This is in contrast to the plain skip connections commonly used in U-Net, which directly fast-forward high-resolution feature maps from the encoder to the decoder network, resulting in the fusion of semantically dissimilar feature maps



## 5.2.6 Model_summary ()

```
Model: "model"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
Connected to
==================================================================================================
 input_1 (InputLayer)        [(None, 256, 256, 3)]     0         []

 A_conv_0 (Conv2D)           (None, 256, 256, 64)      1792
['input_1[0][0]']

 batch_normalization (BatchNorm  (None, 256, 256, 64)  256
['A_conv_0[0][0]']
 alization)

 A_conv_0_1 (Conv2D)         (None, 256, 256, 64)      36928
['batch_normalization[0][0]']

 batch_normalization_1 (BatchNo  (None, 256, 256, 64)  256
['A_conv_0_1[0][0]']
 rmalization)

 max_pooling2d (MaxPooling2D)  (None, 128, 128, 64)    0
['batch_normalization_1[0][0]']

 B_conv_0 (Conv2D)           (None, 128, 128, 12       73856
['max_pooling2d[0][0]']
                             8)

 batch_normalization_2 (BatchNo  (None, 128, 128, 12)  512
['B_conv_0[0][0]']
 rmalization)                8)

 B_conv_0_1 (Conv2D)         (None, 128, 128, 12       147584
['batch_normalization_2[0][0]']
                             8)

 batch_normalization_3 (BatchNo  (None, 128, 128, 12)  512
['B_conv_0_1[0][0]']
 rmalization)                8)

 max_pooling2d_1 (MaxPooling2D)  (None, 64, 64, 128)   0
['batch_normalization_3[0][0]']

 C_conv_0 (Conv2D)           (None, 64, 64, 256)       295168
['max_pooling2d_1[0][0]']

 batch_normalization_4 (BatchNo  (None, 64, 64, 256)   1024
['C_conv_0[0][0]']
 rmalization)

 C_conv_0_1 (Conv2D)         (None, 64, 64, 256)       590080
['batch_normalization_4[0][0]']

 batch_normalization_5 (BatchNo  (None, 64, 64, 256)   1024
['C_conv_0_1[0][0]']
 rmalization)

 max_pooling2d_2 (MaxPooling2D)  (None, 32, 32, 256)   0
['batch_normalization_5[0][0]']

 D_conv_0 (Conv2D)           (None, 32, 32, 512)       1180160
['max_pooling2d_2[0][0]']

 batch_normalization_6 (BatchNo  (None, 32, 32, 512)   2048
['D_conv_0[0][0]']
 rmalization)

 D_conv_0_1 (Conv2D)         (None, 32, 32, 512)       2359808
['batch_normalization_6[0][0]']

 batch_normalization_7 (BatchNo  (None, 32, 32, 512)   2048
['D_conv_0_1[0][0]']
 rmalization)

 max_pooling2d_3 (MaxPooling2D)  (None, 16, 16, 512)   0
['batch_normalization_7[0][0]']

 E_conv_0 (Conv2D)           (None, 16, 16, 1024       4719616
['max_pooling2d_3[0][0]']
                             )

 batch_normalization_8 (BatchNo  (None, 16, 16, 1024)  4096
['E_conv_0[0][0]']
 rmalization)                )

 E_conv_0_1 (Conv2D)         (None, 16, 16, 1024       9438208
['batch_normalization_8[0][0]']
                             )

 batch_normalization_9 (BatchNo  (None, 16, 16, 1024)  4096
['E_conv_0_1[0][0]']
 rmalization)                )

 D_trans_1 (Conv2DTranspose)  (None, 32, 32, 512)      2097664
['batch_normalization_9[0][0]']

 C_trans_1 (Conv2DTranspose)  (None, 64, 64, 256)      524544
['batch_normalization_7[0][0]']

 D_concat_1 (Concatenate)    (None, 32, 32, 1024       0
['batch_normalization_7[0][0]',
                             )
'D_trans_1[0][0]']

 B_trans_1 (Conv2DTranspose)  (None, 128, 128, 12      131200
['batch_normalization_5[0][0]']
                             8)

 C_concat_1 (Concatenate)    (None, 64, 64, 512)       0
['batch_normalization_5[0][0]',

'C_trans_1[0][0]']

 D_conv_1 (Conv2D)           (None, 32, 32, 512)       4719104
['D_concat_1[0][0]']

 A_trans_1 (Conv2DTranspose)  (None, 256, 256, 64)     32832
['batch_normalization_3[0][0]']
                             )

 B_concat_1 (Concatenate)    (None, 128, 128, 25       0
['batch_normalization_3[0][0]',
                             6)
'B_trans_1[0][0]']

 C_conv_1 (Conv2D)           (None, 64, 64, 256)       1179904
['C_concat_1[0][0]']

 batch_normalization_13 (BatchN  (None, 32, 32, 512)   2048
['D_conv_1[0][0]']
 ormalization)

 A_concat_1 (Concatenate)    (None, 256, 256, 12       0
['batch_normalization_1[0][0]',
                             8)
'A_trans_1[0][0]']

 B_conv_1 (Conv2D)           (None, 128, 128, 12       295040
['B_concat_1[0][0]']
                             8)

 batch_normalization_12 (BatchN  (None, 64, 64, 256)   1024
['C_conv_1[0][0]']
 ormalization)

 C_trans_2 (Conv2DTranspose)  (None, 64, 64, 256)      524544
['batch_normalization_13[0][0]']

 A_conv_1 (Conv2D)           (None, 256, 256, 64       73792
['A_concat_1[0][0]']
                             )

 batch_normalization_11 (BatchN  (None, 128, 128, 12)  512
['B_conv_1[0][0]']
 ormalization)               8)

 B_trans_2 (Conv2DTranspose)  (None, 128, 128, 12      131200
['batch_normalization_12[0][0]']
                             8)

 C_concat_2 (Concatenate)    (None, 64, 64, 768)       0
['batch_normalization_5[0][0]',

'batch_normalization_12[0][0]',

'C_trans_2[0][0]']

 batch_normalization_10 (BatchN  (None, 256, 256, 64)  256
['A_conv_1[0][0]']
 ormalization)               )

 A_trans_2 (Conv2DTranspose)  (None, 256, 256, 64)     32832
['batch_normalization_11[0][0]']
                             )

 B_concat_2 (Concatenate)    (None, 128, 128, 38       0
['batch_normalization_3[0][0]',
                             4)
'batch_normalization_11[0][0]',

'B_trans_2[0][0]']

 C_conv_2 (Conv2D)           (None, 64, 64, 256)       1769728
['C_concat_2[0][0]']

 A_concat_2 (Concatenate)    (None, 256, 256, 19       0
['batch_normalization_1[0][0]',
                             2)
'batch_normalization_10[0][0]',

'A_trans_2[0][0]']

 B_conv_2 (Conv2D)           (None, 128, 128, 12       442496
['B_concat_2[0][0]']
                             8)

 batch_normalization_16 (BatchN  (None, 64, 64, 256)   1024
['C_conv_2[0][0]']
 ormalization)

 A_conv_2 (Conv2D)           (None, 256, 256, 64       110656
['A_concat_2[0][0]']
                             )

 batch_normalization_15 (BatchN  (None, 128, 128, 12)  512
['B_conv_2[0][0]']
 ormalization)               8)
```




...

*Epoch 18/20*

50/50 [==============================] - 84s 2s/step - loss: 0.3522 - accuracy: 0.9619 - iou: 0.7837 - auc: 0.9789 - val_loss: 0.2346 - val_accuracy: 0.9761 - val_iou: 0.8409 - val_auc: 0.9918 - lr: 5.0000e-05; Learning rate:  5e-05

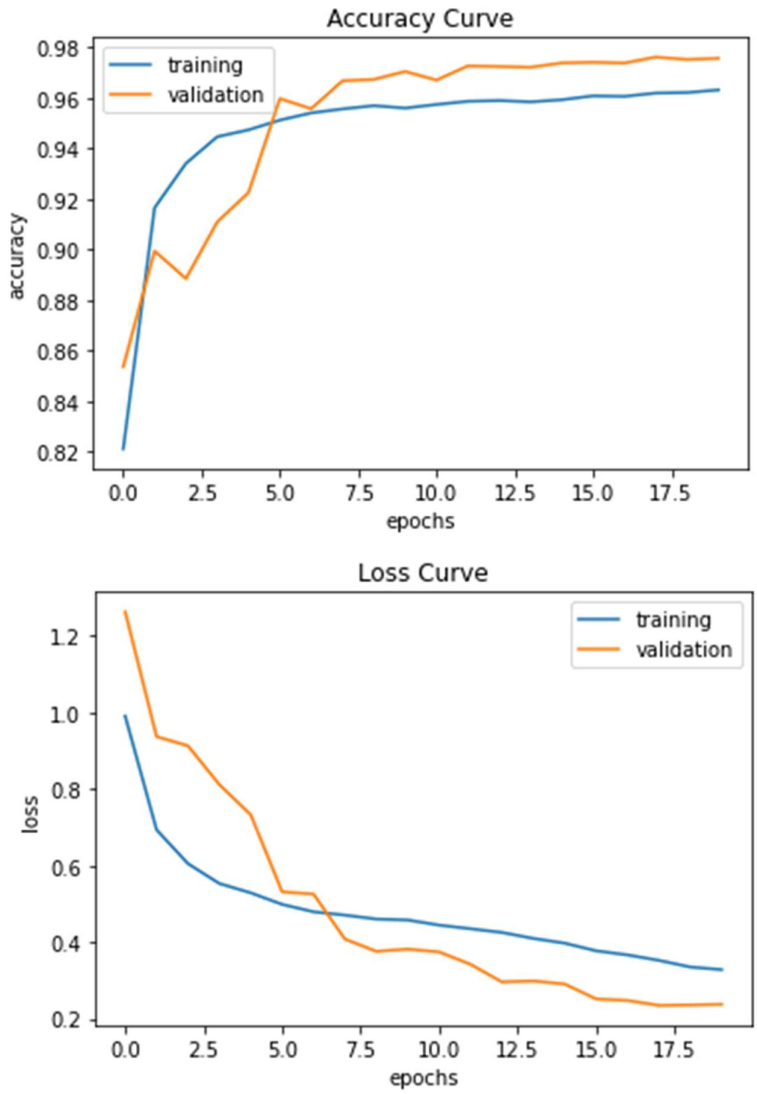
*Epoch 19/20*

50/50 [==============================] - 84s 2s/step - loss: 0.3348 - accuracy: 0.9621 - iou: 0.7848 - auc: 0.9819 - val_loss: 0.2356 - val_accuracy: 0.9752 - val_iou: 0.8385 - val_auc: 0.9919 - lr: 5.0000e-05; Learning rate:  5e-05

*Epoch 20/20*

50/50 [==============================] - 84s 2s/step - loss: 0.3279 - accuracy: 0.9631 - iou: 0.7902 - auc: 0.9817 - val_loss: 0.2373 - val_accuracy: 0.9756 - val_iou: 0.8389 - val_auc: 0.9911 - lr: 5.0000e-05

# 6 Results Obtained




To evaluate the models, five metrics are employed for verifying the quality of the segmentation results, including the Dice Similarity Coefficient (DSC), Accuracy, Specificity, Precision and Recall criteria.

$$\text{DSC} = \frac{2\text{TP}}{\text{FP} + 2\text{TP} + \text{FN}}$$

$$\text{Accuracy} = \frac{\text{TP} + \text{TN}}{\text{TP} + \text{TN} + \text{FP} + \text{FN}}$$

$$\text{Specificity} = \frac{\text{TN}}{\text{FP} + \text{TN}}$$

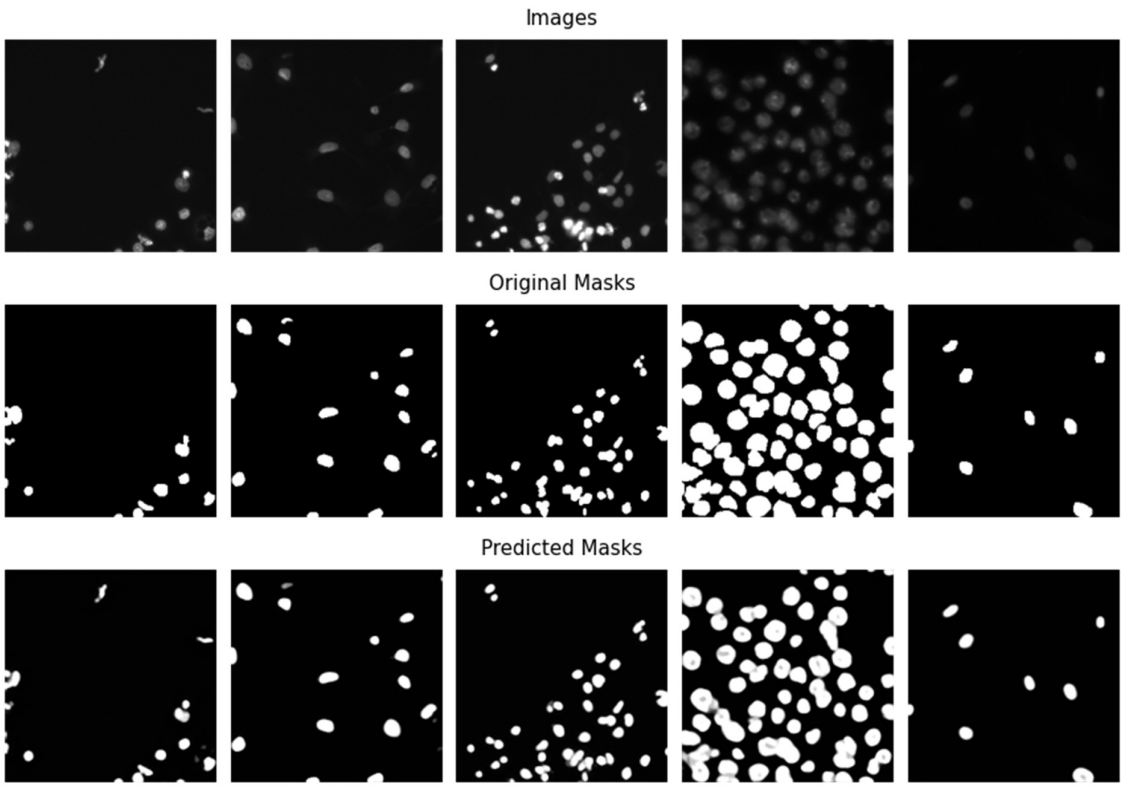
$$\text{Precision} = \frac{\text{TP}}{\text{TP} + \text{FP}}$$

$$\text{Recall} = \frac{\text{TP}}{\text{TP} + \text{FN}}$$

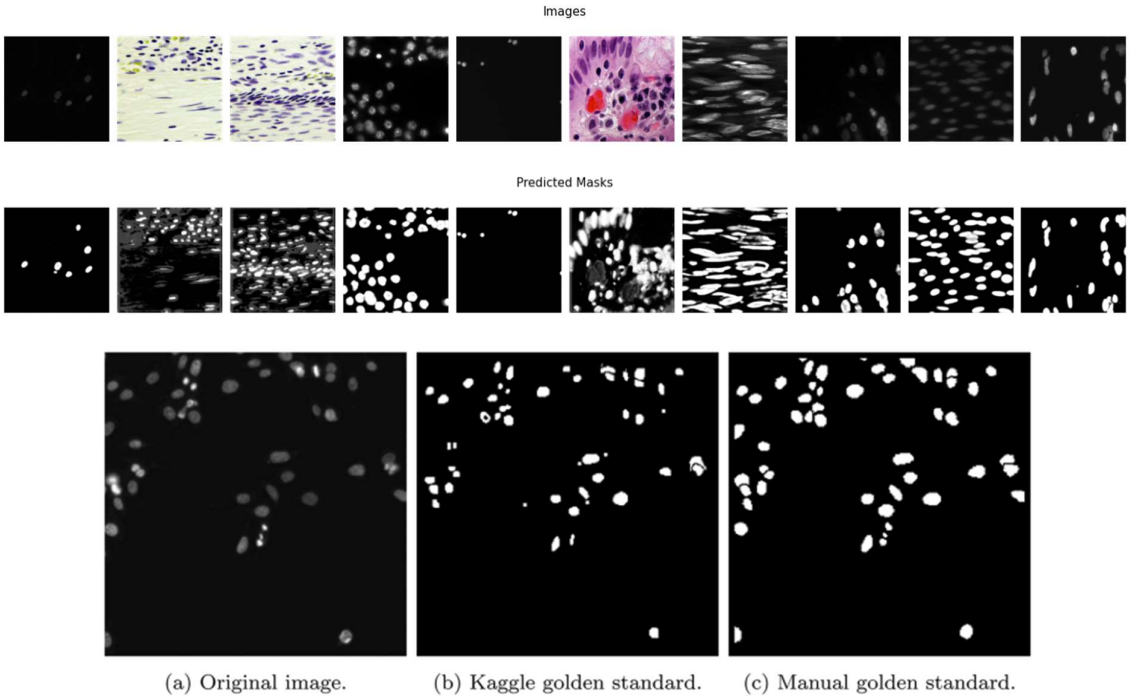
TP, FP, TN and FN are the numbers of true positive, false positive, true negative and false negative detections.

| | |
|---|---|
| **Precision Score** | 0.91 |
| **Recall Score** | 0.85 |
| **F1-Score** | 0.88 |
| **Sensitivity** | 0.85 |
| **Specificity** | 0.99 |
| **IOU** | 0.79 |
| **AUC** | 0.92 |

| | **Precision** | **Recall** | **F1-Score** | **Support** |
|---|---|---|---|---|
| **False** | 0.98 | 0.99 | 0.98 | 5707370 |
| **True** | 0.91 | 0.85 | 0.88 | 846230 |
| **Accuracy** | | | 0.97 | 6553600 |
| **Macro avg** | 0.95 | 0.92 | 0.93 | 6553600 |
| **Weighted avg** | 0.97 | 0.97 | 0.97 | 6553600 |



### 6.1 Final Result




(a) Original image. (b) Kaggle golden standard. (c) Manual golden standard.

## 7 Applications

- The main goal of segmenting this data is to identify areas of the anatomy required for a particular study, for example, to simulate physical properties or virtually positioning CAD-designed implants within a patient.
- Blood cells count plays a vital role in determining a person's health rate. Doctors perform a complete blood count in order to obtain important information such as kinds and number of cells in the blood.

## 8 Future of Cell Segmentation

It has taken the community many decades and great endeavour to segment, identify, and analyze cells for explaining the cellular and molecular processes of health and disease. Studies have showed that cell segmentation can contribute to the better cell recognition in various fields of cell biology (1–14) and great advances have been achieved. However, it still has a long way to go before meeting the ultimate goal that the developed method can segment and identify different types of cells autonomously and the biologists can trust the segmentation and identification results blindly. To achieve this ultimate goal, researchers all over the world including biologists, computer scientists, and software engineers should work together closely and industriously.

With the rise of size and complexity of cell images, the requirements for cell segmentation methods are also increasing. The basic image processing algorithms developed decades ago should not be the golden standards for dealing with these challenging cell segmentation problems any more. On the contrary, development of more effective image processing algorithms is more promising for the progress of cell segmentation. In the meantime, comparison of these newly developed algorithms and teaching the biologists to use these newly developed algorithms are also very important. Hence, the open access and authoritative platforms are necessary for researchers all over the world to share, learn, and teach the data, codes, and algorithms.

# 9 Conclusion

To address the need for more accurate medical image segmentation, we formed UNet++. The suggested architecture takes advantage of re-designed skip pathways and deep supervision. The re-designed skip pathways aim at reducing the semantic gap between the feature maps of the encoder and decoder sub networks, resulting in a possibly simpler optimization problem for the optimizer to solve. Deep supervision also enables more accurate segmentation particularly for lesions that appear at multiple scales such as polyps in colonoscopy videos. We evaluated UNet++ using one medical imaging datasets which comprised of cell nuclei segmentation.

# References


[1] Junbong Jang, Chuangqi Wang, Xitong Zhang, Hee June Choi, Xiang Pan, Bolun Lin, Yudong Yu, Carly Whittle, Madison Ryan, Yenyu Chen, Kwonmoo Lee,A deep learning-based segmentation pipeline for profiling cellular morphodynamics using multiple types of live cell microscopy, Cell Reports Methods, Volume 1, Issue 7,2021,100105,ISSN 2667-2375, (https://doi.org/10.1016/j.crmeth.2021.100105).

[2] Johannes Stegmaier, Fernando Amat, William C. Lemon, Katie McDole, Yinan Wan, George Teodoro, Ralf Mikut, Philipp J. Keller, Real-Time Three-Dimensional Cell Segmentation in Large-Scale Microscopy Data of Developing Embryos,Developmental Cell,Volume 36, Issue 2,2016,Pages 225-240,ISSN 1534-5807,https://doi.org/10.1016/j.devcel.2015.12.028.

[3] Katarzyna Hajdowska, Sebastian Student, Damian Borys,Graph based method for cell segmentation and detection in live-cell fluorescence microscope imaging,Biomedical Signal Processing and Control,Volume 71, Part A,2022,103071,ISSN 1746-8094,https://doi.org/10.1016/j.bspc.2021.103071.

[4] De Rong Loh, Wen Xin Yong, Jullian Yapeter, Karupppasamy Subburaj, Rajesh Chandramohanadas, A deep learning approach to the screening of malaria infection: Automated and rapid cell counting, object detection and instance segmentation using Mask R-CNN, Computerized Medical Imaging and Graphics, Volume 88, 2021, 101845, ISSN 0895-6111, https://doi.org/10.1016/j.compmedimag.2020.101845. (https://www.sciencedirect.com/science/article/pii/S0895611120301403)

[5] Chiu-Han Hsiao, Tzu-Lung Sun, Ping-Cherng Lin, Tsung-Yu Peng, Yu-Hsin Chen, Chieh-Yun Cheng, Feng-Jung Yang, Shao-Yu Yang, Chih-Horng Wu, Frank Yeong-Sung Lin, Yennun Huang, A deep learning-based precision volume calculation approach for kidney and tumor segmentation on computed tomography images, Computer Methods and Programs in Biomedicine, Volume 221, 2022, 106861, ISSN 0169-2607, https://doi.org/10.1016/j.cmpb.2022.106861.

[6] Zhou, Z., Rahman Siddiquee, M.M., Tajbakhsh, N., Liang, J. (2018). UNet++: A Nested U-Net Architecture for Medical Image Segmentation. In: , *et al.* Deep Learning in Medical Image Analysis and Multimodal Learning for Clinical Decision Support. DLMIA ML-CDS 2018 2018. Lecture Notes in Computer Science(), vol 11045. Springer, Cham. https://doi.org/10.1007/978-3-030-00889-5_1

[7] Dimitris Metaxas, Hui Qu, Gregory Riedlinger, Pengxiang Wu, Qiaoying Huang, Jingru Yi, Subhajyoti De, Chapter 8 - Deep learning-based nuclei segmentation and classification in histopathology images with application to imaging genomics, Editor(s): Mei Chen, In Computer Vision and Pattern Recognition, Computer Vision for Microscopy Image Analysis, Academic Press, 2021, Pages 185-201, ISBN 9780128149720, https://doi.org/10.1016/B978-0-12-814972-0.00008-4.

[8] Fatemeh Hoorali, Hossein Khosravi, Bagher Moradi, Automatic Bacillus anthracis bacteria detection and segmentation in microscopic images using UNet++, Journal of Microbiological Methods, Volume 177, 2020, 106056, ISSN 0167-012, https://doi.org/10.1016/j.mimet.2020.106056.

[9] Ryan Jacobs, Mingren Shen, Yuhan Liu, Wei Hao, Xiaoshan Li, Ruoyu He, Jacob R.C. Greaves, Donglin Wang, Zeming Xie, Zitong Huang, Chao Wang, Kevin G. Field, Dane Morgan,Performance and limitations of deep learning semantic segmentation of multiple defects in transmission electron micrographs,Cell Reports Physical Science,Volume 3, Issue 5,2022,100876,ISSN 2666-3864,https://doi.org/10.1016/j.xcrp.2022.100876.

[10] Rew, J, Kim, H & Hwang, E 2021, 'Hybrid segmentation scheme for skin features extraction using dermoscopy images', *Computers, Materials and Continua*, vol. 69, no. 1, pp. 801-817. https://doi.org/10.32604/cmc.2021.017892

[11] Junbong Jang, Chuangqi Wang, Xitong Zhang, Hee June Choi, Xiang Pan, Bolun Lin, Yudong Yu, Carly Whittle, Madison Ryan, Yenyu Chen, Kwonmoo Lee, A deep learning-based segmentation pipeline for profiling cellular morphodynamics using multiple types of live cell microscopy,Cell Reports Methods, Volume 1, Issue 7, 2021,100105, ISSN 2667-2375, https://doi.org/10.1016/j.crmeth.2021.100105.

[12] Vicar, T., Balvan, J., Jaros, J. *et al.* Cell segmentation methods for label-free contrast microscopy: review and comprehensive comparison. *BMC Bioinformatics* 20, 360 (2019). https://doi.org/10.1186/s12859-019-2880-8

[13] E. Meijering, "Cell Segmentation: 50 Years Down the Road [Life Sciences]," in IEEE Signal Processing Magazine, vol. 29, no. 5, pp. 140-145, Sept. 2012, doi: 10.1109/MSP.2012.2204190.

[14] Wang, Z., 2019. Cell segmentation for image cytometry: advances, insufficiencies, and challenges. *Cytometry A*, *95*(7), pp.708-711. https://onlinelibrary.wiley.com/doi/abs/10.1002/cyto.a.23686

[15] Dimopoulos, S., Mayer, C.E., Rudolf, F. and Stelling, J., 2014. Accurate cell segmentation in microscopy images using membrane patterns. *Bioinformatics*, *30*(18), pp.2644-2651 https://academic.oup.com/bioinformatics/article/30/18/2644/2475348

[16] Dawoud, Y., Hornauer, J., Carneiro, G., Belagiannis, V. (2021). Few-Shot Microscopy Image Cell Segmentation. In: Dong, Y., Ifrim, G., Mladenić, D., Saunders, C., Van Hoecke, S. (eds) Machine Learning and Knowledge Discovery in Databases. Applied Data Science and Demo Track. ECML PKDD 2020. Lecture Notes in Computer Science(), vol 12461. Springer, Cham. https://doi.org/10.1007/978-3-030-67670-4_9

[17] Liu, Xiangbin & Song, Liping & Liu, Shuai & Zhang, Yu-Dong. (2021). A Review of Deep-Learning-Based Medical Image Segmentation Methods. Sustainability. 13. 1224. 10.3390/su13031224.