Multi-organ Nucleus Segmentation Using UNet++

Yixuan Deng
University of Michigan, Ann Arbor
yixuand@umich.edu

Xuhui Guo
University of Michigan, Ann Arbor
xuhuiguo@umich.edu

Xinru Song
University of Michigan, Ann Arbor
xinrus@umich.edu

Zhiran Wang
University of Michigan, Ann Arbor
zhiran@umich.edu

1 Introduction

The major component of this project is basically a semantic segmentation task for the nucleus in H&E stained histology images of multi-organ tumor cells. Accurate segmentation of cell nuclei allows the analysis of key features like density, size ratios, and shape variations. These are critical for evaluating cancer severity and predicting treatment outcomes. In this project, we will employ UNet++[10] model to perform this task. From a performance perspective, the innovatively designed UNet++ model demonstrates significant improgements compared to the traditional segmentation model UNet[7], enabling more precise image segmentation tasks. While this advancement may not be of paramount significance in the segmentation of natural images, it hold profound importance in the realm of medical image analysis since even minor segmentation errors can potentially lead to divergent diagnostic outcomes. Therefore, the heightened accuracy provided by UNet++ is crucial for obtaining precise clinically meaningful insights from medical imaging data.

2 Problem Statement

To be more specific, we mainly focus on the nucleus segmentation task on MoNuSeg 2018 challenge dataset with UNet++.

The detailed structure will be discussed in method session and here we will dig a little bit in every node in UNet++. Each node in U-Net++ is defined by:

$$X^{(i,j)} = C\left(U\left(X^{(i-1,j)}\right) + X^{(i,j-1)}\right) \tag{1}$$

where $X^{(i,j)}$ represents the feature map at depth i and layer j, C is the convolution operation, and U is the upsampling operation.

The output of the network is reconstructed through cross-layer connections and feature integration:

$$Y = \sigma\left(\sum_{i=0}^{D} W_i \cdot X^{(i,0)}\right) \tag{2}$$

where Y is the final output or segmentation map, σ is an activation function such as softmax, W_i are the weights that combine features from different depths, and D is the maximum depth of the network.

3 Related Work

3.1 Semantic Segmentation

Based on FCN: Fully convolutional network[6] was first introduced in 2015. It improves upon the VGG-16 network by replacing the fully connected layers in traditional CNNs with convolutional

layers. Then, using a skip layer approach to combine feature maps produced by intermediate convolutional layers. After that, through the bilinear interpolation (BI) algorithm for upsampling and finally converts coarse segmentation results into fine-grained segmentation results.

DeepLab related models: DeepLab[2]processes images by an FCN combined with the Hole algorithm to obtain rough feature maps. It then uses BI algorithm to upsample the FCN output and get a coarse segmentation image. After that, fully connected conditional random field is used to generate the image semantic segmentation result.

Compared to DeepLab, DeepLab-V2[3] network not only uses atrous convolution as the upsampling filter for dense feature extraction, but also combines it with the spatial pyramid pooling method[5]. Then, it introduces atrous spatial pyramid pooling (ASPP) and uses it to integrate multi-scale features. Consequently, this network increases the receptive field and improves segmentation accuracy without significantly increasing the number of parameters.

Based on Encoder-Decoder: UNet[7] is a typical encoder-decoder network architecture used for semantic segmentation. The encoding process involves downsampling operations which gradually reduce the resolution of the feature maps. In the decoding process, upsampling operations are performed, which gradually restore the details of objects and the resolution of the image.

SegNet-Basic[1] is another example of this type of network architecture. It calculates the classification of each pixel based on prior probabilities and features a symmetric structure. The left side of the network consists of an encoder containing a fully convolutional network that performs downsampling through convolution and pooling operations. The right side of the network is a decoder composed of a deconvolutional network that uses transposed convolutions to perform upsampling.

3.2 Deep Supervision

Deep supervision is a technique used in training deep neural networks by introducing additional loss functions at different layers within the network. This approach helps to combat potential optimization obstacles and reach a faster convergence rate. Dou et al.[4] supports that statement by introducing a deep supervision combined with predictions from varying resolutions of feature maps. In addition, Zhou et al.[10] suggested that UNet++ is suitable for deep supervision because "the high dimension features affect every output through back-propagation" and "the networks are embedded at various depths of U-Net". Within the process of supervision, the output of each subnetwork is already the result of image segmentation. In that case, if the output of the smaller sub-network is good enough, then we can prune those redundant parts freely. That allows the parameter-heavy deep network to reduce the number of parameters within an acceptable range of accuracy, which makes it less challenging for UNet++ to be deployed under resource-constrained environments.

4 Methods

4.1 Data Preprocessing

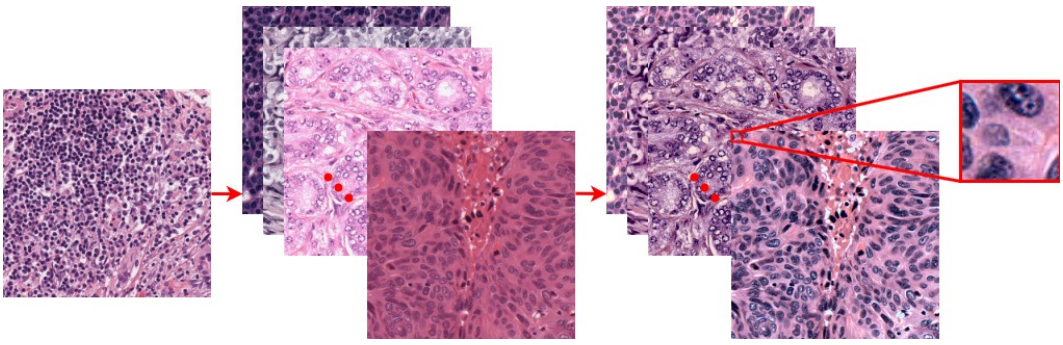

Figure 1: Flow chart for the preprocessing steps.

In our project, we generally conducted 2 preprocessing steps on the raw dataset prior to model training.

Color Transfer: Firstly, the raw dataset exhibited inconsistencies in cellular image staining, which could result in variations in cellular structures and features across different images, thereby impacting the model's performance and stability. Consequently, we employed a Color Transfer[8] function to correct the staining, effectively reducing errors in model processing of diverse images and enhancing its robustness, thus enabling more reliable application in cellular image analysis tasks.

Sliding Window Augmentation: Secondly, to ensure comprehensive model training, we utilized a sliding window of size 96*96 to augment the dataset from 30+ to over 10,000 images, thereby fully leveraging the available data resources and mitigating the impact of data scarcity on model training.

4.2 Unet++ Model

U-Net++ is an innovative modification of the U-Net architecture, specifically optimized for the task of biomedical image segmentation.The paper believed that the way that U-Net combines the encoder and decoder is asymmetric and needed to be explored by convolutions and concatenation. Therefore, it introduces several key architectural innovations that address limitations of the original U-Net, particularly in capturing fine-grained details and bridging the semantic gap between encoder and decoder feature maps showed as Figure 2.

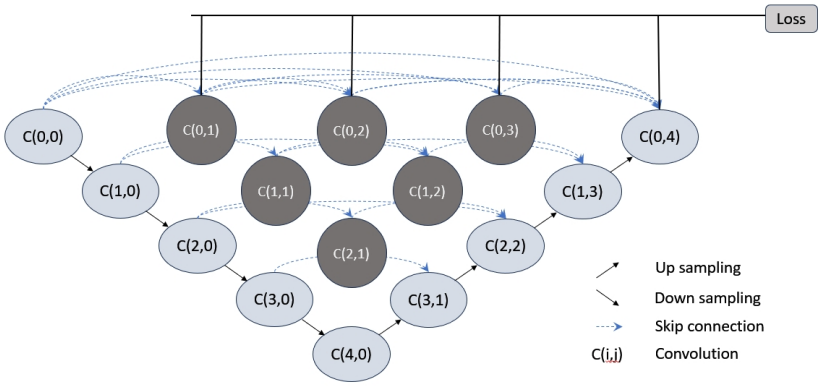

Figure 2: Unet++ Structure

4.2.1 Encoder and Decoder with Nested Skip Connections

Encoder: The encoder in U-Net++ is responsible for capturing the context in the input images. This is achieved through a series of convolutional layers, each followed by batch normalization and ReLU activation functions. Convolutional Layers apply a set of learnable filters to the input image or feature maps from the previous layer. The purpose is to extract high-level features such as edges, textures, and other relevant structures in the image. The convolution operation maintains the spatial hierarchy between input and output feature maps, which is crucial for capturing local contexts in images.

Decoder: The decoder utilizes upsampling and convolutional layers to reconstruct the segmentation map from encoded features. Upsampling is typically achieved through transposed convolution or simple interpolation methods. Upsampling gradually increases the resolution of the input feature maps to reconstruct the finer details needed for precise localization in the segmentation map. After each upsampling step, convolutional layers refine the upsampled feature maps, adjusting and improving the clarity and quality of the output. These layers help in recovering the information lost during downsampling in the encoder.

Nested Skip Connections: Nested skip connections in U-Net++ significantly enhance the model's ability to capture and utilize multi-scale spatial information compared to the original U-Net. Instead of direct skip connections, U-Net++ employs a sophisticated network of intermediate convolutional nodes that bridge the encoder and decoder layers.

These nodes are arranged in a dense grid, where each node receives inputs from both the prior node in the same layer and the corresponding feature map in the encoder via upward connections, as well as from the output of the previous layer's node in the decoder via horizontal connections. This

configuration creates a rich, multi-path feature flow that enhances feature propagation across the network and improves gradient flow during backpropagation.

4.2.2 Deep Supervision

Deep supervision is implemented by adding auxiliary outputs at various levels of the decoder. These contribute to the loss during training, enhancing learning at multiple scales and improving convergence.

Implementing deep supervision requires careful consideration of how the auxiliary losses are weighted relative to the main loss. If the auxiliary losses are weighted too heavily, they might dominate the training process, leading to poor performance on actual output targets. Conversely, if they are weighted too lightly, they might not provide sufficient signal to earlier layers. Balancing these weights is key to leveraging the full benefits of deep supervision in U-Net++.

Deep supervision in U-Net++ represents a significant advancement in the design of segmentation networks, allowing for more nuanced and effective training of deep neural networks, particularly in the challenging domain of medical image analysis.

4.2.3 Feature Fusion and Final Segmentation Map

The final segmentation output is obtained by fusing the outputs from multiple final layer nodes, utilizing learnable weights to effectively integrate multi-scale information.

5 Experiment

5.1 Dataset

Utilizing a novel dataset for model implementation can foster innovation and provide a better assessment of the algorithm's generalization capabilities. Therefore, we opted to innovatively utilize preprocessed MoNuSeg 2018 challenge dataset[1] as a novel dataset. After preprocessing, our dataset comprises 10,830 H&E stained tissue images captured at 40x magnification from the TCGA archive and around 22,000 nuclear boundary annotations. To ensure that the training set is sufficiently large to train the model parameters, while also enabling timely monitoring of the model's performance and validating its generalization ability,we randomly assigned the images into training set (80%), validation set (10%), and test set (10%).

	Percentage	Number of Images
Training Set	80%	8664
Validation Set	10%	1083
Testing Set	10%	1083

Table 1: Statistics for the augmented dataset.

5.2 Implementation Details

We trained our UNet++ model without the Deep Supervision module on the revised MoNuSeg 2018 challenge dataset using an NVIDIA GeForce RTX 4060 GPU with 16GB VRAM for 100 epochs. The Combo Loss [9], namely the summation of Cross Entropy Loss and Dice Loss, was selected as our loss function. The model is trained with a batch size of 16. An Adam optimizer is employed to adjust the model's parameters, with a specified learning rate set as 1e-4 to guide the optimization process. Additionally, a learning rate scheduler is utilized, which reduces the learning rate by a factor of 0.1 every 10 epochs to help in stabilizing the training as it progresses. During training, we kept track of training loss and evaluate the model on the validation set based on Dice coefficient and Intersection over Union (IoU) for just the foreground part (the nucleus) per epoch.

[1] `https://monuseg.grand-challenge.org/Data/`

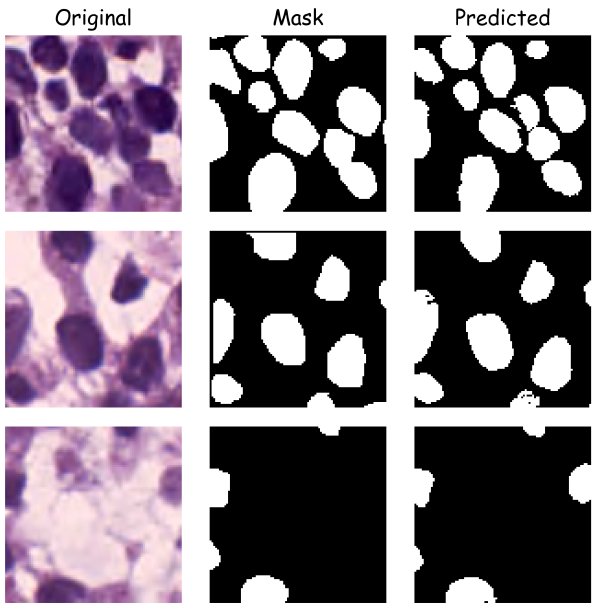

Figure 3: Qualitative comparison of model inference results on single slice between different levels of cell densities

5.3 Results and Evaluation

5.3.1 Metrics

Intersection over Union (IoU): The Intersection over Union (IoU) score is a key metric used in nucleus segmentation to quantify the accuracy of predicted segmentations, which is defined as:

$$\text{IoU} = \frac{\text{Area of Intersection}}{\text{Area of Union}}$$

It measures the overlap between the predicted and actual segmented areas, providing a direct assessment of segmentation precision and reliability in identifying individual nuclei.

Dice coefficient: Unlike the IoU, which divides the intersection by the union of predicted and ground truth areas, the Dice coefficient, which is defined as:

$$\text{Dice} = \frac{2 \times \text{Area of Intersection}}{\text{Area of Prediction} + \text{Area of Ground Truth}}$$

divides twice the intersection by the sum of both areas, often yielding slightly higher values due to its method of normalization, especially in cases of small overlaps.

Due to the relatively small number of foreground pixels compared with the extremely larger proportion of background pixels in the nucleus segmentation task, we primarily focus on the foreground IoU metric to evaluate the model's performance.

5.3.2 Results and Discussion

Table 2 shows the foreground class IoU, and the Dice metrics on the test set. Figure 3 shows a qualitative comparison of the model inference results on a single slice with varying cell densities.

Data	Model	IoU(%)	Dice(%)
MoNuSeg 2018	UNet++	67.85	74.15

Table 2: Quantitative evaluation of the trained UNet++ model w\o deep supervision on the test set of revised MoNuSeg 2018 challenge dataset in terms of Foreground Class IoU and Dice coefficient.

The UNet++ model achieves respectable results in nucleus segmentation with an IoU of 67.85 and a Dice coefficient of 74.15, indicative of its competency in segmenting cell nuclei. From the qualitative perspective, UNet++ displays variability in handling different cell densities within a single slice. It seems to maintain the general morphology of the nuclei without significant over-segmentation, a common issue in high-density areas. However, the model occasionally fails to delineate closely packed nuclei accurately, as it tends to either slightly widen or compress the cell nucleus in its predictions. Furthermore, the images provide an insight into the model's limitations in differentiating between adjacent nuclei, where the predictions sometimes merge separate nuclei or exclude smaller ones. These challenges are expected, given the complexity of the task and the inherent variability in biological tissue samples. Nevertheless, the UNet++ model has shown its potential in managing the trade-off between sensitivity and specificity, with a tendency to preserve structural integrity over detecting every single nucleus, which can be a strategic choice depending on the application context.

6 Conclusion

During our project, we discovered that the appropriate loss function varies depending on the dataset type. The MoNuSeg 2018 challenge data, with its high background-to-target ratio, initially led to undesired results using cross-entropy loss, which favors the majority class, often at the expense of smaller but critical target areas like nuclei. To overcome this, we switched to a combined loss function of Dice Loss and Cross-entropy in later experiment stages, achieving the expected outcomes.

Furthermore, the UNet++ model struggles with differentiating adjacent nuclei, often merging them or missing smaller ones due to the complexity of biological tissues. Despite these challenges, it effectively balances sensitivity and specificity, prioritizing structural integrity. This strategic approach is suitable for applications where preserving structure is more crucial than detecting every nucleus.

7 Contribution

Xinru Song: Responsible for composing the introduction, preprocessing, and dataset sections of the final report. Participated in group meetings and discussed project content. Took charge of data preprocessing, authored the `Mask.ipynb` notebook.

Yixuan Deng: Participated in discussions. Responsible for writing the U-Net++ related works section. Collaborated the `train.py` and `dataset.py` files. Performed debugging over project.

Zhiran Wang: Participated in discussions, selected the project direction, and discussed task assignments. Responsible for writing the U-Net++ model section, the problem statement and conclusion. Coded the model from scratch, authored the `model.py` and `dataset.py` files, and performed debugging over project.

Xuhui Guo: Participated in discussions. Conducted literature research, made decision for final project topic. Carried on data preprocessing and wrote code for `color_transform.ipynb` and `sliding_augmentation.ipynb`. Trained the model and conducted evaluation. Authored the `train.py`, `inference.py`, and `test.py` files. Responsible for writing the implementation details and results sections of the final report.

References

[1] Vijay Badrinarayanan, Ankur Handa, and Roberto Cipolla. Segnet: A deep convolutional encoder-decoder architecture for robust semantic pixel-wise labelling. *arXiv preprint arXiv:1505.07293*, 2015.

[2] Liang-Chieh Chen, George Papandreou, Iasonas Kokkinos, Kevin Murphy, and Alan L. Yuille. Semantic image segmentation with deep convolutional nets and fully connected crfs. *arXiv preprint arXiv:1412.7062*, 2014.

[3] Liang-Chieh Chen, George Papandreou, Iasonas Kokkinos, Kevin Murphy, and Alan L. Yuille. Deeplab: Semantic image segmentation with deep convolutional nets, atrous convolution, and fully connected crfs. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 40(4):834–848, 2018.

[4] Qi Dou, Hao Chen, Yueming Jin, Lequan Yu, Jing Qin, and Pheng-Ann Heng. 3d deeply supervised network for automatic liver segmentation from ct volumes. In *International Conference on Medical Image Computing and Computer-Assisted Intervention*, pages 149–157. Springer, 2016.

[5] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Spatial pyramid pooling in deep convolutional networks for visual recognition. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 37(9):1904–1916, 2015.

[6] Jonathan Long, Evan Shelhamer, and Trevor Darrell. Fully convolutional networks for semantic segmentation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2015.

[7] Olaf Ronneberger, Philipp Fischer, and Thomas Brox. U-net: Convolutional networks for biomedical image segmentation. In Nassir Navab, Joachim Hornegger, William M. Wells, and Alejandro F. Frangi, editors, *Medical Image Computing and Computer-Assisted Intervention – MICCAI 2015*, pages 234–241, Cham, 2015. Springer International Publishing.

[8] Peter Shirley. Color transfer between images. *IEEE Corn*, 21(34-41):10–1109, 2001.

[9] Saeid Asgari Taghanaki, Yefeng Zheng, S Kevin Zhou, Bogdan Georgescu, Puneet Sharma, Daguang Xu, Dorin Comaniciu, and Ghassan Hamarneh. Combo loss: Handling input and output imbalance in multi-organ segmentation. *Computerized Medical Imaging and Graphics*, 75:24–33, 2019.

[10] Zongwei Zhou, Md Mahfuzur Rahman Siddiquee, Nima Tajbakhsh, and Jianming Liang. Unet++: Redesigning skip connections to exploit multiscale features in image segmentation. *IEEE transactions on medical imaging*, 39(6):1856–1867, 2019.