REPUBLIC OF TÜRKİYE
YILDIZ TECHNICAL UNIVERSITY
GRADUATE SCHOOL OF SCIENCE AND ENGINEERING

NUCLEI INSTANCE SEGMENTATION IN HISTOPATHOLOGY IMAGES WITH DEEP LEARNING

Serdar YILDIZ

MASTER OF SCIENCE THESIS
Department of Computer Engineering
Program of Computer Engineering

Supervisor
Prof. Dr. Songül VARLI

Co-supervisor
Assist. Prof. Dr. Abbas MEMİŞ

July, 2024

REPUBLIC OF TÜRKİYE
YILDIZ TECHNICAL UNIVERSITY
GRADUATE SCHOOL OF SCIENCE AND ENGINEERING

NUCLEI INSTANCE SEGMENTATION IN HISTOPATHOLOGY IMAGES WITH DEEP LEARNING

A thesis submitted by Serdar YILDIZ in partial fulfillment of the requirements for the degree of **MASTER OF SCIENCE** is approved by the committee on 22.07.2024 in Department of Computer Engineering, Program of Computer Engineering.

Prof. Dr. Songül VARLI
Yildiz Technical University
Supervisor

Assist. Prof. Dr. Abbas MEMİŞ
İstanbul University
Co-supervisor

Approved By the Examining Committee

Prof. Dr. Songül VARLI, Supervisor
Yildiz Technical University ______________________

Prof. Dr. Mine Elif KARSLIGİL, Member
Yildiz Technical University ______________________

Prof. Dr. Çiğdem GÜNDÜZ DEMİR, Member
Koc University ______________________

I hereby declare that I have obtained the required legal permissions during data collection and exploitation procedures, that I have made the in-text citations and cited the references properly, that I haven't falsified and/or fabricated research data and results of the study and that I have abided by the principles of the scientific research and ethics during my Thesis Study under the title of Nuclei Instance Segmentation in Histopathology Images with Deep Learning supervised by my supervisor, Prof. Dr. Songül VARLI. In the case of a discovery of false statement, I am to acknowledge any legal consequence.

Serdar YILDIZ

Signature

Dedicated to my family

ACKNOWLEDGEMENTS

This thesis would not have been possible without the guidance and support of my supervisors, Prof. Dr. Songül VARLI and Assist. Prof. Dr. Abbas MEMİŞ. Their insights and expertise have been invaluable throughout this research journey. I am deeply grateful for their patience, encouragement, and immense knowledge. Their willingness to give their time so generously has been very much appreciated. I would also like to express my gratitude to the Examining Committee for their insightful comments and suggestions.

Serdar YILDIZ

TABLE OF CONTENTS

LIST OF SYMBOLS

AJI	Aggregated Jaccard Index
AJI^{+}	Aggregated Jaccard Index Plus
DQ	Detection Quality
IoU	Intersection over Union
PQ	Panoptic Quality
PQ^{+}	Panoptic Quality Plus
SQ	Segmentation Quality

LIST OF ABBREVIATIONS

CE	Cross-entropy
COCO	Common Objects in Context
CoNIC	The Colon Nuclei Identification and Counting
CoNSeP	The Colorectal Nuclear Segmentation and Phenotype
CPM	The Computational Precision Medicine
FT	Focal-Tversky
LCD	Log-Cosh Dice
MLP	Multilayer Perceptron
MoNuSeg	The Multi-organ Nucleus Segmentation Challenge
MSE	Mean Square Error
R-CNN	Region Based Convolutional Neural Networks
ReLU	Rectified Linear Unit
RoI	Region of Interest
RPN	Region Proposal Network
SGD	Stochastic Gradient Descent
SiLU	Sigmoid Linear Unit
ViT	Vision Transformer
WSI	Whole Slide Image
YOLO	You Only Look Once

LIST OF FIGURES

LIST OF TABLES

ABSTRACT

Nuclei Instance Segmentation in Histopathology Images with Deep Learning

Serdar YILDIZ

Department of Computer Engineering
Master of Science Thesis

Supervisor: Prof. Dr. Songül VARLI
Co-supervisor: Assist. Prof. Dr. Abbas MEMİŞ

Nuclear structures play a crucial role in disease diagnosis, as abnormalities in the shape, size, and distribution of nuclei serve as key indicators in the meticulous examination of cancerous tissues. Therefore, the task of nuclei instance segmentation is essential since it allows comprehensive and detailed morphological assessments of individual nuclei. By improving segmentation techniques, we can facilitate quicker and more accurate diagnoses, which are absolutely essential for effective treatment planning and better health care solutions. This thesis rigorously examines nuclei instance segmentation in histopathology images by developing and evaluating various models. It explores UNet-based semantic segmentation, general-purpose instance segmentation models, and medical field specific models like HoVer-Net and StarDist. We also introduce novel approaches, such as HR-YOLO and YOLO-U, and provide a thorough analysis using cross-domain tests with MoNuSeg, CMP17, CoNIC, and CoNSeP datasets. Our proposed nuclei instance segmentation model, HR-YOLO, is an enhanced version of the YOLO architecture that optimizes small-scale object detection. The other proposed model, YOLO-U, is an ensemble technique that combines UNet semantic segmentation capabilities with the detection capabilities of YOLOv7. This study establishes a standardized experimental setup to enable fair comparison across studies, contributing significantly to the field by setting a benchmark for future research on nuclei instance segmentation.

Keywords: Nuclei segmentation, deep learning, instance segmentation, object

detection

ÖZET

Derin Öğrenme ile Histopatoloji Görüntülerinde Çekirdek Örnek Bölütleme

Serdar YILDIZ

Bilgisayar Mühendisliği Anabilim Dalı
Yüksek Lisans Tezi

Danışman: Prof. Dr. Songül VARLI
Eş-Danışman: Dr. Öğr. Üyesi Abbas MEMİŞ

Çekirdeklerin şekli, boyutu ve dağılımındaki anormallikler, kanserli dokuların titizlikle incelenmesinde temel göstergeler olarak hizmet ettiğinden, çekirdek yapıları hastalık teşhisinde çok önemli bir rol oynar. Bu nedenle, çekirdek örnek bölütleme görevi, her bir çekirdekğin kapsamlı ve ayrıntılı morfolojik değerlendirmelerine izin verdiği için çok önemlidir. Segmentasyon tekniklerini geliştirerek, etkili tedavi planlaması ve daha iyi sağlık hizmetleri çözümleri için kesinlikle gerekli olan daha hızlı ve daha doğru teşhisleri kolaylaştırabiliriz. Bu tez, çeşitli modeller geliştirip ölçümleyerek histopatoloji görüntülerinde çekirdek örneği segmentasyonunu titizlikle incelemektedir. UNet tabanlı semantik segmentasyonu, genel amaçlı örnek segmentasyon modellerini ve HoVer-Net ve StarDist gibi medikal alana özgü modelleri araştırmaktadır. Ayrıca HR-YOLO ve YOLO-U gibi yeni yaklaşımlar sunuyoruz ve MoNuSeg, CMP17, CoNIC ve CoNSeP veri kümeleriyle alanlar arası testleri kullanarak kapsamlı bir analiz sağlıyoruz. Önerilen çekirdek örnek bölütleme modelimiz HR-YOLO, YOLO mimarisinin küçük ölçekli nesne algılamayı optimize eden geliştirilmiş bir versiyonudur. Önerilen diğer model olan YOLO-U, UNet anlamsal bölümleme yeteneklerini YOLOv7'nin tespit yetenekleriyle birleştiren bir topluluk tekniğidir. Bu çalışma, çalışmalar arasında adil karşılaştırmayı mümkün kılmak için standartlaştırılmış bir deney ortamı oluşturmakta ve çekirdek örnek bölütlemesi konusunda gelecekteki araştırmalar için bir referans noktası oluşturarak alana önemli ölçüde katkıda bulunmaktadır.

Anahtar Kelimeler: Hücre bölütlemesi, derin öğrenme, örnek bölütleme, obje tespit

1
INTRODUCTION

Histology is a branch of biology that examines the microscopic structure of tissues and cells [1]. Pathologists in the field of histology scrutinize various tissues under the microscope to diagnose diseases. Nuclear structures play a crucial role in disease diagnosis [2]. Particularly, abnormalities in the shape, size, and distribution of nuclei serve as key indicators in the examination of cancerous tissues [3].

It is a well-established fact that early diagnosis saves lives. However, the manual examination of tissue samples under a microscope is a challenging and time-consuming task. Additionally, the inherent difficulty of the task increases the likelihood of errors [4]. Prolonged tissue analysis can lead to treatment delays for potentially life-threatening diseases, which might be preventable with an earlier diagnosis.

Another significant issue is a lack of expert pathologists. Pathological expertise requires extensive training and time. The scarcity of specialists contributes to an increased workload for existing professionals, thereby prolonging the diagnostic timeline for patients. Even in developed countries, the number of expert pathologists is insufficient, and in developing countries, it is markedly lower [5]. This shortfall inevitably leads to prolonged diagnostic times, which can increase mortality rates.

Nuclei instance segmentation presents several challenges [6]. One primary issue is the small size of the target objects, as nuclei can be as small as 20×20 pixels. Many general-purpose instance segmentation models may fail to detect and segment such small objects effectively. Additionally, nuclei are frequently situated in close proximity to one another, complicating the task further. This spatial arrangement requires models not only to perform accurate segmentation but also to determine which pixels belong to which nucleus. This task becomes extremely difficult when closely spaced nuclei belong to the same class, necessitating sophisticated methods to successfully distinguish between them.

Another significant challenge in nuclei instance segmentation is the variability across scanners and tissue samples. The properties of the scanner can lead to significant differences between images of the same organ, even if they are stained identically [7]. Additionally, tissues from different organs often exhibit distinct nuclei distributions. These factors cause models to perform well on the dataset they were trained on but significantly underperform on other datasets.

Our motivation is to automate the process of nuclei instance segmentation in scanned tissue images, which has traditionally been performed by pathologists who spend a significant amount of time under the microscope. This automation not only aims to reduce the time required for diagnosis but also minimizes the margin of error, enabling the production of standardized results. Furthermore, this automation allows for the assessment of tissue images without the direct involvement of expert pathologists, which is especially useful in settings where specialists are scarce.

Additionally, the literature on nuclei instance segmentation lacks a standardized experimental setup for benchmarking. Many studies share results using different combinations of datasets, but comparing models trained on varied datasets can be unfair, especially in nuclei instance segmentation, where even the scanner model can significantly impact performance. To address this, our study ensures a consistent training and testing methodology across all experiments, aiming to establish a reliable benchmark for comparing models.

To address the challenges in nuclei instance segmentation, our contributions are as follows:

- Developed the HR-YOLO model, an enhanced version of YOLO optimized for nuclei instance segmentation.
- Introduced the YOLO-U model, a confidence threshold-free methodology that combines the capabilities of UNet and YOLOv7.
- Proposed an adaptive thresholding technique to improve precision in semantic segmentation-based models.
- Established a standardized benchmark for evaluating nuclei instance segmentation models, facilitating fair comparisons across studies.

This thesis is organized as follows: Chapter 2 reviews the datasets and metrics used in this study. In Chapter 3, we explore nuclei instance segmentation using UNet-based semantic segmentation models. Chapter 4 evaluates the performance

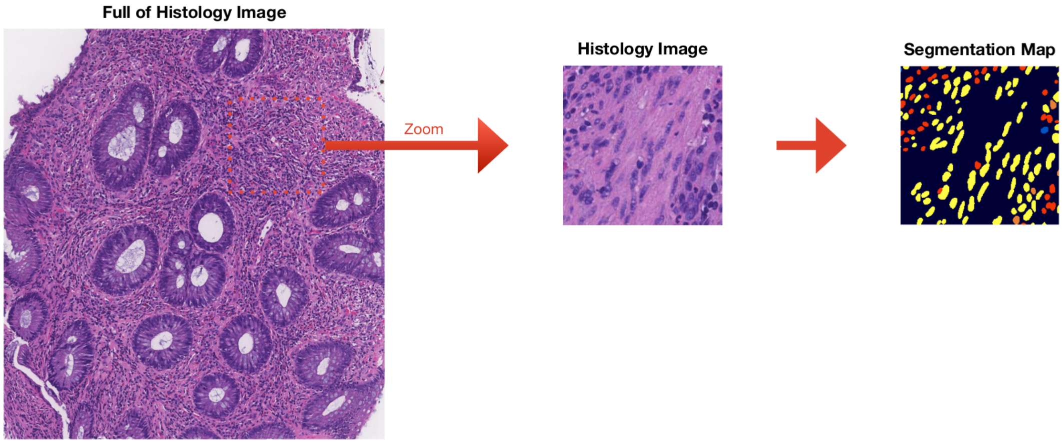


Figure 1.1 Patching process

of general-purpose instance segmentation models on nuclei segmentation tasks. Chapter 5 introduces the YOLO-U model. Chapter 6 reviews the most effective instance segmentation models in the medical field. Chapter 7 presents the HR-YOLO model, an adaptation of the YOLO architecture for high-resolution instance segmentation. Chapter 8 discusses the experimental results and provides a comparative analysis of the different models. The thesis concludes with Chapter 9. Results within each chapter are assessed independently, with extensive comparative evaluations reserved for Chapter 8, where the performances of all models are thoroughly analyzed.

1.1 Problem Definition

In computer vision, the task of object detection involves identifying and bounding target objects within an image. Semantic segmentation, in contrast, assigns a class to each pixel of an image. The instance segmentation task can be considered a combination of both semantic segmentation and object detection. It aims to determine the position of the target object and identify its boundaries.

Nuclei instance segmentation involves identifying individual nuclei and defining borders. This process allows for the measurement of morphological characteristics such as nucleus size and shape. It also makes it possible to measure the density and count of nuclei in a specified area, which can be accomplished through the nuclei instance segmentation task [6, 8].

Performing instance segmentation directly on whole slide images (WSIs) is not scalable due to the large input size. As illustrated in Figure 1.1, training and testing processes are typically conducted on patches extracted from WSIs. However, to maintain realism and practical relevance, it is crucial to generate test results that reflect the entire WSI. Thus, model outputs derived from individual patches must be combined to form a result for the WSI. The combination process must also address issues arising from objects located at the borders of patches, which is another challenge that needs a solution.

1.2 Literature Review

In the field of nuclei instance segmentation, there is no standard experimental setup, complicating direct comparisons between studies. Some research utilizes custom datasets for training and testing on established datasets [9], while others combine well-known datasets for both training and testing [10–12]. The performance of nuclei instance segmentation models heavily depends on factors such as the tissue source and the scanner used, making a fair comparison challenging. Therefore, this section focuses on the methodologies and datasets used in various models, deliberately omitting performance results to prevent biased evaluations. This study aims to include well-known models under consistent conditions to facilitate a fair comparison of performance.

In the literature, nuclei instance segmentation models can be categorized into three main categories. The first category includes models based on semantic segmentation, which are among the most commonly used approaches [13–15]. These models initially aim to create an accurate semantic segmentation map. This map is then utilized to perform instance segmentation, typically using the watershed algorithm [16]. While these models are effective at creating precise segmentation maps, they often struggle to separate closely located nuclei.

The second approach involves using multiple heads to generate several segmentation maps [17–20]. In addition to semantic segmentation, these models predict a separate segmentation map that identifies the borders of nuclei. It is suggested that this method can more effectively separate closely located nuclei. However, this methodology also requires multiple post-processing steps and has numerous parameters that need fine-tuning, which can complicate the model's implementation and optimization.

The final approach in nuclei instance segmentation directly predicts the segmentation results [21, 22]. This method represents nuclei using bounding box like annotations, similar to techniques used in object detection, to predict both the location and shape of each nucleus. Since it can produce instance segmentation results in a single step, it requires minimal post-processing and reduces the number of hyperparameters required. Although this detection-based methodology can more effectively distinguish closely located nuclei, it exhibits lower performance in representing the boundaries of the nuclei compared to semantic segmentation based models.

In Table 1.1, studies in the literature are listed chronologically by their year of publication. Due to the transition from semantic to instance segmentation in many studies, works focused on nuclei semantic segmentation are also included. Additionally, a separate column lists the datasets utilized in these studies. From Table 1.1, it can be inferred that the most commonly used models in the literature employ the The Multi-organ Nucleus Segmentation Challenge (MoNuSeg) [23], The Computational Precision Medicine (CPM) 2017 [24], and The Colorectal Nuclear Segmentation and Phenotypes (CoNSeP) datasets [20]. Consequently, this thesis utilizes these datasets to establish a baseline for assessing nuclei instance segmentation tasks.

Table 1.1 Literature review

Year	Study	Dataset	Methodology
2018	Zhang et al. [25]	CPM17	It introduces a dual contour-enhanced GAN using contour-highlighted masks and distance-transformed with a U-Net architecture.
2019	FullNet [26]	GlaS, MoNuSeg	It introduces a full-resolution convolutional neural network and maintains full-resolution feature maps to preserve edge details.
2019	Nuclei R-CNN [27]	DSB2018	It proposes an adaptation of Mask R-CNN, featuring a two-stage process with a block-merging algorithm and data preprocessing.
2020	Cheng et al. [28]	DSB2018, MoNuSeg	It presents a method using a feature pyramid network-based fusion module for enhanced detection and segmentation of nuclei.
2020	Qu et al. [29]	DSB2018	It introduces an integration CNN that combines SSD with FPN for bounding box detection and U-Net for segmentation.
2020	Wang et al. [30]	MoNuSeg	It introduces a network with a bending loss to penalize high curvature in contours, effectively separating overlapping nuclei in histopathology images.

Table 1.1 Literature review (continued)

2020	Hu et al. [31]	MoNuSeg	It presents a weakly supervised method using generative adversarial training to generate nuclear centroids likelihood maps.
2021	MDC-net [32]	DATA ORGANS, DATA BREAST	It leverages distance maps and contour information to segment touching nuclei effectively.
2021	Bancher et al. [33]	MoNuSeg, PanNuke	It proposes enhancements to Mask R-CNN by adding distance maps and hematoxylin-stain intensities as input channels.
2021	Rahmon et al. [34]	MoNuSeg	It introduces an extension of UNet that incorporates squeeze and excitation blocks and combines binary cross-entropy loss with dice loss.
2021	CDNet et al. [35]	MoNuSeg, CPM17	It introduces a model that uses centripetal direction features and a direction difference map to enhance boundary identification and handle overlapping nuclei.
2022	Haq et al. [36]	TNBC, MoNuSeg	It proposes a region-level SSL methodology with a novel triplet loss to create a pretrained vision transformer model.
2022	CircleSnake [37]	Custom Dataset	It introduces a circle representation-based segmentation model that has been developed for ball-shaped objects.
2022	Nguyen et al. [38]	MoNuSeg	It presents an anchor-free detection framework based on a circle representation, with heatmaps for center location and radius prediction.
2022	Ciscnet [39]	Lizard, CoNIC	It introduces a single-branch CNN using a V-Net architecture, incorporating normalized distance maps and watershed post-processing.
2022	Mahbod et al. [40]	CryoNuSeg, NuInsSeg, MoNuSAC	It proposes a dual-decoder UNet model that uses an encoder and two decoders to produce foreground and distance maps.

Table 1.1 Literature review (continued)

2022	HUNIS [41]	MoNuSeg	It introduces a two-stage unsupervised method that starts with adaptive thresholding and size priors for initial mask generation, followed by self-supervised refinement using these masks as pseudo-labels.
2022	Weigert et al. [21]	CoNIC	It extends StarDist to the CoNIC dataset, incorporating enhancements like geometric and color augmentations.
2022	Chanchal et al. [42]	Kidney, TNBC, MoNuSeg	It introduces the deep-structured residual encoder-decoder network, utilizing a wide path and residual connections.
2022	Liu et al. [43]	MultiOrgan, TNBC	It introduces a weakly supervised method using a dual-branch network combining an instance encoding network and a semantic proposal network, utilizing a discriminative loss.
2023	BAWGNet [44]	DSB2018, MoNuSeg, TNBC	It uses the discrete wavelet transform in a U-net-based architecture, which helps to improve the identification of closely located nuclei.
2023	Martos et al. [9]	Custom Dataset, MoNuSeg	Techniques such as advanced image preprocessing, including blurred artifact removal and stain normalization, are utilized to enhance segmentation accuracy.
2023	Rettenberger et al. [45]	Synthetic Custom Dataset	It evaluates Mask R-CNN and UNet on their ability to segment overlapping cells.
2023	PROnet [10]	CPM17, MoNuSeg	It proposes a weakly supervised nuclei segmentation method that employs geodesic distance-based Voronoi diagrams and k-means clustering.
2023	DARC [11]	CoNSeP, CPM17	It introduces a methodology that employs a recoloring technique and uses a novel instance normalization approach.

Table 1.1 Literature review (continued)

2023	Chen et al. [46]	MoNuSeg	It proposes an intra-image self-similarity methodology, which is a label-free approach for unsupervised nuclei segmentation, using a UNet trained on the pseudo-labels.
2023	Chen et al. [47]	MoNuSeg, BCData	It introduces an unsupervised method using Prior Self-activation Maps for pseudo mask generation through self-supervised learning and semantic clustering.
2023	Ahmad et al. [48]	PanNuke, CPM17, MoNuSeg	It presents a novel framework that enhances detection capabilities with attention mechanisms and multiple decoder heads.
2023	SA2-Net [49]	MoNuSeg, SegPC-2021, ISIC-2018	It presents a scale-aware attention network with novel scale-aware and adaptive up-attention modules by focusing on scale variations and local context.
2023	Parulekar et al. [50]	MoNuSAC, PanNuke, CoNSeP	It proposes a method for training deep neural networks on multiple datasets using a coarse-to-fine class hierarchy to adjust loss functions, enhancing generalization.
2023	FusionU-Net [51]	MoNuSeg, PanNuke	It incorporates a novel fusion module in a U-Net architecture to reduce semantic gaps and enhance bi-directional information exchange across skip connections.
2023	Silva et al. [52]	CryoNuSeg, MoNuSeg	It introduces an ensemble approach using FPN, Mask R-CNN, U-Net, and PSPNet for nuclei segmentation.
2023	ConDANet [53]	DSB2018, MoNuSeg, TNBC	It presents a contourlet-driven attention network that enhances segmentation by capturing fine edge details with wavelet pooling, as well as a contourlet transform-based attention mechanism.
2023	CPP-net [54]	DSB2018, BBBC006v1, BBBC039, PanNuke	It introduces a context-aware polygon proposal network that uses a confidence-based weighting module and shape-aware perceptual loss.

Table 1.1 Literature review (continued)

2023	TopoSeg [55]	MoNuSeg, CPM17, PBNulei	It presents a topology-aware method that integrates topological constraints into the loss function.
2023	cyto-Knet [56]	SegPC2021	It presents a hybrid model that combines heuristic techniques with deep learning, employing watershed postprocessing, conditional kernels, and color balancing.
2023	Rashid et al. [57]	CoNSeP, PanNuke, Lizard	It introduces an encoder decoder based CNN that predicts centroid maps and nuclei probabilities.
2023	Senapati et al. [58]	JPI 2016, IEEE TMI 2019, PSB 2015	It presents sharp dense UNet, an enhancement over dense UNet with sharp blocks.
2023	Li et al. [59]	CPM15, CPM17, TNBC	It introduces a dual-path network that learns nuclei and contour features independently, integrating them through fusion and attention modules.
2023	Zhao et al. [60]	MoNuSeg, CoNSeP, TNBC	It proposes an unsupervised model using a siamese network with contrastive learning, employing a convolution and attention network.
2024	Song et al. [61]	DSB2018, MoNuSeg	It suggests a way to improve segmentation by combining external sparse representation-based multi-head self-attention with a latent entropy-quantified channel attention mechanism.
2024	SPPNet [62]	MoNuSeg	It employs a single-point prompt strategy in a lightweight vision transformer for efficient nuclei segmentation, reducing computational costs while maintaining performance.
2024	Mahbod et al. [12]	MoNuSeg, TNBC, CryoNuSeg, CPM15, CPM17, CoNSeP	It introduces a hybrid approach that enhances nuclei segmentation by combining non-deterministic train time and deterministic test time stain normalization, utilizing DDU-Net with novel training techniques and test time augmentations.

Table 1.1 Literature review (continued)

2024	Song et al. [63]	Kather, MoNuSeg, Lizard, Crag, PanNuke	It presents a self-supervised methodology for unpaired image translation using CycleGAN, which incorporates a co-modulated GAN.

2
MATERIALS AND METRICS

As discussed in Chapter 1, there is no universally accepted training and testing protocol in the nuclei-instance segmentation literature, which complicates comparisons between studies. To establish a standardized training and testing protocol within this study, widely used datasets have been selected for training and testing. The Multi-organ Nucleus Segmentation Challenge (MoNuSeg) dataset [23] has been utilized for model training, with patches derived from the dataset. Validation was performed using the test subset of the MoNuSeg dataset. Cross-domain tests were conducted using the Computational Precision Medicine 2017 [24] (CPM17) and the Colorectal Nuclear Segmentation and Phenotypes datasets [20] (CoNSeP). All tests were conducted on region of interest (RoI) to maintain consistency.

Performance metrics have been categorized into two main groups: instance segmentation and semantic segmentation. This chapter reports on all commonly used metrics in nuclei segmentation literature, ensuring comprehensive coverage of evaluation criteria.

2.1 Nuclei Instance Segmentaion Datasets

2.1.1 Multi-organ Nucleus Segmentation Challenge

The Multi-organ Nucleus Segmentation Challenge (MoNuSeg) [23] is a significant dataset in digital pathology, designed specifically to assess model performance in nuclei instance segmentation tasks. We chose it as the primary training dataset for this study due to its widespread recognition as one of the most authoritative datasets in the literature. The dataset comprises whole slide images stained with hematoxylin and eosin, scanned at 40x magnification, sourced from the Cancer Genome Atlas database. The training and testing sets have been predefined as part of the challenge.

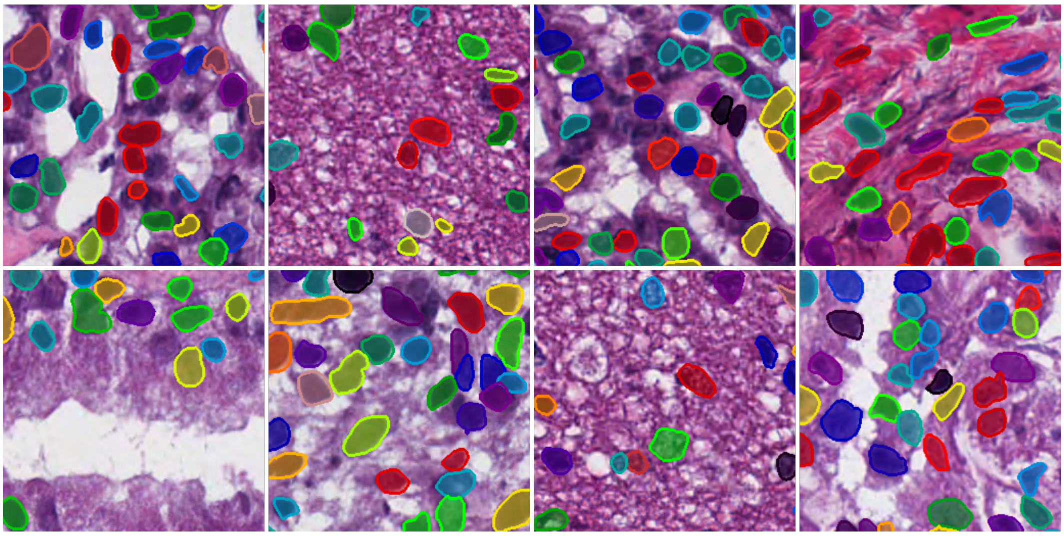


Figure 2.1 MoNuSeg example patches

The MoNuSeg dataset encompasses tissues from nine different organs: kidney, breast, lung, liver, colon, brain, bladder, stomach, and prostate. Nuclei density varies significantly across these tissue types. Figure 2.1 presents sample patches of 256×256 pixels for each type of tissue.

The training subset of the MoNuSeg dataset consists of 37 RoI, and the test subset includes 14 RoI. Each RoI comprises regions of 1000×1000 pixels. Training patches of 256×256 pixels were extracted to conform to common model scale requirements and ensure consistency across methods. These patches are created with a 50% overlap, also serving as a form of data augmentation. This process resulted in a total of 1813 input images for the training set.

2.1.2 The Colorectal Nuclear Segmentation and Phenotypes

The Colorectal Nuclear Segmentation and Phenotypes (CoNSeP) dataset [20] is another widely used resource in the literature. This dataset exclusively includes images of colon tissues. It comprises images derived from 16 colorectal adenocarcinoma RoI. The dataset consists of 41 images, each sized at 1000×1000 pixels, with 27 designated for training and 14 reserved for testing.

The CoNSeP dataset originates from the Department of Pathology at University Hospitals Coventry and Warwickshire, UK. The tissues were scanned at 40x magnification using an Omnyx VL120 scanner. Sample images of size 256×256 are presented in Figure 2.2.

In this study, only the 14 image test subset of the CoNSeP dataset was utilized. The evaluations were conducted on the original 1000×1000 pixel whole slide images,

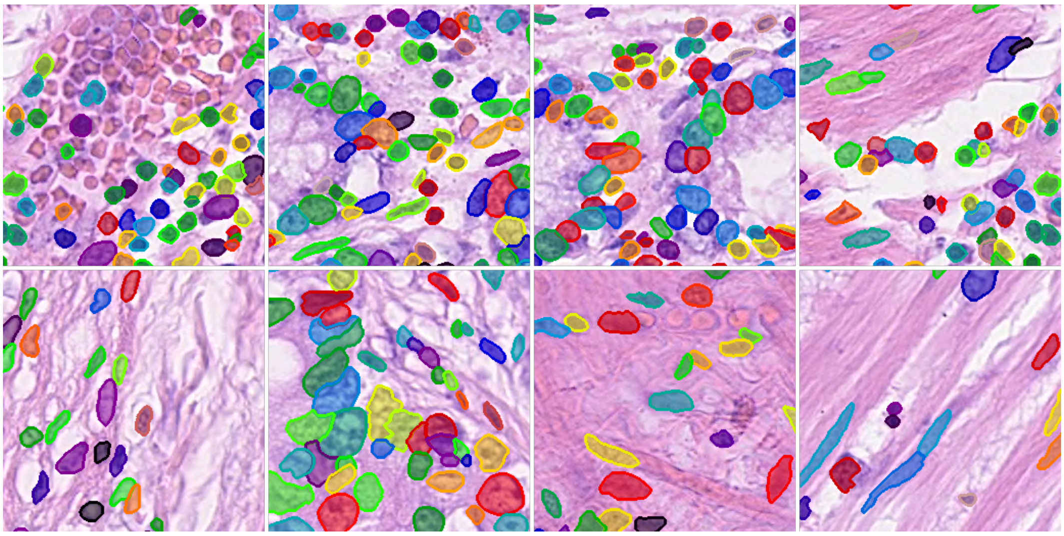


Figure 2.2 CoNSeP example patches

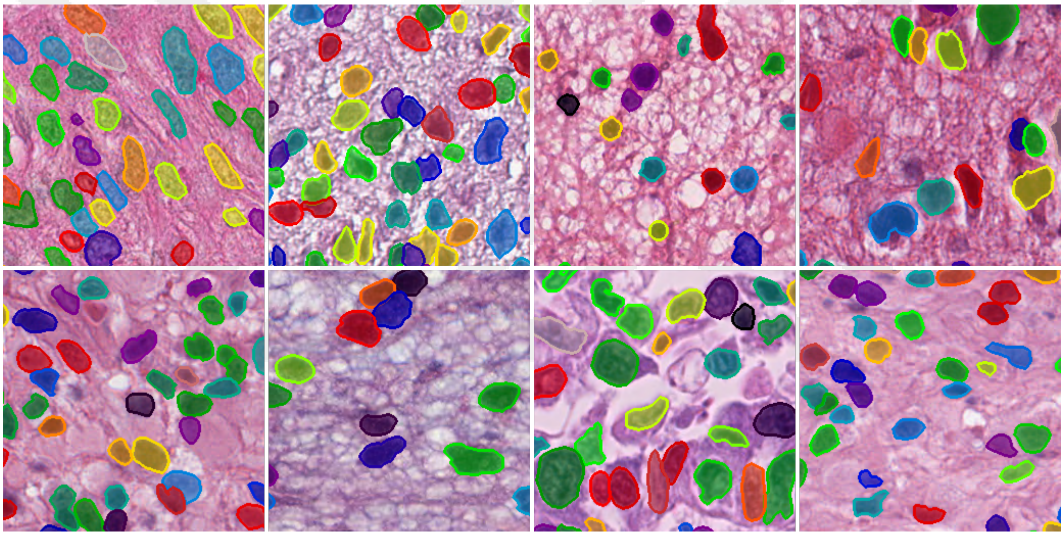


Figure 2.3 CPM17 example patches

without any patching process.

2.1.3 The Computational Precision Medicine 2017

The Computational Precision Medicine (CPM) nuclei segmentation challenge dataset [24], part of MICCAI 2017, is another widely used dataset for nuclei instance segmentation in the literature. Each nucleus within the dataset has been manually annotated. The CPM17 dataset includes image tiles extracted from whole slide tissue images of various cancers, including head and neck squamous cell carcinoma, glioblastoma multiforme, non-small cell lung cancer, and lower grade glioma.

Similar to the MoNuSeg dataset, the CPM17 originates from the Cancer Genome

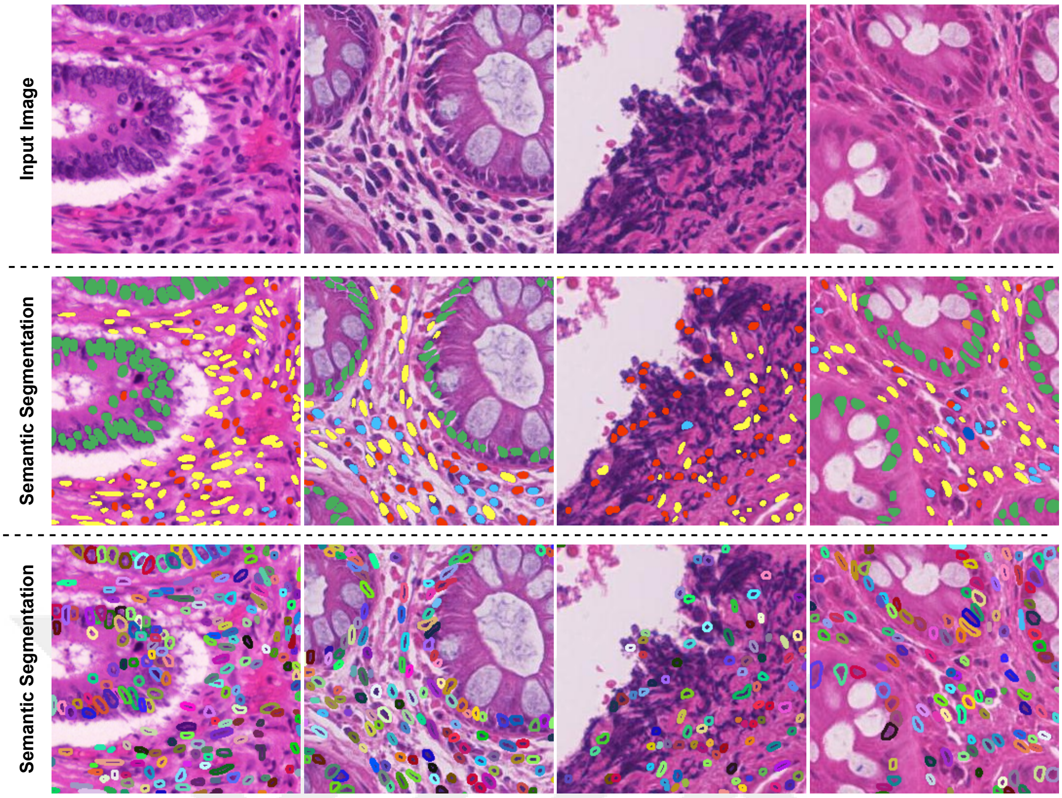


Figure 2.4 CoNIC example patches

Atlas database. The dataset comprises a total of 64 image tiles, evenly split between training and testing. These tiles are in the sizes of 500×500 and 600×600 pixels.

This study used the CPM17 dataset, similar to the CoNSeP dataset, for cross-domain testing. The performance of the models was evaluated using 32 test images of varying sizes. Since it was not included in the training dataset and contains different tissue types similar to the MoNuSeg dataset, all sourced from TCGA, it can be considered the primary test set.

2.1.4 Colon Nuclei Identification and Counting Challenge 2022

The Colon Nuclei Identification and Counting (CoNIC) dataset [64], although not widely used in the literature, has been included in this study due to its representation of six different nuclei classes, providing a basis to measure model classification performances. The CoNIC dataset is known for its highly imbalanced sample distribution among classes, making it one of the most challenging datasets to achieve high performance in the literature. It consists of a total of 4,981 RGB images, each with a size of 256×256. Example segmentation images from this dataset are shown in Figure 2.4.

The models were evaluated using a 5-fold cross-validation on the CoNIC dataset.

This process involves using each 20% segment of the dataset for testing, randomly selecting another 20% for validation, and utilizing the remaining data for training. This method aims to achieve a more realistic and distribution-independent assessment of model performance. Although this measurement strategy is accurate, it is costly in terms of computational resources, as it requires five separate trainings for each model evaluation.

2.2 Metrics

This section examines the semantic segmentation and instance segmentation metrics employed in this study. Although the primary aim is not to analyze the semantic segmentation performance of the models, these metrics play a crucial role in analyzing the reasons behind the varying levels of success in instance segmentation. Consequently, commonly used semantic segmentation metrics in the literature, such as dice, intersection over union, precision, and recall, have been employed. For instance segmentation, metrics such as aggregated Jaccard index, aggregated Jaccard index plus, panoptic quality, and panoptic quality plus have been utilized.

2.2.1 Semantic Segmentation Metrics

The Dice metric calculates the overlap ratio between the predicted pixels and the ground truth pixels, producing values ranging from 1 to 0. The formula for the Dice metric is provided in Equation 2.1.

$$\text{Dice} = \frac{2|P \cap G|}{|P| + |G|} \tag{2.1}$$

where P and G represent the sets of predicted and ground truth pixels, respectively. The Dice value is twice the ratio of the intersection of the prediction and ground truth segmentation maps to their summation of areas.

Intersection over Union (IoU) is another metric used for evaluating semantic segmentation. Like the Dice metric, it measures the overlap ratio between the ground truth and prediction maps. The IoU metric is defined by the following formula:

$$\text{IoU} = \frac{|P \cap G|}{|P \cup G|} \tag{2.2}$$

Unlike the Dice metric, IoU calculates the ratio of the intersection to the union of these segmentation maps, with values ranging between 0 and 1.

Precision is a metric that measures how precise the prediction map is. The formula for precision is given below:

$$\text{Precision} = \frac{|P \cap G|}{|P|} \tag{2.3}$$

This metric calculates the ratio of correctly predicted pixels to the total pixels in the predicted segmentation.

The recall metric is provided in Formula 2.4. It is calculated by taking the ratio of the intersection of the prediction and the ground truth maps to the ground truth map:

$$\text{Recall} = \frac{|P \cap G|}{|G|} \tag{2.4}$$

Recall is especially important in nuclei segmentation since there is commonly a significant imbalance between background and nuclei classes. Models that inaccurately predict nuclei pixels as background can significantly degrade instance segmentation performance. Thus, models demonstrating high recall for background but low recall for nuclei likely suffer from this predictive bias.

2.2.2 Instance Segmentation Metrics

Instance segmentation metrics differ from semantic segmentation metrics by requiring a match between each ground truth instance and its predicted counterpart. To achieve this, the IoU is calculated for all ground truth and predicted instances. Matching starts with the prediction that has the highest IoU overlap, and this process continues to align all predictions with their corresponding ground truth instances. Predictions that do not match any ground truth instance are classified as false positives.

One of the most widely used metrics for nuclei instance segmentation in the literature is the Aggregated Jaccard Index (AJI) [23]. The formula for the AJI metric is given in Equation 2.5. It is calculated by summing the intersection areas of the instances in the ground truth map with the highest overlapping predicted instances, and then dividing this sum by the total area of all ground truth and predicted instances.

$$\text{AJI} = \frac{\sum_{i=1}^{N} |G_i \cap P_{\text{best}(i)}|}{\sum_{i=1}^{N} |G_i \cup P_{\text{best}(i)}| + \sum_{k \in U_G} |G_k| + \sum_{l \in U_P} |P_l|} \tag{2.5}$$

The AJI, which is widely used in the literature to assess nuclei instance segmentation, is calculated as the ratio of the sum of the intersection areas of the best matching ground truth and predicted instances to the sum of all ground truth and predicted areas. AJI values range from 0 to 1. The enhanced metric, AJI^+, enhances this by optimizing the matching process to maximize the sum of intersections. This ensures that the value of AJI^+ is always greater than or equal to AJI.

Another commonly used metric in the literature, and a foundational one for this study, is the Panoptic Quality (PQ). As outlined in Equations 2.6 and 2.7, PQ is calculated by multiplying two separate metrics, similar to the F1 score. Detection Quality (DQ) measures how successfully the model detects nuclei, while Semantic Quality (SQ) evaluates how accurately the detected instances are segmented.

$$PQ_t = \underbrace{\frac{|TP_t|}{|TP_t| + \frac{1}{2}|FP_t| + \frac{1}{2}|FN_t|}}_{\text{detection quality (DQ)}} \times \underbrace{\frac{\sum_{(p_t,g_t)\in TP} IoU(p_t, g_t)}{|TP_t|}}_{\text{segmentation quality (SQ)}} \tag{2.6}$$

$$mPQ = \frac{1}{T}\sum_{t=1}^{T} PQ_t \tag{2.7}$$

In the assessment of panoptic quality, if the IoU between the predicted instance p_t and the ground truth instance g_t is greater than or equal to 0.5, it is considered a true positive (TP). All other predictions are classified as false positives or false negatives. The T denotes the number of classes. The PQ is computed for each class separately, and their average is taken to calculate the mean Panoptic Quality (mPQ).

The panoptic quality (PQ) metric is computed for each patch, and their average represents the dataset's overall performance. However, due to the varying density of nuclei across different patches, the standard deviation of PQ values between patches can be high. Therefore, the PQ^+ metric has been proposed in the literature [64]. Unlike the standard PQ metric, PQ^+ treats the entire dataset as a single image. It calculates the true positives (TP), false positives (FP), and false negatives (FN) across the entire dataset and then computes the PQ^+ metric.

3 NUCLEI INSTANCE SEGMENTATION WITH UNET-LIKE ARCHITECTURES

UNet architectures have a broad range of applications in the literature, from semantic segmentation to specific tasks like tissue segmentation [28, 65, 66]. The extensive use of UNet is not only due to its simple architectural design but also its ability to achieve high performance. This has made UNet a common choice in research on semantic and instance segmentation, especially in studies of nuclei [66].

While various studies report experimental results using different datasets and UNet architectures [21, 67, 68], the lack of a uniform experimental setup makes direct comparisons between models inappropriate. There is a significant gap in the literature regarding the comparison of UNet models for nuclei instance segmentation. To bridge this gap, we developed several UNet models under a consistent experimental setup using the MoNuSeg and CoNIC datasets. Additionally, we discussed factors affecting model performance and proposed methods for improvement. Furthermore, we used the simple watershed algorithm [16] to convert semantic segmentation predictions into instance segmentation, emphasizing that the architecture of the model is a significant factor influencing performance.

In this chapter, we explore nuclei instance segmentation using various UNet architectures. Initially, we employ the standard UNet model and strive to achieve the highest possible accuracy. We evaluate different loss functions and explore other transformer-based UNet architectures, such as TransUNet and Swin-UNet. Furthermore, we propose an adaptive thresholding technique aimed at enhancing the accuracy of the instance segmentation models and reducing model uncertainty. Lastly, we compare all the UNet architectures examined and assess the impact of our proposed thresholding methodology.

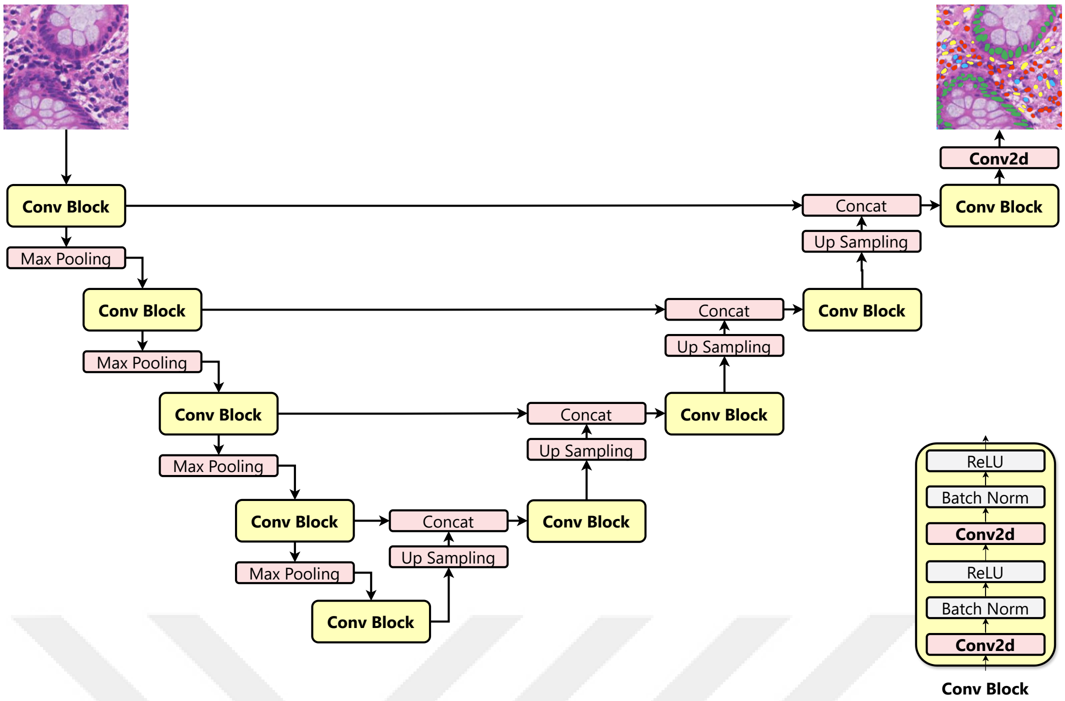


Figure 3.1 UNet architecture

3.1 Classical UNet Architecture

The classical UNet model was first introduced in 2015 by Ronneberger et al. for medical image segmentation [69]. This is a fully convolutional neural network that only uses convolutional filters to transform an input image into an output probability matrix. The original UNet architecture does not apply any padding to the input of the convolutional filters. Instead, cropping is used on the output image to achieve the desired dimensions before performing the concatenation process. The final output, a probability matrix, is then scaled to fit the input image's dimensions. In contemporary approaches, padding is applied to the input matrices to ensure that the dimensions of the output matrix remain unchanged, and the concatenation process is performed with the resulting feature vectors [70]. In this study, the modern approach is implemented, with the model that generates an output probability matrix the same size as the input image.

Figure 3.1 displays the architecture of the UNet model, which consists of two main components: the encoder and the decoder. The encoder is comprised of successive convolution blocks (Conv Block) and max pooling layers. However, the architecture's success is primarily due to interconnections between levels rather than this sequential structure. The first level can be termed as $P0$, which produces an output of the same dimensions as the input image. Subsequently, pooling is applied to reduce the dimension by half, achieving the $P1$ level. This process is repeated until the image size is reduced to 1/16 at the $P4$ level. As the model progresses from

$P0$ to $P4$, it transitions from fine to coarse feature vectors. Intuitively, while coarse feature vectors are indicative of the object classes, fine feature vectors delineate the object boundaries [69].

The decoder in the UNet architecture concatenates coarse and fine feature vectors obtained at each level, transferring both the class information and boundary details to the next higher level. At the $P0$ level, this generates a comprehensive feature vector that includes both of these aspects while remaining the same size as the original input image. This feature vector is then processed with a convolution filter with an equal number of channels as classes, producing a matrix representing the class probability distribution.

A Convolution Block (Conv Block) in the UNet architecture consists of successive 3×3 convolution layers, followed by batch normalization [71] and the ReLU (Rectified Linear Unit) [72] activation function. The convolution filters are applied to extract feature vectors from the image. Batch normalization then standardizes these feature vectors to a specific distribution, reducing vanishing or exploding gradient problems during training and allowing for the stable training of deeper models [73]. The ReLU activation function introduces non-linearity, allowing the model to learn more complex patterns in the data while avoiding the vanishing gradient problem by not constraining the output to a narrow range [72].

Let us define $UNet(.)$ as the function representing the UNet semantic segmentation model, where $Block^p(.)$ denotes the p^{th} convolution block, and $X \in R^{3\times h\times w}$ is an input image. The process of obtaining a semantic segmentation probability matrix for a given input image X is formalized as follows:

$$Block^p_{encoder}(x_p) = E_p \quad and \quad X = x_0 \quad where \quad p \in 0, 1, \ldots, n \tag{3.1}$$

$$MaxPool(E_p) = x_{p+1} \tag{3.2}$$

$$Block_{stem}(x_n) = E_n = D_n \tag{3.3}$$

$$UpSample(D_p) = U_p \tag{3.4}$$

$$Block^p_{decoder}([E_{p-1}, U_p]) = D_{p-1} \tag{3.5}$$

$$Conv(D_0) = Y \tag{3.6}$$

where n is the number of levels, and $Y \in R^{c\times h\times w}$ represents the semantic segmentation map, with c denoting the number of classes.

Since the model was developed for the semantic segmentation task, it is necessary to convert these results into instance segmentation predictions. To achieve this, the watershed algorithm has been utilized. For each class prediction mask, points furthest from the background are identified and assigned as nucleus centers. A distance map, which measures the proximity to the nearest background pixel, along with the center map and the semantic segmentation map, are provided to the watershed algorithm. This process yields the final instance segmentation output.

3.1.1 UNet Experimental Setup

The UNet model was trained with semantic segmentation labels from the CoNIC and MoNuSeg datasets. During training on the CoNIC dataset, the model also performs a classification task for 6 classes. As suggested in Section 2.1.4, 5-fold cross-validation was used to better measure the model's performance.

The MoNuSeg dataset was only used for training the UNet model for nuclei instance segmentation. In adherence to established practices in the literature, the same subset was used for both validation and testing. However, to achieve a more realistic assessment of the model's performance, cross-testing was conducted. The model trained with the MoNuSeg dataset was tested using the test subsets of the CPM17 and CoNSeP datasets. This approach not only facilitated a realistic measurement of performance but also enabled cross-domain testing.

The AdamW optimizer [74] was utilized, and the batch size was set to 16. The Cyclic Learning Rate Scheduler [75] was employed, with the maximum learning rate set at 0.0006. Cross entropy served as the loss function. In total, the training process involved 40,000 steps. The model was trained using input sizes of 256x256 patches without any resizing, although testing was conducted on RoI.

3.1.2 UNet Results

We evaluated the UNet model from two different perspectives. Initially, we assessed both instance segmentation and classification performance using the CoNIC dataset. In the second phase, we trained the model exclusively with the MoNuSeg dataset, which labels only nuclei, and subsequently performed in-domain and cross-domain tests using the CPM17 and CoNSeP datasets. Furthermore, we evaluated the model's semantic segmentation performance to investigate its relationship with instance segmentation performance.

The semantic segmentation results for the UNet model are presented in Table

Table 3.1 UNet semantic segmentation results on CoNIC dataset

	Fold	μ	Bg	Neu	Epi	Lym	Pla	Eos	Con
Dice	**1**	0.7471	0.9716	0.5315	0.8389	0.8089	0.6599	0.6292	0.7896
	2	0.7480	0.9706	0.5328	0.8416	0.8075	0.6624	0.6402	0.7811
	3	0.7456	0.9720	0.5226	0.8405	0.8101	0.6547	0.6321	0.7874
	4	0.7446	0.9714	0.5288	0.8359	0.7997	0.6649	0.6291	0.7824
	5	0.7470	0.9712	0.5439	0.8413	0.8059	0.6516	0.6357	0.7794
	μ	0.7465	0.9714	0.5319	0.8396	0.8064	0.6587	0.6333	0.7840
IoU	**1**	0.6160	0.9448	0.3620	0.7225	0.6792	0.4925	0.4590	0.6524
	2	0.6166	0.9429	0.3632	0.7265	0.6771	0.4952	0.4708	0.6409
	3	0.6147	0.9455	0.3537	0.7248	0.6808	0.4867	0.4621	0.6493
	4	0.6125	0.9444	0.3594	0.7181	0.6663	0.4980	0.4589	0.6426
	5	0.6152	0.9441	0.3735	0.7260	0.6749	0.4833	0.4660	0.6386
	μ	0.6150	0.9443	0.3624	0.7236	0.6757	0.4911	0.4634	0.6448
Recall	**1**	0.7328	0.9711	0.4725	0.8468	0.8186	0.6321	0.6056	0.7827
	2	0.7437	0.9689	0.4941	0.8575	0.8073	0.6567	0.6533	0.7683
	3	0.7385	0.9716	0.4679	0.8441	0.8063	0.6285	0.6564	0.7950
	4	0.7368	0.9706	0.4749	0.8456	0.8017	0.6588	0.6328	0.7732
	5	0.7449	0.9680	0.5265	0.8642	0.8074	0.6209	0.6417	0.7852
	μ	0.7393	0.9700	0.4872	0.8516	0.8083	0.6394	0.6380	0.7809
Precision	**1**	0.7646	0.9722	0.6074	0.8312	0.7995	0.6903	0.6548	0.7966
	2	0.7535	0.9723	0.5781	0.8263	0.8077	0.6682	0.6276	0.7945
	3	0.7554	0.9724	0.5917	0.8368	0.8139	0.6831	0.6095	0.7800
	4	0.7545	0.9721	0.5965	0.8265	0.7978	0.6712	0.6255	0.7919
	5	0.7500	0.9745	0.5624	0.8195	0.8044	0.6856	0.6298	0.7737
	μ	0.7556	0.9727	0.5872	0.8281	0.8047	0.6797	0.6294	0.7873

3.1. It is evident that the UNet model effectively segments the background class. However, considering that approximately 84% of the CoNIC dataset consists of background and only 16% is nuclei, achieving the highest performance metrics for background segmentation is inevitable. In contrast, the model also exhibits substantial effectiveness in segmenting nuclei. However, as seen in the examples in Figure 3.2, there are instances where a single nucleus contains pixels from two different classes. This issue significantly undermines the class-based semantic segmentation performance for nuclei.

Table 3.2 presents the 5-fold cross-validation instance segmentation results for the UNet model on the CoNIC dataset. The consistency of mPQ^{+} values across different folds suggests a balanced distribution of instances within the dataset folds. The highest class-based performance was achieved in the lymphocyte class with a PQ^{+} of 0.6134. Although high performance was anticipated for the epithelial class, which has the most examples, it recorded a surprisingly low PQ^{+} of 0.3859. As illustrated in Figure 3.3, despite the model's effective segmentation, its instance segmentation performance is compromised because it cannot separate

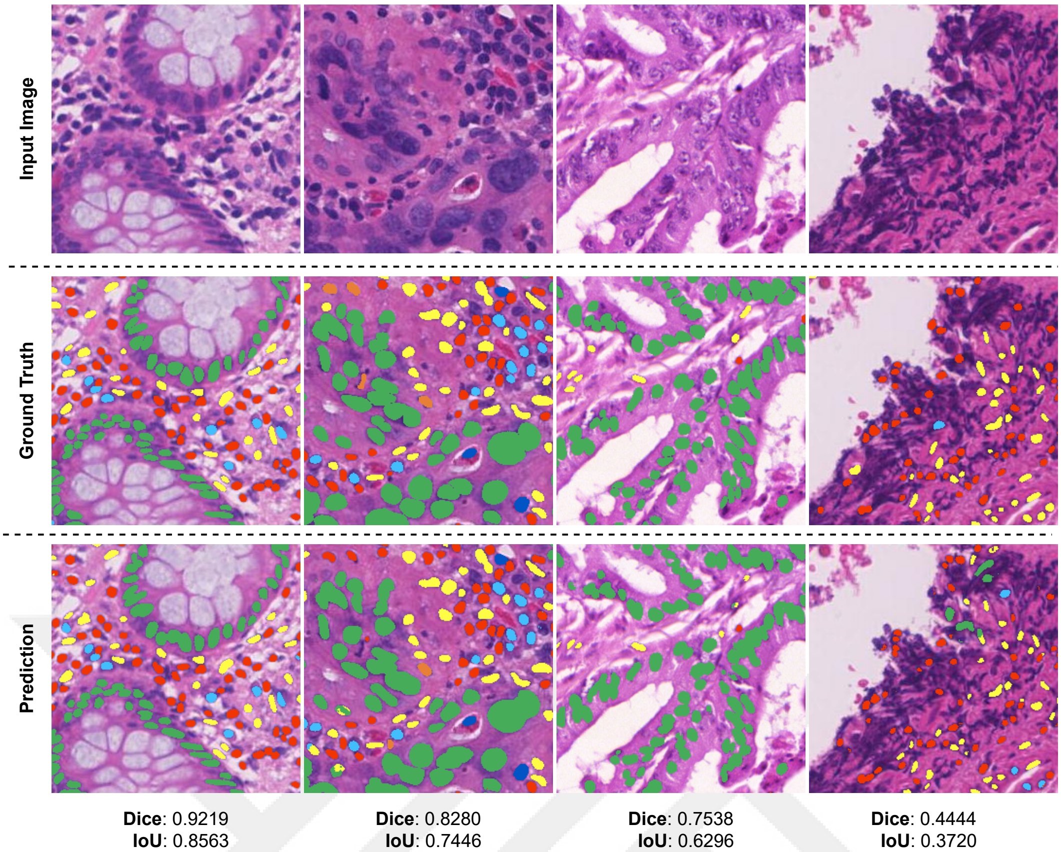


Figure 3.2 UNet semantic segmentation prediction on CoNIC patches

Table 3.2 UNet instance segmentation results on CoNIC dataset

Fold	mPQ$^+$	PQ$^+_{\text{Neu}}$	PQ$^+_{\text{Epi}}$	PQ$^+_{\text{Lym}}$	PQ$^+_{\text{Pla}}$	PQ$^+_{\text{Eos}}$	PQ$^+_{\text{Con}}$
1	0.4525	0.3019	0.3991	0.6052	0.4954	0.3555	0.5578
2	0.4518	0.2937	0.3706	0.6229	0.5062	0.3617	0.5556
3	0.4473	0.2786	0.3881	0.6136	0.4945	0.3542	0.5548
4	0.4520	0.2954	0.3955	0.6164	0.5014	0.3523	0.5511
5	0.4443	0.2888	0.3764	0.6088	0.4939	0.3518	0.5463
μ	0.4496	0.2917	0.3859	0.6134	0.4983	0.3551	0.5531

closely located nuclei. Additionally, the neutrophil class, which achieved the lowest performance with a PQ^+ of 0.2717, also displayed poor results in semantic segmentation. There is a clear correlation between the poor performance and the limited number of examples in the dataset for this class.

The in-domain and cross-domain semantic segmentation results of the UNet model trained with the MoNuSeg dataset are presented in Table 3.3. Both the in-domain MoNuSeg and the cross-domain CPM17 datasets yield closely matched results, demonstrating high effectiveness in segmenting the targeted classes. However, the performance on the CoNSeP dataset is comparatively lower. Despite achieving a recall of 0.9597 and a precision of 0.9258 on the CoNSeP dataset, the relatively

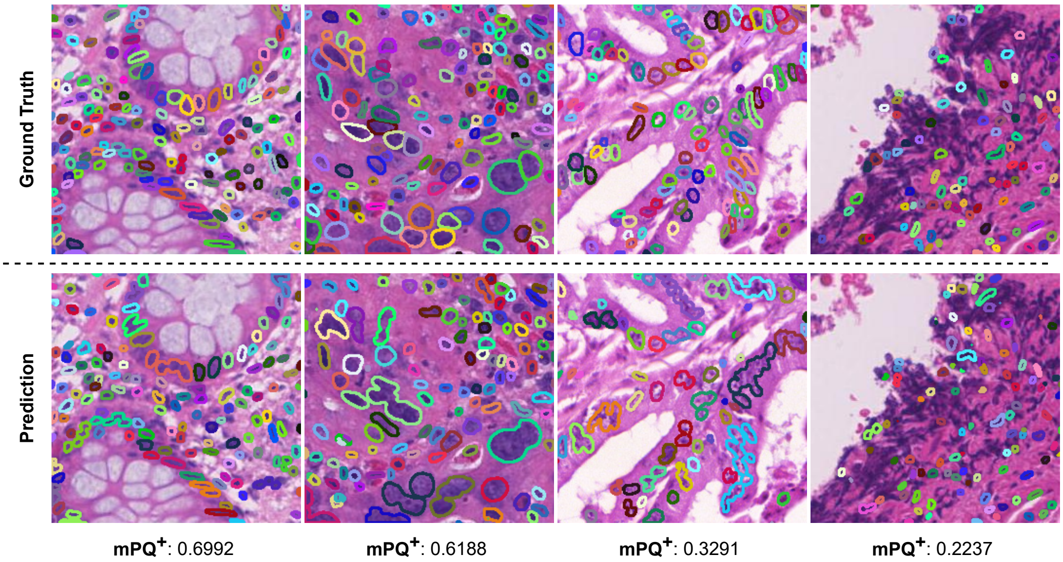


Figure 3.3 UNet instance segmentation prediction on CoNIC patches

Table 3.3 Semantic segmentation results of UNet trained on MoNuSeg dataset

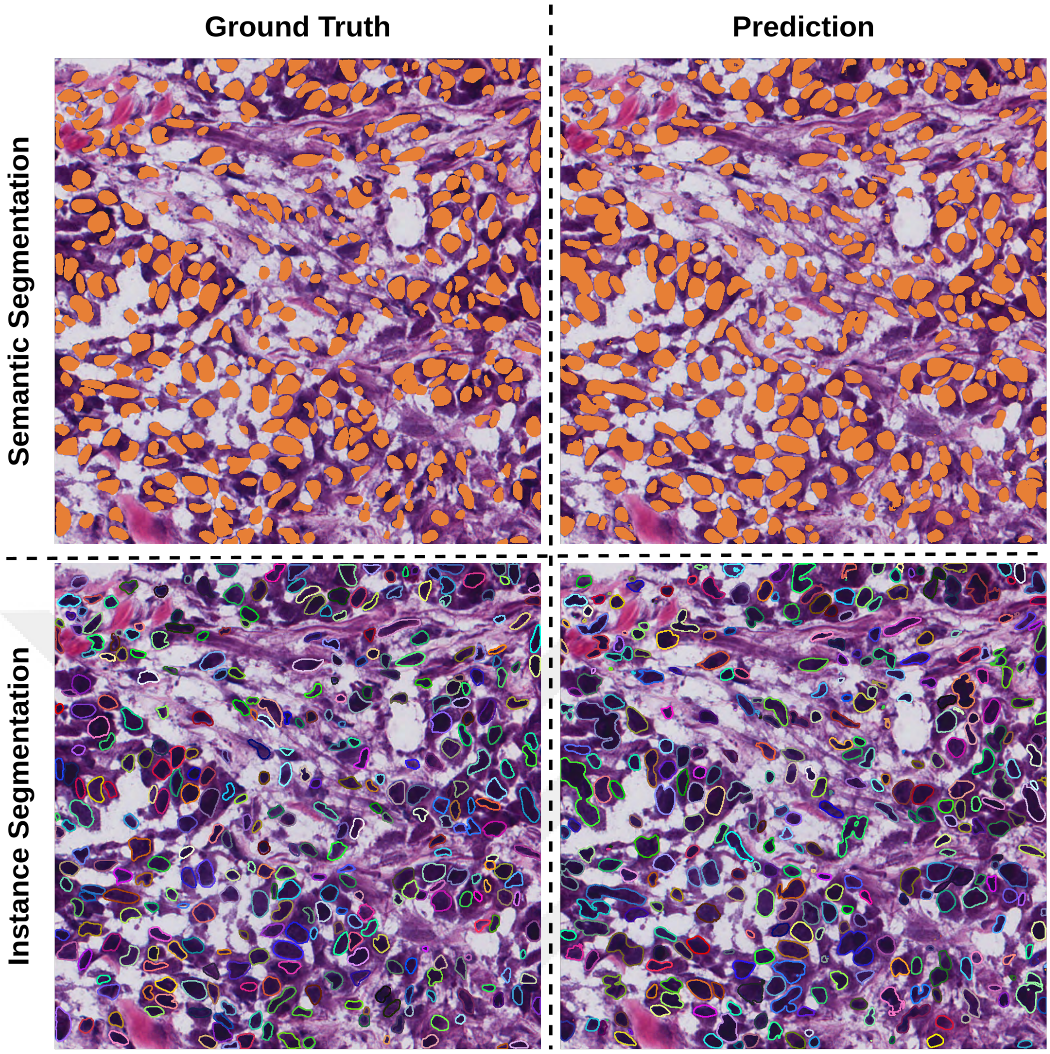
	Test Dataset	μ	Background	Nuclei
Dice	**MoNuSeg**	0.8814	0.9491	0.8137
	CPM17	0.8802	0.9583	0.8021
	CoNSeP	0.8412	0.9424	0.7399
IoU	**MoNuSeg**	0.7946	0.9031	0.6860
	CPM17	0.7948	0.9200	0.6695
	CoNSeP	0.7391	0.8911	0.5872
Recall	**MoNuSeg**	0.8786	0.9518	0.8055
	CPM17	0.8448	0.9882	0.7013
	CoNSeP	0.8220	0.9597	0.6842
Precision	**MoNuSeg**	0.8843	0.9465	0.8222
	CPM17	0.9335	0.9302	0.9367
	CoNSeP	0.8656	0.9258	0.8054

Table 3.4 Instance segmentation results of UNet trained on MoNuSeg dataset

Test Dataset	PQ^+	PQ	SQ	DQ	AJI^+	AJI
MoNuSeg	0.5414	0.5392	0.7631	0.7056	0.5819	0.5530
CPM17	0.5301	0.5282	0.7765	0.6758	0.5899	0.5782
CoNSeP	0.2608	0.2497	0.7448	0.3334	0.2978	0.2120

lower success rate in nuclei segmentation suggests that the model tends to produce background predictions when applied to a different domain.

The instance segmentation results of the UNet model trained with the MoNuSeg dataset are presented in Table 3.4. Similar to the semantic segmentation results, the instance segmentation outcomes on the MoNuSeg and CPM17 datasets are comparable. However, the UNet model has demonstrated significantly lower



UNet model success on this RoI for semantic Segmentation Dice: 0.8803, IoU:0.7910 for instance segmentation mPQ^+: 0.5200

Figure 3.4 UNet model prediction on MoNuSeg RoI

performance on the CoNSeP dataset, with a PQ^+ of 0.2608. While a DQ performance of 0.7056 was achieved on the MoNuSeg dataset, only a DQ of 0.3334 was obtained on the CoNSeP dataset, indicating the model's inability to effectively detect nuclei there. As observed in Figure 3.4, the nuclei in the MoNuSeg dataset are more distinctly separated compared to those in the CoNSeP dataset, where nuclei are closer together. Consequently, it is easier to successfully derive instance segmentation predictions from semantic segmentation predictions in datasets where nuclei are more spaced apart.

A significant factor affecting performance across all tests conducted on the CoNIC, MoNuSeg, CPM17, and CoNSeP datasets is the model's classification of nuclei that are only a few pixels in size. These small-scale classifications can be problematic because each tiny blob is considered a separate nuclei instance. While this does

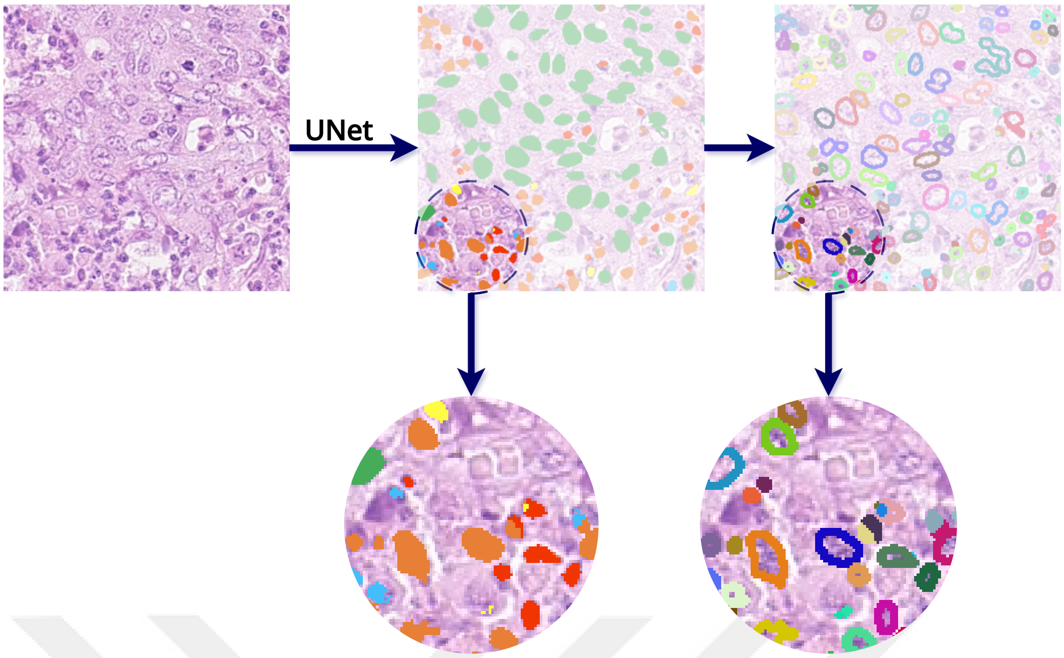


Figure 3.5 UNet false prediction examples

not greatly diminish semantic segmentation performance, it substantially reduces the performance of instance segmentation. An example of how these erroneous predictions impact instance segmentation is illustrated in Figure 3.5.

3.2 Adaptive Thresholding

As previously mentioned, the performance of instance segmentation is compromised by predictions involving only a few pixels. To address this issue, we propose an adaptive thresholding post-processing method. This technique dynamically adjusts the threshold, improving accuracy by effectively distinguishing between true objects and noise.

In general instance segmentation problems, it is impossible to determine the average object size of the target. However, in scanned histopathology images, the magnification level is constant, resulting in minimal size variation between objects of the same class. Leveraging this information enables size data to be used to make informed decisions about whether a predicted nucleus is likely correct or incorrect.

To enable thresholding, it is necessary to determine a threshold value for each class during training. In PQ metric, using a strict match criterion can be too restrictive, as a prediction with 50% or more overlap is typically considered successful, and directly applying thresholding based on ground truth labels may lead to a reduction in true positive cases. Therefore, we suggest that the threshold determination

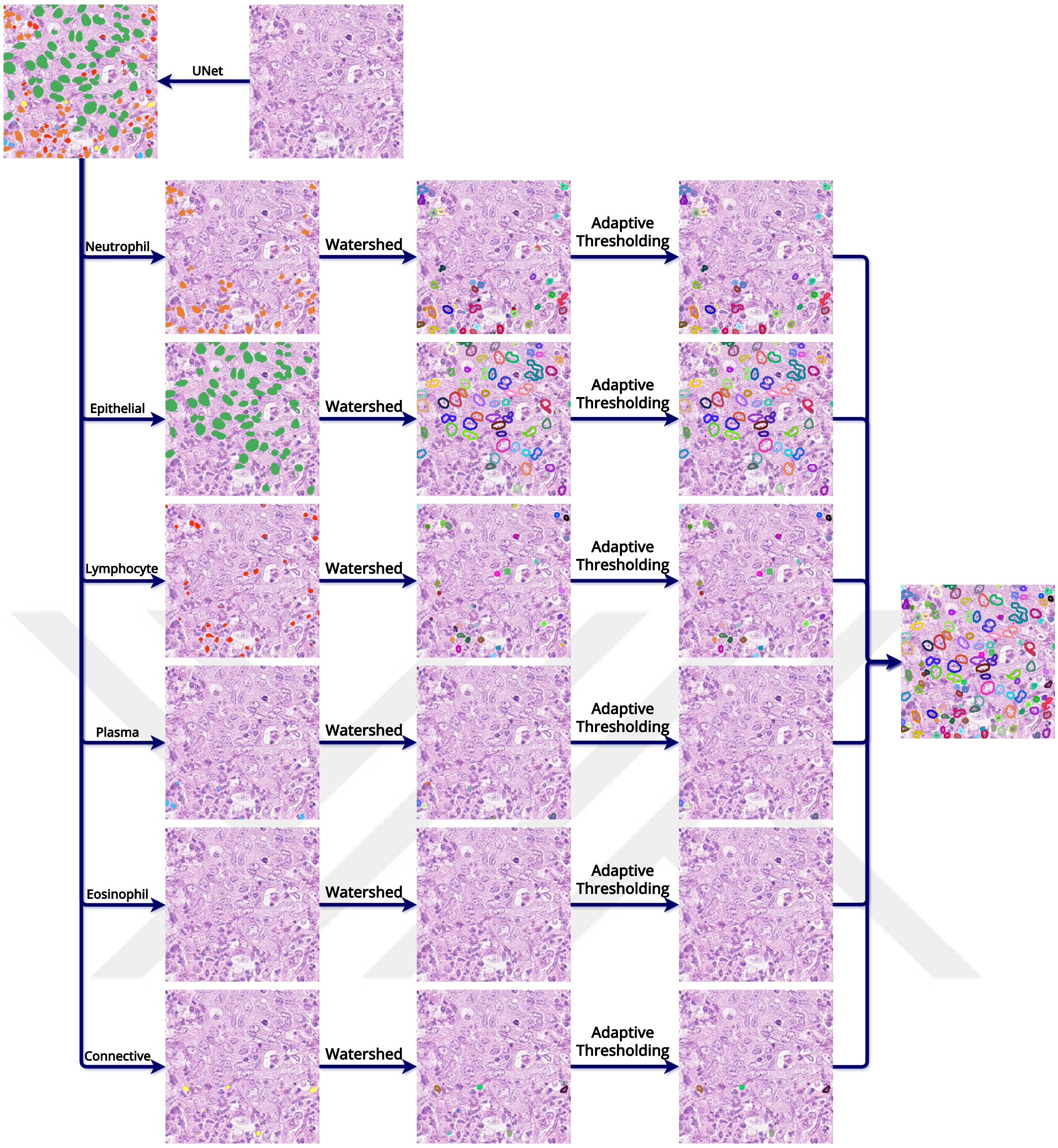


Figure 3.6 Adaptive thresholding methodology

process should be based on measuring performance over the model's validation dataset predictions.

In the adaptive thresholding methodology, semantic segmentation results are first obtained using a model on the validation dataset. Watershed is then applied to the semantic segmentation map for each class to generate instance predictions. Pixel-based thresholding is applied to these instances, yielding PQ^+ performance metrics. The threshold value that produces the best performance for each class is assigned to that class, and this determined threshold is used during testing. The methodology of adaptive thresholding is summarized in Figure 3.6.

Table 3.5 displays the model's performance, trained on Fold-1 of the CoNIC dataset and tested on its validation subset. Thresholding was conducted in increments

Table 3.5 Instance segmentation performance of fold-1 validation based on threshold

Th	**mPQ**	**PQ$^+_{\text{Neu}}$**	**PQ$^+_{\text{Epi}}$**	**PQ$^+_{\text{Lym}}$**	**PQ$^+_{\text{Pla}}$**	**PQ$^+_{\text{Eos}}$**	**PQ$^+_{\text{Con}}$**
5	0.4836	0.3299	0.3841	0.6535	0.5442	0.4076	0.5826
10	0.4939	0.3409	0.3859	0.6623	0.5583	0.4231	0.5930
15	0.5002	0.3489	0.3867	0.6675	0.5663	0.4329	0.5992
20	0.5054	0.3555	0.3866	0.6694	0.5720	0.4451	0.6038
25	0.5085	0.3611	0.3860	0.6674	0.5757	0.4542	0.6066
30	0.5063	0.3622	0.3847	0.6559	0.5732	0.4548	0.6072
35	0.4992	0.3610	0.3825	0.6272	0.5614	0.4615	0.6013
40	0.4857	0.3618	0.3798	0.5828	0.5393	0.4599	0.5904
45	0.4651	0.3596	0.3760	0.5256	0.5013	0.4538	0.5742
50	0.4403	0.3575	0.3709	0.4587	0.4544	0.4487	0.5520
55	0.4093	0.3414	0.3644	0.3838	0.4042	0.4335	0.5286
60	0.3774	0.3269	0.3565	0.3133	0.3473	0.4189	0.5012
65	0.3429	0.2972	0.3483	0.2428	0.2909	0.4075	0.4704

Table 3.6 Comparison of thresholding and adaptive thresholding on CoNIC fold-1 test dataset

Th	**mPQ**	**PQ$^+_{\text{Neu}}$**	**PQ$^+_{\text{Epi}}$**	**PQ$^+_{\text{Lym}}$**	**PQ$^+_{\text{Pla}}$**	**PQ$^+_{\text{Eos}}$**	**PQ$^+_{\text{Con}}$**
-	0.4525	0.3019	0.3991	0.6052	0.4954	0.3555	0.5578
25	0.5070	0.3726	0.4107	0.6476	0.5658	0.4343	0.6109
[30, 15, 20, 25, 35, 30]	0.5105	0.3775	0.4114	0.6500	0.5658	0.4468	0.6115

of 5 pixels, ranging from 5 pixels to 65 pixels in size. The highest performance was achieved with a threshold of 25 pixels. However, Table 3.5 indicates in red that different threshold values yield the peak performance for each class. This suggests that class-by-class implementation of adaptive thresholding will result in more successful outcomes.

Table 3.6 presents a comparison between the outcomes of using a fixed threshold and an adaptive threshold for each class, based on the CoNIC Fold-1 test subset. Applying a fixed threshold improves performance by 5.45% compared to using no thresholding. Implementing adaptive thresholding further enhances performance, resulting in a 5.8% increase. Considering that the most successful models trained on the CoNIC dataset in the literature typically show less than a 1% difference in performance, an improvement of approximately 5-6% is notably significant.

The results of the UNet model employing adaptive thresholding for instance segmentation on the CoNIC dataset are presented in Table 3.7. A performance of 0.5090 mPQ^+ was achieved across 5 folds, which is 5.9% higher compared

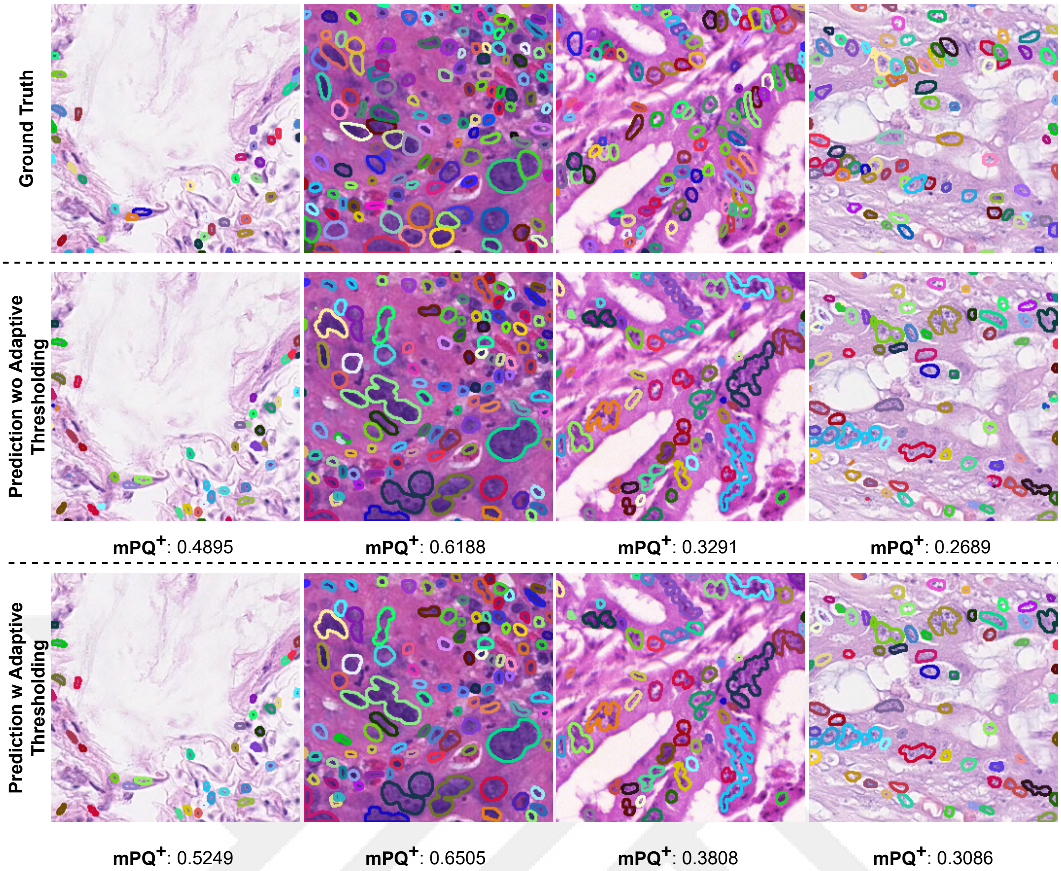


Figure 3.7 Example instance predictions of UNet w/wo adaptive thresholding

Table 3.7 Instance segmentation result of UNet with adaptive thresholding on CoNIC dataset

Fold	mPQ	PQ^+_{Neu}	PQ^+_{Epi}	PQ^+_{Lym}	PQ^+_{Pla}	PQ^+_{Eos}	PQ^+_{Con}
1	0.5105	0.3775	0.4114	0.6500	0.5658	0.4468	0.6115
2	0.5109	0.3748	0.3820	0.6666	0.5755	0.4588	0.6078
3	0.5071	0.3591	0.4000	0.6560	0.5620	0.4551	0.6105
4	0.5109	0.3805	0.4087	0.6580	0.5737	0.4408	0.6037
5	0.5057	0.3962	0.3881	0.6526	0.5627	0.4346	0.6002
μ	0.5090	0.3776	0.3980	0.6566	0.5679	0.4472	0.6067

to results without any thresholding applied. The class-specific PQ^+ values showed notable improvements: Neutrophil increased by 8.5%, epithelial by 1.2%, lymphocyte by 4.3%, plasma by 6.9%, eosinophil by 9.2%, and connective by 5.3%. Figure 3.7 illustrates the impact of these results on instance segmentation performance across CoNIC patches.

Table 3.8 displays the 5-fold semantic segmentation performance of the UNet model on the CoNIC dataset with adaptive thresholding applied. Adaptive thresholding, which reassigns less confident predictions to the background class to enhance instance segmentation performance, has resulted in a slight decrease in semantic

Table 3.8 Semantic segmentation result of UNet with adaptive thresholding on CoNIC dataset

	Fold	μ	Bg	Neu	Epi	Lym	Pla	Eos	Con
Dice	**1**	0.7464	0.9713	0.5267	0.8389	0.8088	0.6586	0.6332	0.7877
	2	0.7475	0.9703	0.5282	0.8416	0.8065	0.6627	0.6434	0.7798
	3	0.7442	0.9717	0.5116	0.8404	0.8089	0.6521	0.6372	0.7872
	4	0.7433	0.9711	0.5203	0.8360	0.7985	0.6623	0.6328	0.7822
	5	0.7474	0.9709	0.5498	0.8413	0.8046	0.6495	0.6366	0.7789
	μ	0.7458	0.9711	0.5273	0.8396	0.8055	0.6570	0.6366	0.7832
IoU	**1**	0.6153	0.9442	0.3575	0.7225	0.6789	0.4910	0.4633	0.6497
	2	0.6160	0.9424	0.3589	0.7264	0.6757	0.4955	0.4742	0.6391
	3	0.6133	0.9450	0.3437	0.7248	0.6791	0.4838	0.4676	0.6491
	4	0.6112	0.9438	0.3516	0.7182	0.6646	0.4951	0.4629	0.6423
	5	0.6154	0.9435	0.3791	0.7261	0.6730	0.4809	0.4670	0.6379
	μ	0.6142	0.9438	0.3582	0.7236	0.6743	0.4893	0.4670	0.6436
Recall	**1**	0.7211	0.9721	0.4532	0.8456	0.8074	0.6169	0.5853	0.7672
	2	0.7313	0.9697	0.4600	0.8564	0.7955	0.6429	0.6371	0.7578
	3	0.7213	0.9727	0.4311	0.8430	0.7946	0.6121	0.6149	0.7808
	4	0.7208	0.9715	0.4357	0.8444	0.7894	0.6353	0.6051	0.7641
	5	0.7293	0.9689	0.4977	0.8631	0.7953	0.6059	0.5986	0.7755
	μ	0.7248	0.9710	0.4555	0.8505	0.7964	0.6226	0.6082	0.7691
Precision	**1**	0.7781	0.9706	0.6285	0.8323	0.8101	0.7065	0.6897	0.8093
	2	0.7675	0.9710	0.6201	0.8272	0.8178	0.6837	0.6498	0.8032
	3	0.7734	0.9708	0.6290	0.8379	0.8236	0.6978	0.6611	0.7936
	4	0.7726	0.9706	0.6458	0.8277	0.8077	0.6917	0.6633	0.8011
	5	0.7691	0.9730	0.6139	0.8205	0.8141	0.6999	0.6798	0.7824
	μ	0.7721	0.9712	0.6275	0.8291	0.8147	0.6959	0.6687	0.7979

Table 3.9 Instance segmentation result of UNet trained on MoNuSeg dataset with adaptive thresholding

Test Dataset	PQ^+	PQ	SQ	DQ	AJI^+	AJI
MoNuSeg	0.5700	0.5701	0.7631	0.7464	0.5824	0.5532
CPM17	0.5602	0.5567	0.7765	0.7126	0.5896	0.5777
CoNSeP	0.2820	0.2721	0.7449	0.3637	0.2972	0.2111

segmentation metrics. The Dice score has decreased from 0.7465 to 0.7458, and the IoU has dropped from 0.6150 to 0.6142. Notably, the recall has fallen more significantly, from 0.7393 to 0.7248, representing a decline of more than 1%. However, as adaptive thresholding aims to reduce false predictions, the precision score has improved from 0.7556 to 0.7721. Given that this study aims to improve instance segmentation performance, the slight decrease in semantic segmentation metrics is considered acceptable.

Table 3.9 presents the instance segmentation results of the UNet model trained on the MoNuSeg dataset, employing adaptive thresholding methodology for both in-domain and out-domain testing. Additionally, the the model's semantic

Table 3.10 Semantic segmentation result of UNet trained on MoNuSeg dataset with adaptive thresholding

	Test Dataset	μ	Background	Nuclei
Dice	**MoNuSeg**	0.8816	0.9492	0.8139
	CPM17	0.8798	0.9583	0.8014
	CoNSeP	0.8408	0.9424	0.7392
IoU	**MoNuSeg**	0.7948	0.9034	0.6862
	CPM17	0.7942	0.9199	0.6686
	CoNSeP	0.7387	0.8911	0.5863
Recall	**MoNuSeg**	0.8785	0.9522	0.8047
	CPM17	0.8442	0.9883	0.7000
	CoNSeP	0.8212	0.9601	0.6823
Precision	**MoNuSeg**	0.8848	0.9463	0.8233
	CPM17	0.9335	0.9300	0.9371
	CoNSeP	0.8659	0.9254	0.8064

Table 3.11 Comparison of the w/wo adaptive thresholding on instance segmentation task

Method	Train Dataset	Test Dataset	mPQ^+
UNet + WS	**MoNuSeg**	**MoNuSeg**	0.5414
		CPM17	0.5301
		CoNSeP	0.2608
	CoNIC	**CoNIC**	0.4496
UNet + WS + AT	**MoNuSeg**	**MoNuSeg**	0.5700
		CPM17	0.5602
		CoNSeP	0.2820
	CoNIC	**CoNIC**	0.5090

segmentation results are given in Table 3.10. Since the MoNuSeg dataset involves only a single class for instance segmentation, applying an adaptive methodology class-by-class is not possible. Moreover, the absence of a separate validation dataset means that conducting step-by-step thresholding on the test subset is not appropriate. Instead, based on empirical assessments and visualization of model predictions (see Figure 3.8), a 40-pixel threshold was selected. This method makes sure that the results are reliable, but direct comparisons with datasets like CoNIC, which have a validation subset, might not show equivalency correctly because they use different validation datasets.

Results with and without adaptive thresholding are compared in Table 3.11, showing significant performance enhancements across multiple datasets. The MoNuSeg and CPM17 datasets each increased by about 3%, followed by CoNSeP with a 2% increase, and CoNIC by 6%. These results affirm the substantial impact of this post-processing method on performance. However, there are drawbacks

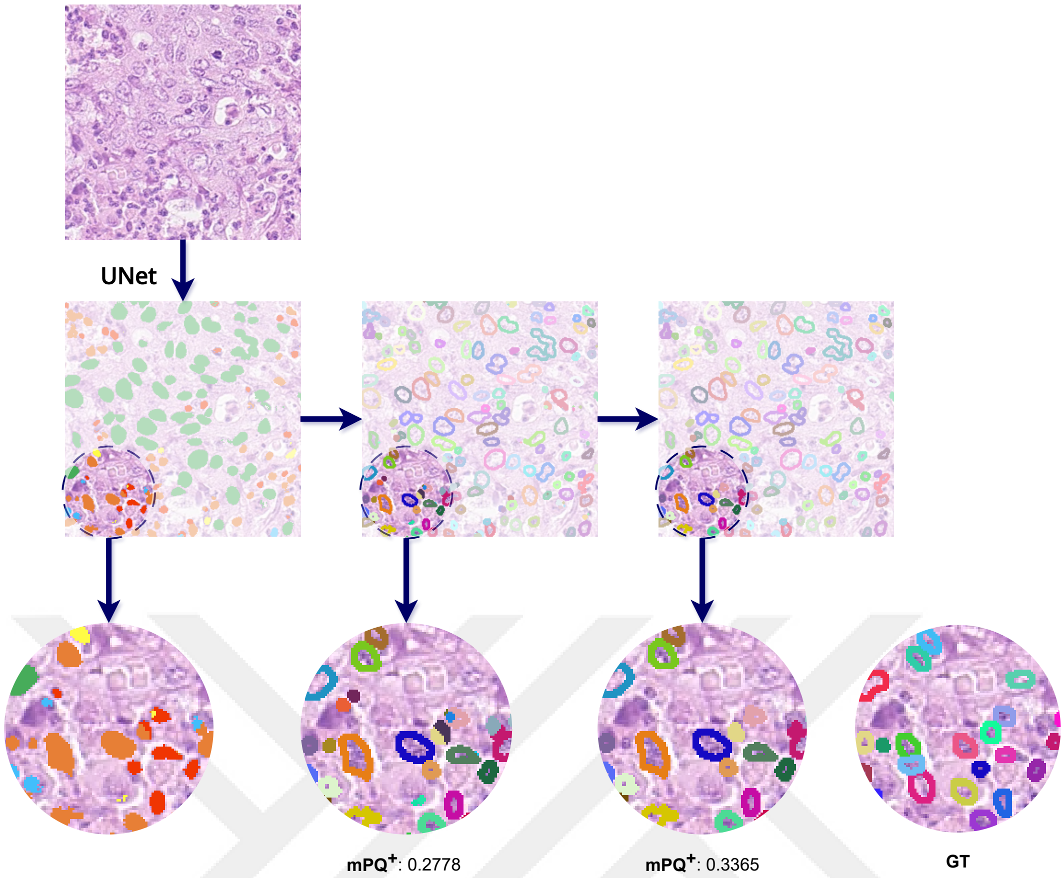


Figure 3.8 Effect of adaptive thresholding

to the technique, particularly its reliance on a validation dataset. In cases like MoNuSeg, where no validation set is available, optimizing an additional threshold hyperparameter becomes necessary, which can complicate the model's application.

3.3 Examination of Loss Functions with Classical U-Net

Experiments on the CoNIC dataset demonstrate a clear correlation between significant class imbalance and reduced performance across nuclei classifications. To address the issue of imbalanced data, several loss functions were evaluated. The UNet model was trained on the CoNIC dataset, utilizing 5-fold cross-validation to assess the effectiveness of each loss function. Additionally, the UNet model was adapted for the instance segmentation task using a combination of watershed and adaptive thresholding methodologies.

The analysis used eight different loss functions: Cross-entropy (CE), Dice [76], Focal [77], Tversky [78], Focal-Tversky (FT) [79], Log-Cosh Dice (LCD) [80], L1, and Mean Square Error (MSE, also known as L2). The mathematical expressions for each of these functions are provided in Table 3.12.

Table 3.12 Mathematical expressions of analyzed loss functions

Loss function	Mathematical Expression
CE loss	$L_{\textit{Cross-entropy}} = -\frac{1}{N}\sum_{c=1}^{C}\sum_{i=1}^{N} g_i^c \log s_i^c$
Dice loss [76]	$L_{\textit{Dice}} = 1 - \frac{2\sum_{c=1}^{C}\sum_{i=1}^{N} g_i^c s_i^c}{\sum_{c=1}^{C}\sum_{i=1}^{N} g_i^c + \sum_{c=1}^{C}\sum_{i=1}^{N} s_i^c}$
Focal loss [77]	$L_{\textit{Focal}} = -\frac{1}{N}\sum_{c=1}^{C}\sum_{i=1}^{N}(1-s_i^c)^\gamma g_i^c \log s_i^c$
Tversky loss [78]	$L_{\textit{Tversky}} = 1 - T(\alpha,\beta)$, $L_{\textit{Tversky}} = 1 - \frac{\sum_{c=1}^{C}\sum_{i=1}^{N} g_i^c s_i^c}{\sum_{c=1}^{C}\sum_{i=1}^{N} g_i^c s_i^c + \alpha\sum_{c=1}^{C}\sum_{i=1}^{N}(1-g_i^c)s_i^c + \beta\sum_{c=1}^{C}\sum_{i=1}^{N} g_i^c(1-s_i^c)}$
FT loss [79]	$L_{\textit{Focal-Tversky}} = (L_{\textit{Tversky}})^{\frac{1}{\gamma}}$
LCD loss [80]	$L_{\textit{Log-cosh dice}} = \log(cosh(L_{\textit{Dice}})), \quad cosh(x) = \frac{e^x + e^{-x}}{2}$
L1 loss	$L_{\textit{L1}} = \frac{1}{N}\sum_{c=1}^{C}\sum_{i=1}^{N} g_i^c\lvert 1-s_i^c\rvert$
MSE loss	$L_{\textit{MSE}} = \frac{1}{N}\sum_{c=1}^{C}\sum_{i=1}^{N} g_i^c(1-s_i^c)^2$

g_i^c is the ground truth binary indicator of class label c of pixel i, and s_i^c is the corresponding predicted segmentation probability [81].
C is the total number of nuclei classes, and N is the total number of the pixels.
γ is an exponential coefficient in focal loss function and it is stated that γ=2 yields best performance [77, 81].
α and β are hyper-parameters that control the balance between the false negatives and false positives [81].

Table 3.13 presents the fold-based instance segmentation results for the CoNIC dataset. A stable loss function should show low variation in performance across similar folds. The cross-entropy loss function best illustrates this stability, with the smallest difference between the highest (0.5109 mPQ) and lowest (0.5057 mPQ) performance. In contrast, other loss functions show significant variability, as indicated by the metrics PQ^+_{Neu}, PQ^+_{Pla}, and PQ^+_{Eos}. Some loss functions result in zero performance, whereas others have more than 10% difference between the best and worst performing folds. Cross-entropy and focal loss consistently generate the most reliable outcomes.

Table 3.14 shows the mean instance segmentation performance by loss function. Cross-entropy loss achieved the highest results with 0.5090 mPQ^+. Although cross-entropy did not produce the best results for each nuclei class, it consistently performed well across all classes. Additionally, L2 (mean square error) and focal

Table 3.13 UNet instance segmentation results based on loss functions

	Fold	mPQ	PQ^+_{Neu}	PQ^+_{Epi}	PQ^+_{Lym}	PQ^+_{Pla}	PQ^+_{Eos}	PQ^+_{Con}
Cross Entropy	**1**	0.5105	0.3775	0.4114	0.6500	0.5658	0.4468	0.6115
	2	0.5109	0.3748	0.3820	0.6666	0.5755	0.4588	0.6078
	3	0.5071	0.3591	0.4000	0.6560	0.5620	0.4551	0.6105
	4	0.5109	0.3805	0.4087	0.6580	0.5737	0.4408	0.6037
	5	0.5057	0.3962	0.3881	0.6526	0.5627	0.4346	0.6002
Dice	**1**	0.3671	0.0000	0.4112	0.6436	0.5346	0.0000	0.6131
	2	0.2706	0.0000	0.3837	0.6362	0.0000	0.0000	0.6035
	3	0.3643	0.0000	0.3992	0.6480	0.5282	0.0000	0.6106
	4	0.2749	0.0000	0.4082	0.6318	0.0000	0.0000	0.6097
	5	0.3515	0.0000	0.4034	0.6435	0.4559	0.0000	0.6061
Focal	**1**	0.5077	0.3797	0.4106	0.6542	0.5757	0.4187	0.6071
	2	0.5094	0.3673	0.3786	0.6690	0.5830	0.4587	0.5995
	3	0.5023	0.3449	0.3949	0.6562	0.5699	0.4418	0.6064
	4	0.5098	0.3798	0.4059	0.6596	0.5797	0.4325	0.6014
	5	0.5079	0.4006	0.3994	0.6594	0.5648	0.4203	0.6033
Focal Tversky	**1**	0.3797	0.1630	0.3721	0.5726	0.4209	0.2527	0.4966
	2	0.3946	0.2166	0.3540	0.6121	0.4242	0.2968	0.4639
	3	0.3763	0.2092	0.3554	0.5712	0.4280	0.2631	0.4307
	4	0.4131	0.2473	0.3722	0.5980	0.4454	0.3412	0.4745
	5	0.4241	0.2641	0.4132	0.6209	0.4570	0.3369	0.4523
L1	**1**	0.3642	0.0000	0.4129	0.6410	0.5210	0.0000	0.6105
	2	0.2712	0.0000	0.3819	0.6378	0.0000	0.0000	0.6078
	3	0.2723	0.0000	0.4000	0.6265	0.0000	0.0000	0.6075
	4	0.3680	0.0000	0.4077	0.6514	0.5381	0.0000	0.6108
	5	0.3635	0.0000	0.4018	0.6501	0.5236	0.0000	0.6056
L2	**1**	0.5074	0.3720	0.4096	0.6523	0.5662	0.4314	0.6126
	2	0.5078	0.3599	0.3878	0.6699	0.5612	0.4587	0.6095
	3	0.5026	0.3469	0.3931	0.6569	0.5617	0.4474	0.6094
	4	0.5015	0.3585	0.4062	0.6599	0.5718	0.4045	0.6078
	5	0.5073	0.3798	0.3989	0.6602	0.5620	0.4339	0.6089
Log Cosh Dice	**1**	0.3880	0.2223	0.2696	0.5797	0.4471	0.3328	0.4766
	2	0.3970	0.2480	0.2956	0.6141	0.4134	0.3166	0.4943
	3	0.3847	0.1993	0.2766	0.5700	0.4349	0.3452	0.4822
	4	0.3893	0.2465	0.2387	0.5817	0.4632	0.3545	0.4511
	5	0.4217	0.2851	0.3472	0.5893	0.4449	0.3637	0.4999
Tversky	**1**	0.4080	0.2368	0.3137	0.6106	0.4571	0.3403	0.4894
	2	0.3909	0.2441	0.2584	0.6075	0.4421	0.3044	0.4888
	3	0.3775	0.1973	0.2430	0.5807	0.4399	0.3496	0.4543
	4	0.3927	0.2269	0.3218	0.5528	0.4273	0.3391	0.4881
	5	0.4161	0.2608	0.3714	0.5959	0.4239	0.3504	0.4944

loss demonstrated nearly equivalent performance to cross-entropy loss.

The results of the semantic segmentation task are shown in Table 3.15. In contrast

Table 3.14 Comparison of the instance segmentation performance of the loss function on CoNIC dataset

	$\mathbf{mPQ^+}$	$\mathbf{PQ^+_{Neu}}$	$\mathbf{PQ^+_{Epi}}$	$\mathbf{PQ^+_{Lym}}$	$\mathbf{PQ^+_{Pla}}$	$\mathbf{PQ^+_{Eos}}$	$\mathbf{PQ^+_{Con}}$
L1	0.3279	0.0000	0.4009	0.6414	0.3165	0.0000	0.6084
Dice	0.3257	0.0000	0.4011	0.6406	0.3037	0.0000	0.6086
Log Cosh Dice	0.3961	0.2402	0.2856	0.5870	0.4407	0.3426	0.4808
Tversky	0.3970	0.2332	0.3017	0.5895	0.4381	0.3368	0.4830
Focal Tversky	0.3975	0.2200	0.3734	0.5950	0.4351	0.2981	0.4636
L2	0.5053	0.3634	0.3991	0.6598	0.5646	0.4352	0.6096
Focal	0.5074	0.3745	0.3979	0.6597	0.5746	0.4344	0.6035
Cross Entropy	0.5090	0.3776	0.3980	0.6566	0.5679	0.4472	0.6067

Table 3.15 Comparison of the semantic segmentation performance of the loss function on CoNIC dataset

	Loss Function	μ	**Bg**	**Neu**	**Epi**	**Lym**	**Pla**	**Eos**	**Con**
Dice	**L1**	0.5371	0.9714	0.0000	0.8407	0.7877	0.3738	0.0000	0.7861
	Dice	0.5356	0.9714	0.0000	0.8411	0.7870	0.3630	0.0000	0.7864
	Log Cosh Dice	0.6504	0.9606	0.3592	0.7925	0.7388	0.5316	0.4924	0.6777
	Tversky	0.6401	0.9618	0.3284	0.7822	0.7232	0.5259	0.4929	0.6661
	Focal Tversky	0.6341	0.9626	0.3305	0.7802	0.7199	0.5271	0.4646	0.6538
	L2	0.7437	0.9716	0.5218	0.8403	0.8078	0.6585	0.6204	0.7857
	Focal	0.7474	0.9712	0.5344	0.8390	0.8090	0.6661	0.6285	0.7834
	Cross Entropy	0.7458	0.9711	0.5273	0.8396	0.8055	0.6570	0.6366	0.7832
IoU	**L1**	0.4626	0.9444	0.0000	0.7252	0.6499	0.2715	0.0000	0.6475
	Dice	0.4611	0.9444	0.0000	0.7258	0.6490	0.2607	0.0000	0.6481
	Log Cosh Dice	0.5126	0.9242	0.2194	0.6564	0.5860	0.3621	0.3272	0.5127
	Tversky	0.5024	0.9264	0.1969	0.6428	0.5670	0.3569	0.3271	0.4999
	Focal Tversky	0.4965	0.9278	0.1983	0.6396	0.5628	0.3580	0.3031	0.4862
	L2	0.6126	0.9448	0.3530	0.7246	0.6776	0.4909	0.4498	0.6471
	Focal	0.6160	0.9440	0.3647	0.7227	0.6792	0.4994	0.4584	0.6439
	Cross Entropy	0.6150	0.9443	0.3624	0.7236	0.6757	0.4911	0.4634	0.6448
Precision	**L1**	0.5349	0.9710	0.0000	0.8293	0.7570	0.3969	0.0000	0.7900
	Dice	0.5350	0.9710	0.0000	0.8292	0.7547	0.3999	0.0000	0.7905
	Log Cosh Dice	0.6211	0.9751	0.3374	0.7415	0.7234	0.5187	0.4190	0.6324
	Tversky	0.6139	0.9718	0.2666	0.7525	0.7370	0.4762	0.4572	0.6361
	Focal Tversky	0.6567	0.9560	0.3448	0.8068	0.7806	0.5099	0.4948	0.7044
	L2	0.7546	0.9729	0.5806	0.8294	0.7999	0.6832	0.6267	0.7894
	Focal	0.7606	0.9719	0.5965	0.8285	0.8054	0.6941	0.6311	0.7965
	Cross Entropy	0.7556	0.9727	0.5872	0.8281	0.8047	0.6797	0.6295	0.7873
Recall	**L1**	0.5405	0.9717	0.0000	0.8524	0.8232	0.3541	0.0000	0.7822
	Dice	0.5377	0.9718	0.0000	0.8533	0.8240	0.3326	0.0000	0.7825
	Log Cosh Dice	0.6943	0.9467	0.3918	0.8558	0.7590	0.5665	0.6045	0.7357
	Tversky	0.6860	0.9520	0.4522	0.8224	0.7123	0.5990	0.5521	0.7117
	Focal Tversky	0.6278	0.9692	0.3488	0.7558	0.6744	0.5485	0.4757	0.6220
	L2	0.7351	0.9703	0.4746	0.8516	0.8159	0.6356	0.6154	0.7821
	Focal	0.7366	0.9705	0.4846	0.8499	0.8127	0.6407	0.6270	0.7707
	Cross Entropy	0.7393	0.9700	0.4872	0.8516	0.8083	0.6394	0.6379	0.7809

to the instance segmentation task, the focal loss function has emerged as the most successful model in terms of the Dice coefficient, IoU, and precision metrics. However, the cross-entropy loss has achieved the best performance in the recall

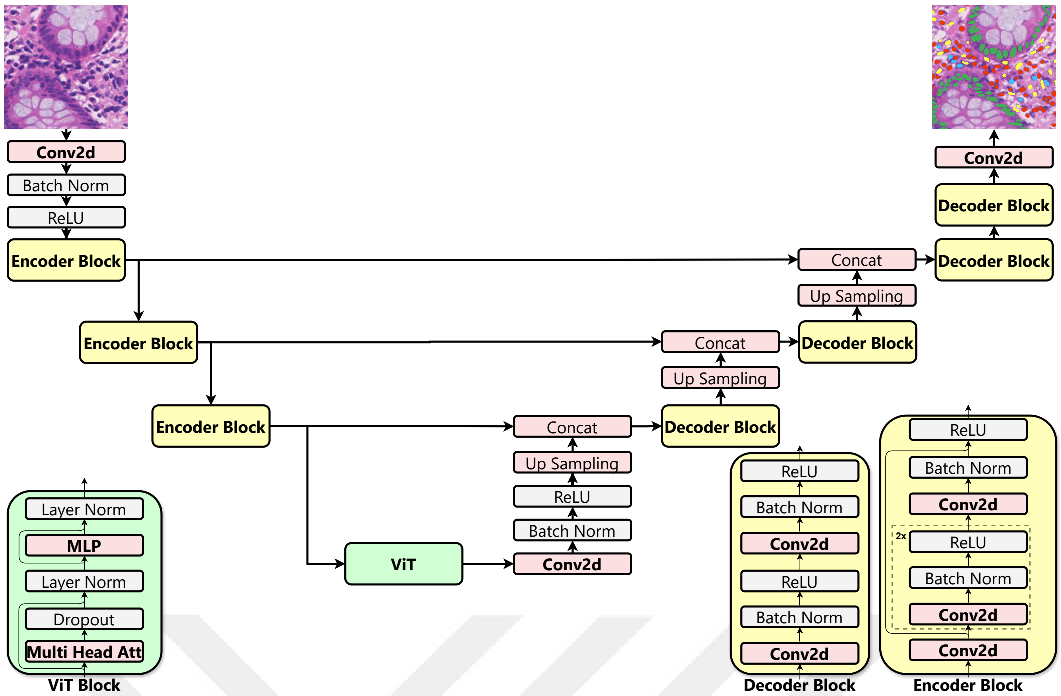


Figure 3.9 TransUNet architecture

metric.

In both instance and semantic segmentation tasks, the L2, focal, and cross-entropy loss functions have produced similar results. The lack of significant differences among these three loss functions makes it challenging to draw definitive conclusions. However, given that the primary focus of this study is on instance segmentation, it can be stated that cross-entropy loss has achieved better results. Additionally, the performance of the other five loss functions has been substantially lower, suggesting that they may not be suitable for nuclei instance segmentation and classification tasks.

3.4 TransUNet Architecture

Transformers [82] have gained substantial popularity in current literature, particularly in the field of natural language processing [83, 84]. Transformer architectures, originally designed to capture relationships between word sequences using an attention mechanism, have since adapted for computer vision tasks [85, 86]. Transformer architectures adapt to image inputs by treating image patches as word embeddings. Vision transformers [87], unlike traditional CNN architectures, enable selective feature extraction through their attention mechanisms [88].

TransUNet [89] is a specialized architectural adaptation of the vision transformer for semantic segmentation tasks. Figure 3.9 shows the TransUNet model's architecture, which is very similar to the UNet architecture. TransUNet, like UNet, uses convolution filters to extract feature vectors, and connections between levels to combine fine and coarse feature vectors. The TransUNet model's decoder block is also based on the same architecture as UNet. However, notable distinctions exist in the encoder block, including an increased number of consecutive convolution filters and the incorporation of residual connections. Additionally, instead of max pooling, convolution filters with a 2×2 stride are used for downsampling. The most distinctive difference is the use of a vision transformer block (ViT) at the final level, which enhances the model's capability to handle complex spatial hierarchies effectively [87].

The input size is reduced by half at each level using a stride of 2×2, creating a 32×32 ViT input from a 256×256 input image. Each downsampled pixel is treated like a patch to which a ViT block is applied. Let $X \in R^{h/8 \times w/8}$ be the feature matrix, representing downsampled input image patches by convolution blocks.

$$\text{Attention}(\mathbf{Q}, \mathbf{K}, \mathbf{V}) = \text{softmax}\left(\frac{\mathbf{QK}^T}{\sqrt{d_k}}\right)\mathbf{V} \tag{3.7}$$

$$\text{head}_i = \text{Attention}(\mathbf{Q}W_i^Q, \mathbf{K}W_i^K, \mathbf{V}W_i^V) \quad \text{for} \quad i \in \{1, \dots, h\} \tag{3.8}$$

$$\text{MultiHead}(\mathbf{Q}, \mathbf{K}, \mathbf{V}) = [\text{head}_1, \dots, \text{head}_h]W^O \tag{3.9}$$

where **Q**, **K**, and **V** represent the query, key, and value matrices, respectively, and W^O, W_i^Q, W_i^K, and W_i^V are learnable parameters. In a vision transformer, **Q**, **K**, and **V** are derived from the input image patches X. The softmax(.) function normalizes the dot product of **Q** and **K**, scaled by $\sqrt{d_k}$, to maintain numerical stability. After computing each head, the results are concatenated, and then the concatenated matrix is multiplied by W^O, another learnable parameter, to produce the final output.

The TransUNet model was developed for semantic segmentation and, like UNet, necessitates the conversion of results into instance segmentation predictions. To achieve this, the watershed algorithm, as utilized in UNet, has been employed. It was observed that the adaptive thresholding approach yields higher performance outcomes. Therefore, it has been adapted to the instance segmentation map, resulting in the final instance segmentation map.

3.4.1 TransUNet Experimental Setup

The TransUNet was trained using the CoNIC and MoNuSeg datasets. For the CoNIC dataset, 5-fold cross-validation was conducted to evaluate the model's

Table 3.16 TranUNet semantic segmentation results on CoNIC dataset with adaptive thresholding

	Fold	μ	Bg	Neu	Epi	Lym	Pla	Eos	Con
Dice	1	0.7179	0.9697	0.4690	0.8291	0.7950	0.6233	0.5795	0.7595
	2	0.7144	0.9679	0.4620	0.8286	0.7887	0.6143	0.5920	0.7474
	3	0.7112	0.9695	0.4443	0.8282	0.7931	0.6131	0.5835	0.7465
	4	0.7090	0.9692	0.4263	0.8240	0.7862	0.6264	0.5820	0.7489
	5	0.7161	0.9697	0.4642	0.8302	0.7956	0.6156	0.5868	0.7505
	μ	0.7137	0.9692	0.4532	0.8280	0.7917	0.6185	0.5847	0.7505
IoU	1	0.5841	0.9411	0.3064	0.7081	0.6598	0.4528	0.4079	0.6123
	2	0.5796	0.9378	0.3004	0.7073	0.6511	0.4433	0.4204	0.5967
	3	0.5771	0.9408	0.2856	0.7067	0.6571	0.4421	0.4119	0.5955
	4	0.5749	0.9403	0.2709	0.7007	0.6477	0.4560	0.4104	0.5985
	5	0.5820	0.9412	0.3023	0.7097	0.6605	0.4446	0.4152	0.6006
	μ	0.5795	0.9402	0.2931	0.7065	0.6553	0.4478	0.4132	0.6007
Recall	1	0.6808	0.9718	0.3839	0.8357	0.7936	0.5498	0.5107	0.7201
	2	0.6760	0.9706	0.3632	0.8343	0.7687	0.5414	0.5485	0.7052
	3	0.6754	0.9728	0.3716	0.8339	0.7858	0.5386	0.5364	0.6887
	4	0.6737	0.9722	0.3223	0.8263	0.7916	0.5646	0.5380	0.7007
	5	0.6760	0.9726	0.3616	0.8316	0.7813	0.5510	0.5167	0.7171
	μ	0.6764	0.9720	0.3605	0.8324	0.7842	0.5491	0.5300	0.7064
Precision	1	0.7688	0.9675	0.6025	0.8226	0.7965	0.7195	0.6697	0.8034
	2	0.7686	0.9652	0.6346	0.8229	0.8098	0.7098	0.6429	0.7950
	3	0.7582	0.9662	0.5524	0.8225	0.8005	0.7115	0.6397	0.8148
	4	0.7628	0.9663	0.6296	0.8218	0.7808	0.7033	0.6337	0.8041
	5	0.7739	0.9668	0.6481	0.8288	0.8103	0.6973	0.6789	0.7872
	μ	0.7665	0.9664	0.6134	0.8237	0.7996	0.7083	0.6530	0.8009

performance. Additionally, the model trained on the MoNuSeg dataset was further cross-tested using the CPM17 and CoNSeP datasets. As is common in the literature, we used the MoNuSeg test dataset as a validation dataset.

The Cyclic learning rate scheduler was employed, with a maximum learning rate set at 0.001. The AdamW optimizer was utilized, and the batch size was determined to be 16. Cross entropy was used as loss function. The model was trained using input sizes of 256×256 patches without any resizing, although testing was conducted on RoI. In total, the training involved 40,000 steps.

3.4.2 TransUNet Results

Table 3.16 presents the performance of the TransUNet model on the CoNIC dataset for semantic segmentation tasks. Across all metrics, the highest results were achieved for the epithelial class, with a dice of 0.8280, an IoU of 0.7065, a recall of 0.8324, and a precision of 0.8237. Conversely, the lowest results were observed for the neutrophil class, with a dice of 0.4532, an IoU of 0.2931, a recall of

Table 3.17 TransUNet instance segmentation results on CoNIC dataset with adaptive thresholding

Fold	mPQ$^+$	PQ$^+_{\text{Neu}}$	PQ$^+_{\text{Epi}}$	PQ$^+_{\text{Lym}}$	PQ$^+_{\text{Pla}}$	PQ$^+_{\text{Eos}}$	PQ$^+_{\text{Con}}$
1	0.4669	0.2978	0.3958	0.6320	0.5195	0.3916	0.5645
2	0.4644	0.3048	0.3693	0.6454	0.5104	0.3935	0.5629
3	0.4535	0.2606	0.3774	0.6315	0.5156	0.3809	0.5551
4	0.4604	0.2742	0.3980	0.6370	0.5250	0.3669	0.5611
5	0.4673	0.3083	0.3931	0.6370	0.5189	0.3811	0.5654
μ	0.4625	0.2891	0.3867	0.6366	0.5179	0.3828	0.5618

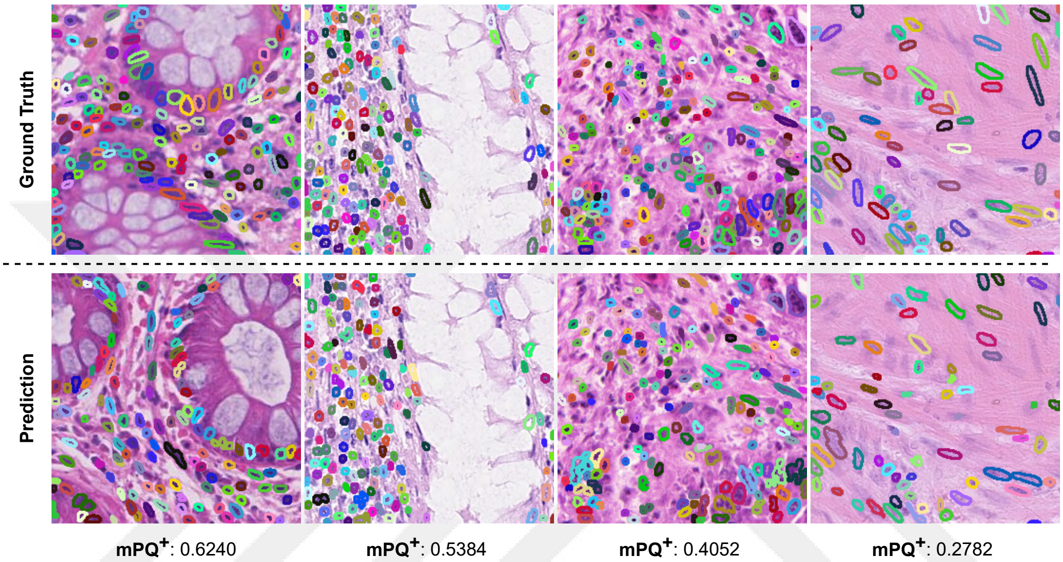


Figure 3.10 TransUNet instance prediction on CoNIC patches

0.3605, and a precision of 0.6134. Considering that the epithelial class had the most instances and the neutrophil class had one of the fewest, it is obvious that semantic segmentation success correlates with the distribution of class instances.

Instance segmentation results on the CoNIC dataset using 5-fold cross-validation are presented in Table 3.17. It is noteworthy that the mPQ^+ values are closely matched across different folds. The highest class-based PQ^+ value of 0.6366 was observed in the lymphocyte class, while the lowest was 0.2891 for the neutrophil class. Considering that the epithelial class had the highest number of examples and the eosinophil class had the fewest, it is not possible to directly correlate performance with the number of class examples. However, upon examining the model instance predictions given in Figure 3.10, the morphological differences between the classes are striking. Given this observation, it can be concluded that one of the primary factors influencing the model's instance segmentation performance is its ability to distinguish between two closely spaced nuclei.

With a mPQ^+ value of 0.4625, the TransUNet has exhibited approximately 4%

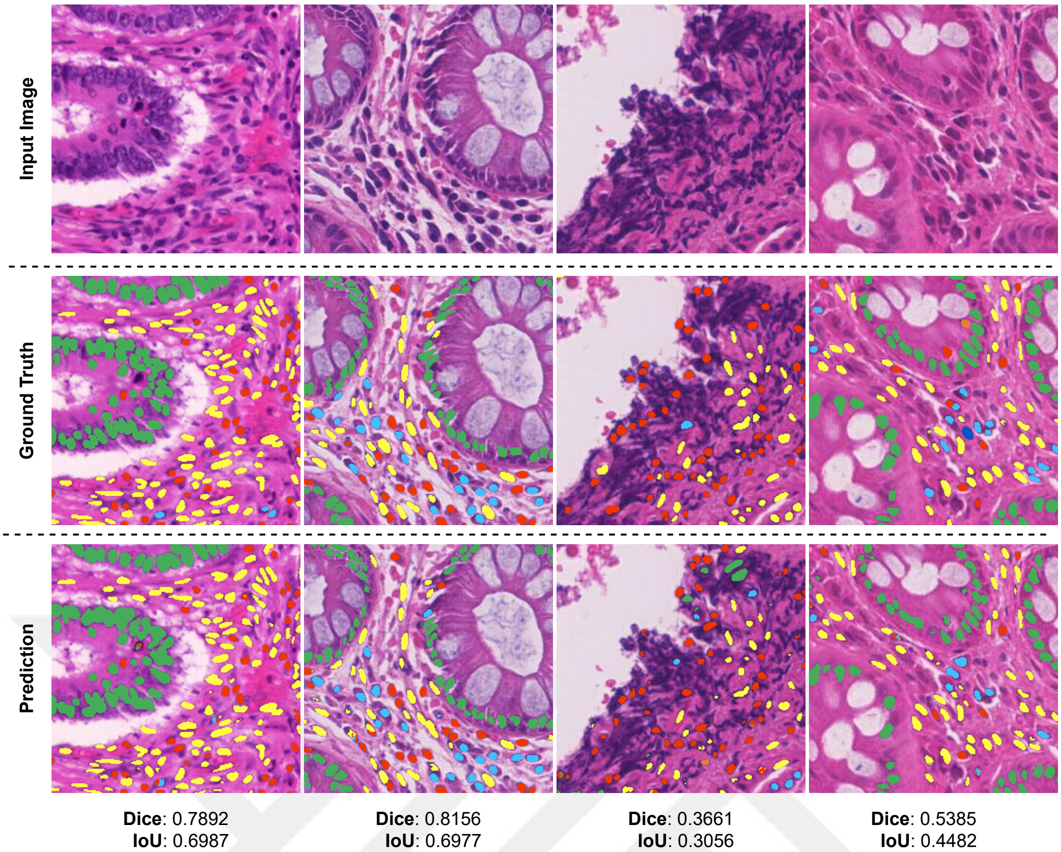


Figure 3.11 TransUNet semantic segmentation on CoNIC patches

Table 3.18 Instance segmentation results TransUNet trained on MoNuSeg with adaptive thresholding

Test Dataset	PQ^+	PQ	SQ	DQ	AJI^+	AJI
MoNuSeg	0.5437	0.5381	0.7599	0.7071	0.5507	0.5143
CPM17	0.5358	0.5244	0.7704	0.6756	0.5605	0.5232
CoNSeP	0.2327	0.2282	0.7332	0.3098	0.2609	0.1900

lower performance than the UNet model. Similarly, its performance in semantic segmentation is lower to that of UNet. As illustrated in Figure 3.11, the TransUNet model exhibits considerable uncertainty. It is evident that a single nucleus contain pixels classified into two different categories. Furthermore, despite the use of an adaptive thresholding approach, a few pixel-sized nuclei are predicted on the background.

The trainings conducted with the MoNuSeg dataset were intended to assess both the nuclei instance segmentation performance and the model's cross-domain performance. The nuclei instance segmentation success of the TransUNet model is provided in Table 3.18. For in-domain testing on the MoNuSeg dataset, a PQ^+ performance of 0.5381 was achieved. In out-domain tests, a closely similar result of

Table 3.19 Semantic segmentation results TransUNet trained on MoNuSeg with adaptive thresholding

	Test Dataset	μ	Background	Nuclei
Dice	**MoNuSeg**	0.8655	0.9406	0.7904
	CPM17	0.8714	0.9529	0.7899
	CoNSeP	0.7852	0.9251	0.6453
IoU	**MoNuSeg**	0.7707	0.8879	0.6534
	CPM17	0.7814	0.9100	0.6527
	CoNSeP	0.6685	0.8606	0.4763
Recall	**MoNuSeg**	0.8702	0.9360	0.8044
	CPM17	0.8496	0.9715	0.7278
	CoNSeP	0.7620	0.9500	0.5741
Precision	**MoNuSeg**	0.8611	0.9453	0.7768
	CPM17	0.8993	0.9350	0.8635
	CoNSeP	0.8191	0.9015	0.7367

0.5437 PQ^{+} was obtained on the CPM17 dataset. However, on the CoNSeP dataset, which contains only colon tissue samples from a single scanner, a significantly lower PQ^{+} performance of 0.2327 was observed.

The semantic segmentation results of the TransUNet model trained with the MoNuSeg dataset are provided in Table 3.19. The CoNSeP dataset semantic segmentation result shows relatively low performance across all metrics, which is consistent with the instance segmentation results. The model exhibits a recall performance of 0.5741, indicating that it predominantly predicts background. Additionally, a detection quality (DQ) of 0.3098 supports this observation.

The similarity of segmentation quality (SQ) and precision values across domains suggests that, while the TransUNet model accurately segments the nuclei it identifies, it only does so for a limited number of instances.

3.5 Swin-Unet Architecture

The Swin Transformer [90] is a pioneering architecture designed specifically for computer vision tasks. It employs a window-based self-attention methodology that not only reduces complexity but also enhances performance. This approach has made the Swin Transformer a popular backbone for a variety of applications, from object detection [91] to object tracking [92].

The Swin-Unet [93] model fundamentally builds upon the UNet architecture to enhance semantic segmentation capabilities. It was specifically developed for multi-organ tissue segmentation in CT images, where it has demonstrated superior

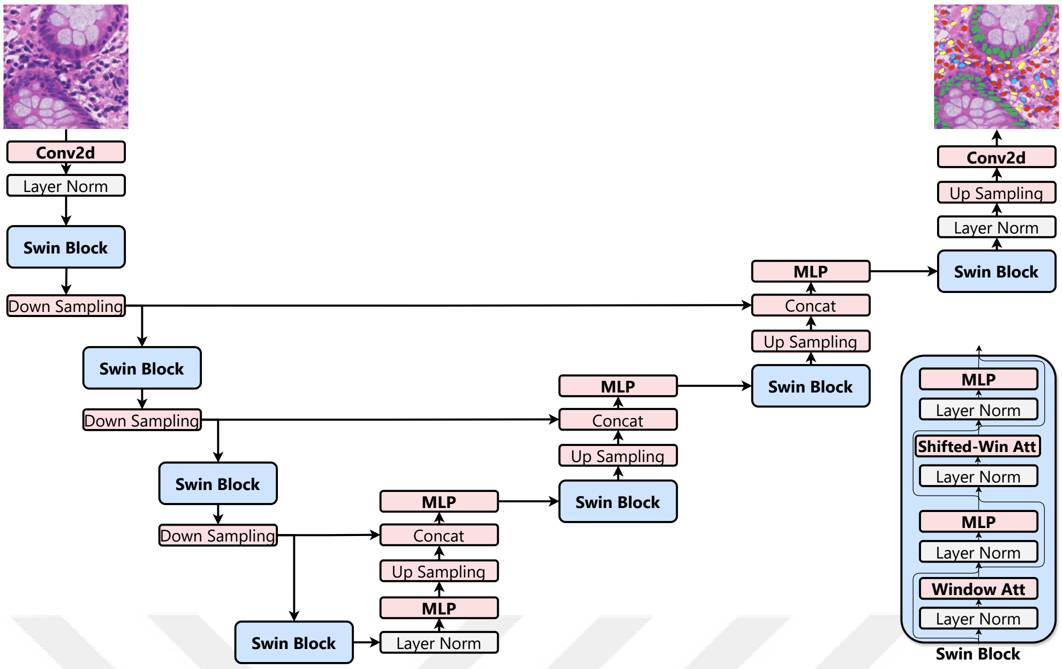


Figure 3.12 Swin-Unet architecture

performance compared to many well-established models in the literature. In the Swin-Unet model, traditional convolution blocks are replaced with Swin blocks, as proposed in the Swin Transformer architecture [90].

Let us define $SwinBlock(.)$ as the function representing a single block of the Swin Transformer model, where W-MSA$(.)$ and SW-MSA$(.)$ denote the window-based multi-head self-attention and shifted window-based multi-head self-attention, respectively, and $z_{l-1} \in R^{d \times h \times w}$ is the input feature map to layer l. The process of obtaining a output feature vector z_{l+1} for a given input vector $\hat{z}_{l-1}$ is formalized as follows:

$$z_{l+1} = SwinBlock(z_{l-1}) \tag{3.10}$$

$$\hat{z}_l = \text{W-MSA}(\text{LN}(z_{l-1})) + z_{l-1} \tag{3.11}$$

$$z_l = \text{MLP}(\text{LN}(\hat{z}_l)) + \hat{z}_l \tag{3.12}$$

$$\hat{z}_{l+1} = \text{SW-MSA}(\text{LN}(z_l)) + z_l \tag{3.13}$$

$$z_{l+1} = \text{MLP}(\text{LN}(\hat{z}_{l+1})) + \hat{z}_{l+1} \tag{3.14}$$

where LN$(.)$ denotes the layer normalization, MLP$(.)$ represents a multi-layer perceptron, and z_l, $_l$,z$_{l+1}$, $\hat{z}_{l+1}$ are intermediate features within the block.

The detailed architecture of the Swin-Unet model, which closely mirrors the UNet, is illustrated in Figure 3.12. Instead of the 16×16 patches typically used in vision transformer models [87], 4×4 convolution filters are applied to the input image to create smaller patches. This finer patch size allows for more precise results in tasks such as semantic segmentation. Instead of using max pooling to reduce the input dimension, the model shifts spatial vectors into the channel dimension. For the upsampling process, it employs MLPs to increase the size of the feature vectors.

3.5.1 Swin-Unet Experintal Setup

In order to ensure comparability, the experimental setup employed for Swin-Unet was adjusted to match that of TransUNet. Therefore, Swin-Unet was trained using the CoNIC and MoNuSeg datasets. We conducted 5-fold cross-validation for the CoNIC dataset. We further cross-tested the model, trained on the MoNuSeg dataset, using the CPM17 and CoNSeP datasets.

We used cross-entropy as a loss function. We utilized the AdamW optimizer and determined the batch size to be 16. We employed the Cyclic Learning Rate Scheduler, setting the maximum learning rate at 0.001. We trained the model using input sizes of 256×256 patches without any resizing. Testing was conducted on RoI. In total, the training involved 40,000 steps.

3.5.2 Swin-Unet Results

Table 3.20 displays the 5-fold cross-validation semantic segmentation results of the Swin-Unet model on the CoNIC dataset. For the background class, the model shows excellent semantic segmentation performance with a recall of 0.9690 and precision of 0.9603. However, performance suffers significantly when examining nuclei, particularly in the neutrophil class, which has extremely low recall and IoU values of 0.2402 and 0.2077, respectively. As shown in Figure 3.13, the Swin-Unet model effectively distinguishes between nuclei and background, but it struggles to accurately classify within the nuclei classes. Inevitably, inadequate semantic segmentation leads to issues related to instance segmentation.

Table 3.21 presents the 5-fold cross-validation instance segmentation results of the Swin-Unet model on the CoNIC dataset. Significant variability of 5% is indicated by the range of values between the lowest and greatest mPQ^{+}, which are 0.3672 and 0.4175, respectively. This level of fluctuation is significantly greater than that seen in the UNet and TransUNet models, where variability is approximately 0.5% and 1% respectively. Such findings highlight the Swin-Unet model's relative instability in nuclei classification, especially evident in the neutrophil class, which exhibits a 9% difference between folds 1 and 2.

Table 3.20 Swin-Unet semantic segmentation results on CoNIC fataset with adaptive thresholding

	Fold	μ	Bg	Neu	Epi	Lym	Pla	Eos	Con
Dice	**1**	0.6837	0.9664	0.4436	0.8142	0.7643	0.5428	0.5362	0.7182
	2	0.6494	0.9633	0.3099	0.8082	0.7497	0.5159	0.5105	0.6880
	3	0.6609	0.9659	0.2980	0.8117	0.7570	0.5424	0.5423	0.7088
	4	0.6382	0.9626	0.3246	0.7955	0.7200	0.5141	0.4850	0.6659
	5	0.6555	0.9649	0.3353	0.8116	0.7505	0.5228	0.5132	0.6901
	μ	0.6575	0.9646	0.3423	0.8082	0.7483	0.5276	0.5174	0.6942
IoU	**1**	0.5463	0.9349	0.2850	0.6866	0.6185	0.3725	0.3663	0.5603
	2	0.5150	0.9292	0.1834	0.6781	0.5997	0.3476	0.3428	0.5243
	3	0.5278	0.9340	0.1751	0.6831	0.6091	0.3721	0.3720	0.5489
	4	0.5014	0.9279	0.1937	0.6604	0.5625	0.3460	0.3201	0.4991
	5	0.5204	0.9322	0.2014	0.6830	0.6006	0.3539	0.3452	0.5269
	μ	0.5222	0.9316	0.2077	0.6783	0.5981	0.3584	0.3493	0.5319
Recall	**1**	0.6405	0.9697	0.3566	0.8159	0.7657	0.4525	0.4451	0.6776
	2	0.5956	0.9679	0.2068	0.8132	0.7329	0.4187	0.4053	0.6247
	3	0.6150	0.9695	0.1983	0.8197	0.7457	0.4530	0.4641	0.6548
	4	0.5840	0.9685	0.2171	0.8007	0.6858	0.4353	0.3883	0.5925
	5	0.6025	0.9692	0.2222	0.8203	0.7234	0.4384	0.4128	0.6308
	μ	0.6075	0.9690	0.2402	0.8140	0.7307	0.4396	0.4231	0.6361
Precision	**1**	0.7487	0.9631	0.5868	0.8125	0.7628	0.6780	0.6739	0.7639
	2	0.7535	0.9587	0.6186	0.8032	0.7674	0.6718	0.6895	0.7655
	3	0.7477	0.9623	0.5990	0.8039	0.7687	0.6756	0.6522	0.7726
	4	0.7401	0.9568	0.6423	0.7903	0.7578	0.6276	0.6456	0.7600
	5	0.7590	0.9607	0.6825	0.8032	0.7796	0.6474	0.6781	0.7618
	μ	0.7498	0.9603	0.6258	0.8026	0.7673	0.6601	0.6679	0.7647

Table 3.21 Swin-UNet instance segmentation results on CoNIC dataset with adaptive thresholding

Fold	mPQ	PQ^{+}_{Neu}	PQ^{+}_{Epi}	PQ^{+}_{Lym}	PQ^{+}_{Pla}	PQ^{+}_{Eos}	PQ^{+}_{Con}
1	0.4175	0.2720	0.3535	0.5866	0.4374	0.3465	0.5091
2	0.3797	0.1828	0.3177	0.5878	0.4057	0.3094	0.4746
3	0.3947	0.1723	0.3400	0.5827	0.4393	0.3358	0.4980
4	0.3672	0.2076	0.3171	0.5543	0.4017	0.2841	0.4381
5	0.3881	0.2135	0.3305	0.5782	0.4191	0.3132	0.4739
μ	0.3894	0.2096	0.3318	0.5779	0.4206	0.3178	0.4787

Figure 3.14 presents the instance segmentation results of the SwinUNet model on CoNIC dataset. One important issue affecting the model's performance is the inability to effectively identify the boundaries of nuclei in semantic segmentation. This error causes nuclei to join together, complicating the separation process. As objects with an IoU value above the 0.5 threshold are considered true positives, the presence of two or more merged nuclei enlarges the union area, thus reducing the IoU below 0.5. Consequently, these predictions are inevitably classified as false.

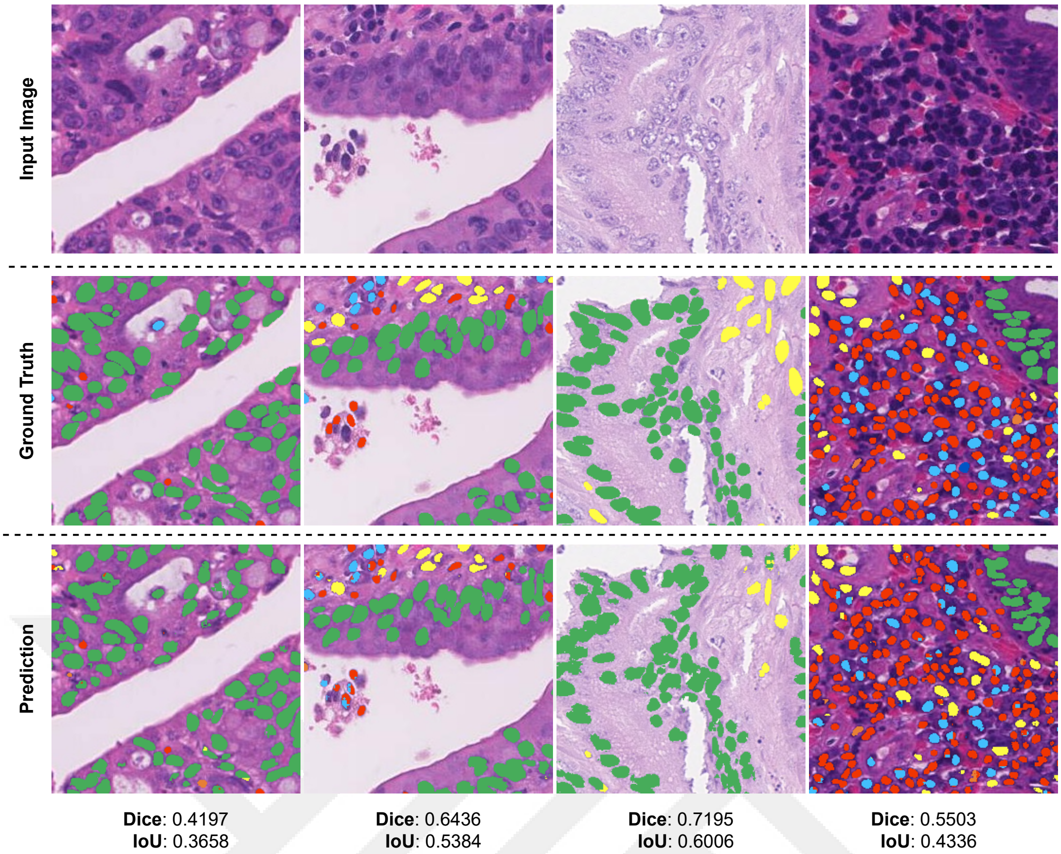


Figure 3.13 Swin-Unet semantic segmentation prediction on CoNIC patches

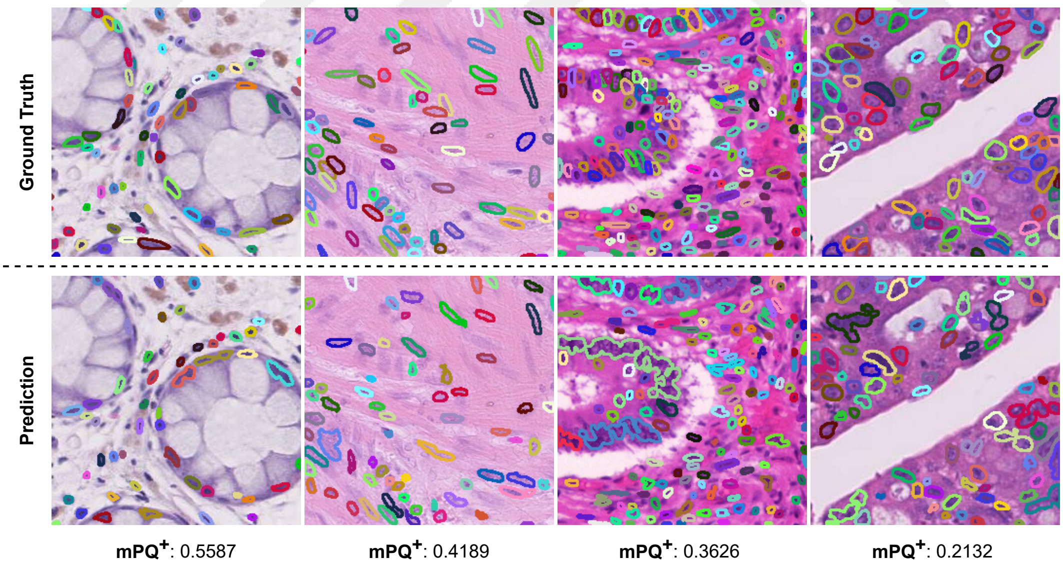


Figure 3.14 Swin UNet instance segmentation prediction on CoNIC patches

The semantic segmentation results of the Swin-UNet model, which is trained on the MoNuSeg dataset and tested in both in-domain and out-domain scenarios, are presented in Table 3.22. Notably, the model achieves higher Dice, IoU, and precision metrics on the CPM17 dataset compared to MoNuSeg. This performance, with a recall of 0.7013 for nuclei and 0.9871 for background on CPM17, shows that

Table 3.22 Semantic segmentation results Swin UNet trained on MoNuSeg with adaptive thresholding

	Test Dataset	μ	Background	Nuclei
Dice	**MoNuSeg**	0.8754	0.9468	0.8040
	CPM17	0.8788	0.9577	0.7999
	CoNSeP	0.8342	0.9370	0.7314
IoU	**MoNuSeg**	0.7856	0.8989	0.6723
	CPM17	0.7927	0.9189	0.6665
	CoNSeP	0.7290	0.8814	0.5766
Recall	**MoNuSeg**	0.8714	0.9507	0.7920
	CPM17	0.8442	0.9871	0.7013
	CoNSeP	0.8266	0.9438	0.7094
Precision	**MoNuSeg**	0.8796	0.9429	0.8164
	CPM17	0.9304	0.9301	0.9308
	CoNSeP	0.8425	0.9302	0.7548

Table 3.23 Instance segmentation results Swin UNet trained on MoNuSeg with adaptive thresholding

Test Dataset	PQ$^+$	PQ	SQ	DQ	AJI$^+$	AJI
MoNuSeg	0.5259	0.5262	0.7487	0.7020	0.5473	0.5139
CPM17	0.5335	0.5340	0.7568	0.7009	0.5750	0.5603
CoNSeP	0.2214	0.2206	0.7135	0.3091	0.2342	0.1435

the model tends to make background predictions when used in different domains, which correlates to the pixel distribution of nuclei to background in CPM17. In contrast, performance on the CoNSeP dataset remains low, similar to results from the UNet and TransUNet models.

The Swin-Unet model, which was trained on the MoNuSeg dataset, presents the instance segmentation results in Table 3.23. Consistent with previous observations, the model shows higher performance on the CPM17 dataset. Despite this, it exhibits notably poor DQ performance, particularly in out-domain CoNSeP dataset tests. The model underperforms on the CoNSeP datasets like the UNet and TransUnet, achieving a mere PQ^+ of 0.2214. This significantly low score on the CoNSeP dataset highlights the limitations of the fully transformer-based Swin-Unet model in adapting to various domains, indicating challenges with domain generalization.

3.6 Comparison of the UNet Architectures

In this section, we compare the performance of UNet, TransUnet, and Swin-Unet models across the CoNIC, MoNuSeg, CPM17, and CoNSeP datasets. The models' performance on the CoNIC dataset has been assessed using the PQ^+ metric, as

Table 3.24 Comparison of the model's instance segmentation performance on CoNIC dataset

Methodology	mPQ	PQ^+_{Neu}	PQ^+_{Epi}	PQ^+_{Lym}	PQ^+_{Pla}	PQ^+_{Eos}	PQ^+_{Con}
Swin UNet	0.2917	0.1175	0.2981	0.4763	0.3126	0.1733	0.3723
Swin UNet + AT	0.3894	0.2096	0.3318	0.5779	0.4206	0.3178	0.4787
Trans UNet	0.3910	0.2061	0.3680	0.5724	0.4284	0.2811	0.4903
Unet	0.4496	0.2917	0.3860	0.6134	0.4983	0.3551	0.5531
Trans UNet + AT	0.4625	0.2891	0.3867	0.6366	0.5179	0.3828	0.5618
UNet + AT	**0.5090**	**0.3776**	**0.3980**	**0.6566**	**0.5680**	**0.4472**	**0.6067**

Table 3.25 Comparison of the model's instance segmentation performance on MoNuSeg dataset

Methodology	PQ^+	PQ	SQ	DQ	AJI^+	AJI
Swin UNet	0.4623	0.4602	0.7487	0.6134	0.5462	0.5136
Trans UNet	0.4964	0.4916	0.7599	0.6455	0.5496	0.5137
Swin UNet + AT	0.5259	0.5262	0.7487	0.7020	0.5473	0.5139
Unet	0.5414	0.5392	0.7631	0.7056	0.5819	0.5530
Trans UNet + AT	0.5437	0.5381	0.7599	0.7071	0.5507	0.5143
UNet + AT	**0.5700**	**0.5701**	0.7631	**0.7464**	**0.5824**	**0.5532**

suggested in [64]. Furthermore, we present the outcomes with and without the implementation of the adaptive thresholding methodology, as proposed in Section 3.2, and provide a detailed analysis of its impact on the model outputs.

Table 3.24 lists the class-specific PQ^+ performance of models trained on the CoNIC dataset, ordered by their mPQ^+ values. Notably, The UNet model with adaptive thresholding outperforms its nearest rival by 4% in mPQ^+ and achieves highest scores across all classes. Moreover, the adaptive thresholding methodology has enhanced performance by 5-9% for each model. It can be concluded that the classical UNet model, especially with adaptive thresholding, is the most successful in the nuclei classification task.

Table 3.25 compares model performances on the MoNuSeg dataset using metrics such as AJI, AJI^+, DQ, SQ, and PQ^+. The models are ranked by their PQ^+ scores, which are consistent with the rankings obtained on the CoNIC dataset. The UNet model achieved the highest success with a PQ^+ of 0.5700, while the SwinUNet model exhibited the lowest, with a PQ^+ of 0.4623.

The SQ metric, calculated for true positive predictions, shows remarkably consistent performance across all models, indicating that when models correctly predict instances, they consistently predict edges well. Additionally, the SQ values remain unchanged whether adaptive thresholding is applied or not. This consistency

Table 3.26 Comparison of the model's instance segmentation performance on CMP17 dataset

Methodology	PQ$^+$	PQ	SQ	DQ	AJI$^+$	AJI
Swin UNet	0.4495	0.4544	0.7563	0.5960	0.5747	0.5604
Trans UNet	0.4759	0.4673	0.7702	0.6013	0.5596	0.5227
Unet	0.5301	0.5282	0.7765	0.6758	**0.5899**	**0.5782**
Swin UNet + AT	0.5335	0.5340	0.7568	0.7009	0.5750	0.5603
Trans UNet + AT	0.5358	0.5244	0.7704	0.6756	0.5605	0.5232
UNet + AT	**0.5602**	**0.5567**	0.7765	**0.7126**	0.5896	0.5777

Table 3.27 Comparison of the model's instance segmentation performance on CoNSeP dataset

Methodology	PQ$^+$	PQ	SQ	DQ	AJI$^+$	AJI
Swin UNet	0.1677	0.1626	0.7130	0.2270	0.2347	0.1445
Trans UNet	0.1972	0.1926	0.7327	0.2612	0.2618	0.1916
Swin UNet + AT	0.2214	0.2206	0.7135	0.3091	0.2342	0.1435
Trans UNet + AT	0.2327	0.2282	0.7332	0.3098	0.2609	0.1900
Unet	0.2608	0.2497	0.7448	0.3334	**0.2978**	**0.2120**
UNet + AT	**0.2820**	**0.2721**	**0.7449**	**0.3637**	0.2972	0.2111

confirms that the adaptive thresholding methodology does not eliminate any true positives but enhances the DQ performance by reducing false positives.

Table 3.26 presents the results obtained on the CPM17 dataset, where the UNet model with adaptive thresholding achieved the highest performance with a PQ^+ of 0.5602. However, for the AJI, and AJI^+ metrics, the UNet model without adaptive thresholding emerges as the most effective. In the PQ metric, only predictions with an IoU of 0.5 or higher against the ground truth are counted as true positives. Conversely, for AJI, and AJI^+, any non-zero intersection contributes to the metric. The fact that the UNet without adaptive thresholding scores highest on these metrics indicates that it retains predictions with even minimal overlap with the ground truth.

Table 3.27 displays the results of models trained on the MoNuSeg dataset and evaluated on the CoNSeP dataset. The UNet model with adaptive thresholding achieved the highest performance, with a PQ^+ of 0.2820. Although the thresholding strategy eliminated some true positives, it significantly improved the DQ performance and led to the model's optimal PQ score. Similarly, as observed with the CPM17 dataset, the UNet without the adaptive thresholding model also scored highest in the AJI, and AJI^+ metrics on the CoNSeP dataset.

The requirement for substantial data to effectively capture spatial features is a well-known limitation of transformer architectures [94–96]. This constraint is especially problematic in nuclei instance segmentation, where dataset availability is limited. Therefore, transformer models might not be the most suitable choice for this specific task. Furthermore, all models' performance on the out-domain CoNSeP dataset is significantly below acceptable levels. Therefore, obtaining definitive conclusions about the generalization capabilities of these models is hard.

4
GENERAL-PURPOSE MODELS ADAPTED FOR NUCLEI INSTANCE SEGMENTATION

General-purpose instance segmentation models are specialized versions of general-purpose object detection methods [97]. Initially, these models generate a bounding box prediction to define the boundaries of the detected object. The widely accepted architectures in the literature consist of three main components: the backbone for feature extraction, the neck for combining features, and the head for producing predictions. The backbone is designed to extract feature vectors, the neck combines coarse and fine feature vectors to enhance feature representation, and the head component is responsible for producing the final predictions.

Mask R-CNN [98] is one of the most well-established and well-known instance segmentation models, with applications across various fields [99–101]. It extends the Faster R-CNN [102], a two-stage object detection model, to enable instance segmentation. In contrast, the YOLOv7 instance segmentation model [103] is derived from one of the latest object detection models, YOLOv7, which is a one-stage detection model. The architectural differences between the one-stage and two-stage models not only affect their prediction performance but also influence their inference speeds [104].

In this chapter, we assess the nuclei instance segmentation performance of general-purpose models such as Mask R-CNN and YOLOv7. Although these models were not specifically designed for medical datasets, we analyzed their performance on medical images. The analysis aims to identify each model's strengths and weaknesses in handling specialized medical imaging tasks.

4.1 Mask R-CNN

Mask R-CNN is one of the most popular instance segmentation models in the literature, with applications across various fields. It builds on the Faster R-CNN

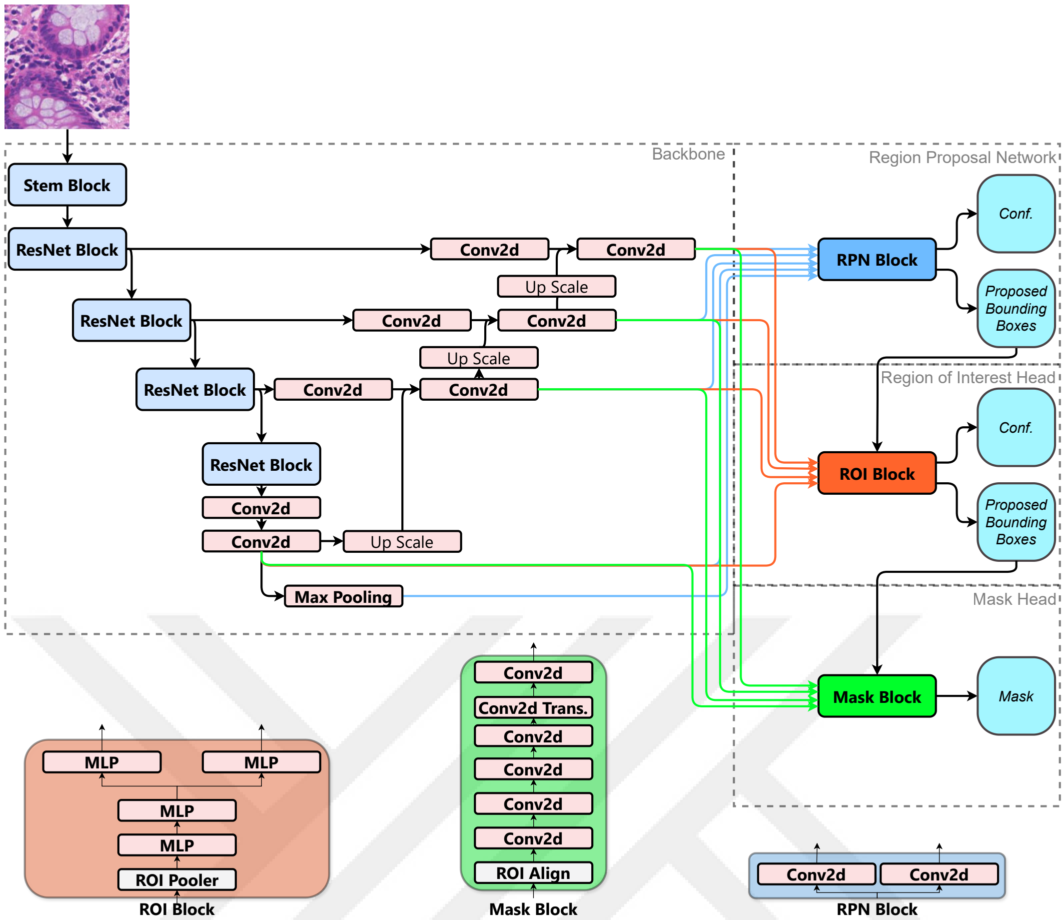


Figure 4.1 Mask R-CNN architecture

architecture [105] by including a mask head that generates masks for predicted bounding boxes. With the addition of an extra head to its two-stage object detection framework, Mask R-CNN is slower than many of the instance segmentation models in the literature [104].

Mask R-CNN architecture is illustrated in Figure 4.1. The backbone architecture of Mask R-CNN is noticeably similar to that of UNet. The process begins with the stem block, which applies sequential convolution filters to the input image. Subsequently, it utilizes ResNet blocks to generate outputs at four different levels, with each output half the size of the previous one, similar to the down scaling in UNet. The final output is further processed using the max pooling layer to serve only as input for the region proposal network, while outputs from the other levels are sent to three different heads.

These proposals are then filtered based on IoU and confidence scores before proceeding to the next stage. The Region Proposal Network (RPN) in Mask R-CNN is used to generate proposed bounding boxes. Its primary function is to identify the presence of objects within a specific area. To detect objects of different sizes, the

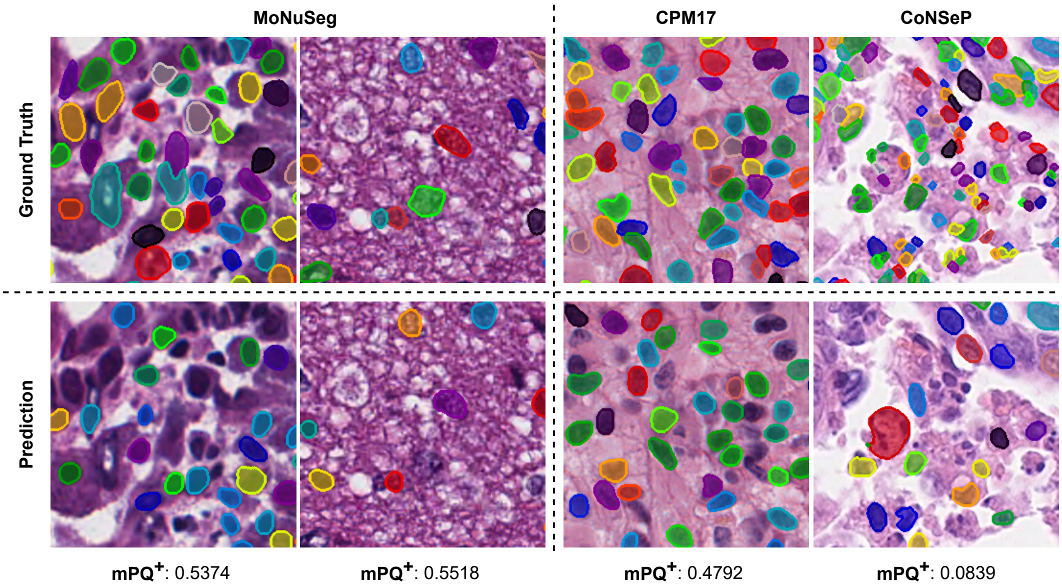


Figure 4.2 Mask R-CNN model prediction

RPN applies convolution filters to feature vectors obtained at each level, producing proposed bounding boxes and confidence scores.

In the second stage, the Region of Interest (RoI) head takes the proposed bounding boxes and feature vectors extracted from each level of the backbone and applies a pooling operation. This pooling operation transforms the feature vectors, which are then fed into a series of multi-layer perceptrons (MLP). These MLPs refine the proposed bounding boxes to generate more precise final bounding boxes. This detailed and precise refinement process, especially pooling, is a significant factor in the model's slower inference speed.

After the bounding boxes are predicted, the final step is to create masks for the detected objects. This is accomplished by feeding predicted bounding boxes and feature vectors from each level of the backbone into the mask head. In the mask head, following the ROI Align operation, a series of convolution filters are applied to produce binary object masks.

The Mask R-CNN model, designed for general use, does not specifically address the issue of occlusions between objects, so it does not enforce separation between overlapping object masks. However, in histopathology images, occlusion is typically absent. This is a post-processing issue because the masks produced by Mask R-CNN may overlap, which does not occur in histopathology images. Therefore, an instance segmentation map has been derived based on the highest confidence scores.

Table 4.1 Instance segmentation results of Mask R-CNN

Test Dataset	PQ$^+$	PQ	SQ	DQ	AJI$^+$	AJI
MoNuSeg	0.4882	0.4936	0.7333	0.6727	0.4344	0.4304
CPM17	0.5235	0.5104	0.7145	0.7080	0.5015	0.4919
CoNSeP	0.2635	0.2764	0.7498	0.3682	0.2478	0.2431

Table 4.2 Semantic segmentation results of Mask R-CNN

	Test Dataset	μ	Background	Nuclei
Dice	**MoNuSeg**	0.7648	0.9238	0.6058
	CPM17	0.8122	0.9399	0.6846
	CoNSeP	0.6664	0.9147	0.4181
IoU	**MoNuSeg**	0.6465	0.8584	0.4346
	CPM17	0.7035	0.8865	0.5205
	CoNSeP	0.5536	0.8429	0.2643
Recall	**MoNuSeg**	0.7206	0.9884	0.4528
	CPM17	0.7683	0.9852	0.5514
	CoNSeP	0.6325	0.9923	0.2727
Precision	**MoNuSeg**	0.8912	0.8671	0.9152
	CPM17	0.9005	0.8985	0.9026
	CoNSeP	0.8724	0.8484	0.8964

4.1.1 Mask R-CNN Experimental Setup

The Mask R-CNN model was trained on the MoNuSeg dataset and tested on the MoNuSeg, CPM17, and CoNSeP datasets. A ResNet-50 backbone [106], initialized with ImageNet [107] weights, was employed due to its widespread popularity and proven effectiveness. The training adhered closely to the standard Mask R-CNN parameters, using a specific optimizer and learning rate for a predefined number of steps. The model's input size was set at 1000x1000 pixels, and the testing was conducted on RoI. No scale augmentation was performed during training, and test time augmentation was not included during inference.

4.1.2 Mask R-CNN Results

Table 4.1 presents the semantic segmentation results for the Mask R-CNN model trained on the MoNuSeg dataset. The model's performance on nuclei is particularly weak. Although the recall metric shows very high scores for the background class, it is markedly low for nuclei. Despite this, the model demonstrates high precision, indicating that while its detection capabilities are limited, it performs well in segmenting the nuclei instances it does detect.

Table 4.2 provides the instance segmentation results for the Mask R-CNN model trained on the MoNuSeg dataset. Consistent with the semantic segmentation findings, the instance segmentation performance is also relatively low. However, the model has achieved approximately 4% better results on the CPM17 dataset compared to the MoNuSeg dataset. Model predictions are illustrated in Figure 4.2. From these visual representations and the performance metrics, it becomes clear that while the model's detection capabilities are somewhat limited, its segmentation accuracy is notably higher.

4.2 YOLOv7 Instance Segmentation

YOLO architectures are known for their high efficiency and robust performance in object localization [108]. Unlike the slower two-stage object detection architectures like Faster R-CNN, YOLO successfully predicts the position and class of target objects in one stage without involving any pooling operations. The typical YOLO object detection architecture consists of three main components: the backbone, which extracts feature vectors; the neck, which manages transitions between different levels in a similar way that the UNet architecture does; and the head, which generates predictions.

One of the key methodological enhancements that have significantly improved the performance of YOLO architectures involves the effectively use of anchor boxes [109]. These predefined bounding boxes serve as references for predicting the extent to which the centers of detected objects align with expected dimensions and positions. The anchor boxes are optimized during the training process through an auto-anchor algorithm, which aims to achieve the maximum possible recall [109]. Once established, these boxes are used as constant parameters in the model to ensure consistency during the training and testing phases.

In recent studies, the YOLOv7 model stands out as one of the most commonly used object detection models within the YOLO family. It has demonstrated its effectiveness across various domains. For instance, one study employs it to address a specific problem in detection of gastric lesions [110], while another applies it to solve issues in underwater target detection[111]. These examples demonstrate the broad applicability of the YOLOv7 architecture. In this study, we extend its utility by applying the YOLOv7 instance segmentation model to nuclei instance segmentation tasks.

The detailed architecture of the YOLOv7 instance segmentation model is presented in Figure 4.3, consisting of three main components: the backbone, neck, and head. The backbone, designed for extracting feature vectors, includes Convolution Blocks

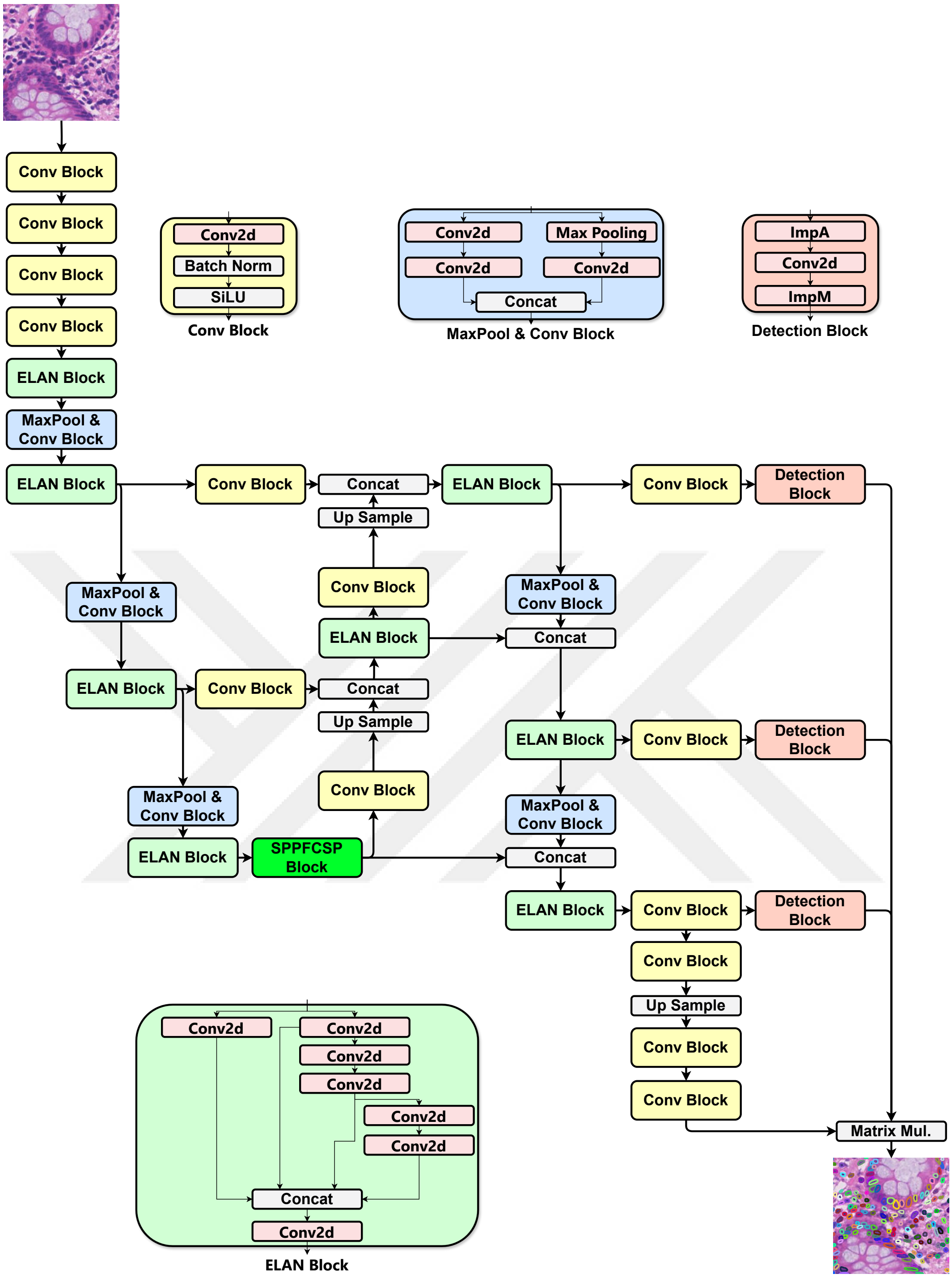


Figure 4.3 YOLOv7 instance segmentation model architecture

(Conv Block), Efficient Layer Aggregation Networks (ELAN) blocks [112], and Max Pooling & Convolution Blocks (MaxPool & Conv Block). The Convolution Block includes a 3×3 convolution filter, batch normalization layer, and SiLU (sigmoid linear unit) activation function. The MaxPool & Conv Block is responsible for reducing the input size by half, achieving this not only through max pooling but also by applying 2×2 stride convolution filters and concatenating the two resultant feature vectors. While max pooling enhances the model's robustness to minor variations, it can negatively impact gradient flow by only transmitting the maximum values to the next layer [113, 114]. The combination of convolution blocks and max pooling processes effectively halves the input dimensions and preserves gradient flow, enhancing the model's overall robustness [112].

The Efficient Layer Aggregation Network (ELAN) block is a pivotal feature in the YOLOv7 architecture, developed with a focus on optimizing gradient flow [112]. The ELAN block differs from the ResNet model's approach of simple addition between the input and output vectors. It applies a 1x1 convolution on the input, which is essential for effectively managing gradient transitions, and then a concatenation method similar to DenseNet [115] to merge feature vectors. This configuration not only ensures efficient training, but it also improves the model's ability to extract detailed feature vectors, unlike traditional residual connections.

The neck architecture in YOLOv7, similar to UNet, combines coarse and fine feature vectors. It processes feature vectors extracted from three levels of the backbone using convolution blocks and efficient layer aggregation network (ELAN) blocks to produce outputs at various scales, specifically 1/8, 1/16, and 1/32 of the original input size. These outputs are then fed into the detection block, where each feature vector is added to a learnable parameter vector (ImpA). A convolution filter is applied next, and the output is multiplied by another learnable parameter vector (ImpM) to generate the final detection boxes.

For object mask generation in YOLOv7, feature vectors from the highest level of the neck are processed through additional convolution blocks and upsampling steps. These enhanced feature vectors are then multiplied with outputs from the detection block to create masks. The predicted bounding boxes constrain these masks, finalizing the instance segmentation output.

4.2.1 YOLOv7 Experimental Setup

The YOLOv7 model was trained for the instance segmentation task using the MoNuSeg dataset. Training involved 300 epochs with 256x256 patches, utilizing the SGD optimizer. The initial learning rate was set at 0.01, and no learning rate scheduler was employed. To avoid reducing performance, scale augmentation was not applied. However, flip up-down and left-right augmentations were set at a 50% probability. The batch size was established at 16, and a gradient accumulation parameter of 4 was utilized. No pretrained models were used during the training phase.

Cross-testing was conducted on the CPM17 and CoNSeP datasets to provide a more

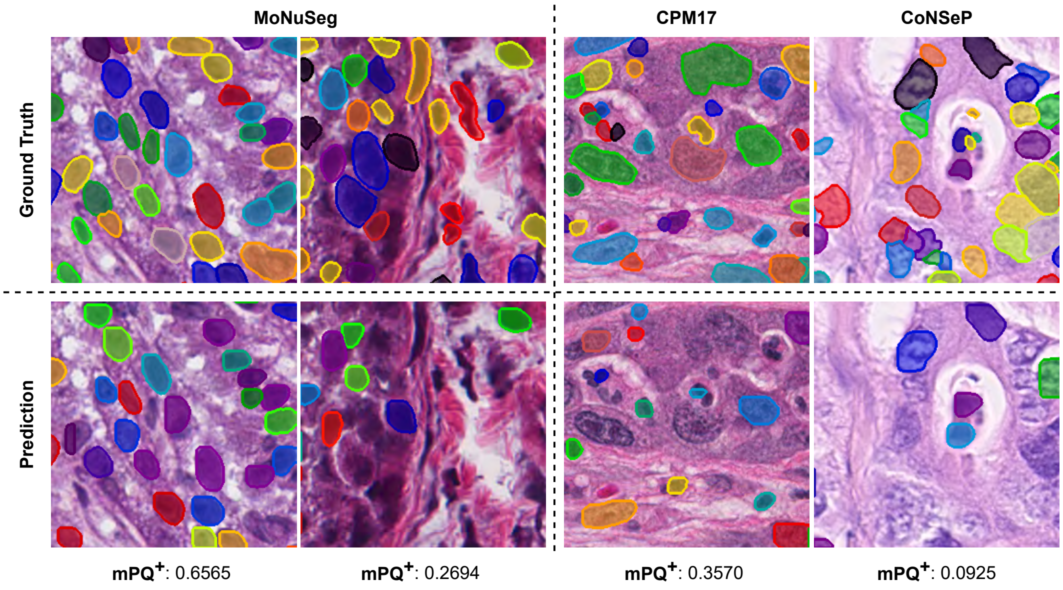


Figure 4.4 YOLOv7 model prediction

Table 4.3 Instance segmentation results of YOLOv7

Test Dataset	PQ^+	PQ	SQ	DQ	AJI^+	AJI
MoNuSeg	0.5698	0.5548	0.7522	0.7363	0.5284	0.5190
CPM17	0.4762	0.4580	0.7389	0.6178	0.4167	0.4110
CoNSeP	0.1992	0.1860	0.7270	0.2561	0.1655	0.1639

accurate assessment of the YOLOv7 model's performance. Although the model was initially trained on image patches, the testing phase was conducted on RoI. Patches of size 256x256 were extracted from these RoI and used as inputs in the model. The segmented results were then compiled to produce comprehensive RoI predictions.

The YOLOv7 instance segmentation model inherently allows a pixel to be assigned to multiple instance masks. However, this approach is not suitable for nuclei instance segmentation tasks. To enhance performance and ensure more accurate measurements, a confidence score-based approach has been developed. In this approach, the mask with the highest confidence score is placed on the RoI first. Subsequent predictions are inserted only if a certain percentage of the area remains unassigned. Additionally, similar to the adaptive thresholding approach, predictions smaller than 40 pixels are excluded. The confidence threshold has been set at 0.7, with all hyperparameters empirically determined to optimize results.

4.2.2 YOLOv7 Results

Table 4.3 presents the performance results of the YOLOv7 instance segmentation model trained on the MoNuSeg dataset, evaluated across the MoNuSeg, CPM17,

Table 4.4 Semantic segmentation results of YOLOv7

	Test Dataset	μ	Background	Nuclei
Dice	**MoNuSeg**	0.8240	0.9357	0.7123
	CPM17	0.7650	0.9315	0.5984
	CoNSeP	0.6099	0.9055	0.3142
IoU	**MoNuSeg**	0.7162	0.8792	0.5531
	CPM17	0.6494	0.8718	0.4270
	CoNSeP	0.5069	0.8274	0.1864
Recall	**MoNuSeg**	0.7882	0.9766	0.5998
	CPM17	0.7159	0.9933	0.4385
	CoNSeP	0.5920	0.9899	0.1941
Precision	**MoNuSeg**	0.8874	0.8981	0.8766
	CPM17	0.9094	0.8770	0.9418
	CoNSeP	0.8291	0.8344	0.8238

and CoNSeP datasets. The model achieved a PQ^+ of 0.5698 on MoNuSeg, with DQ and SQ scores of 0.7363 and 0.7522, respectively. However, it is important to note that these levels of performance were not observed on the other datasets.

On the CPM17 dataset, the YOLOv7 model obtained a PQ^+ of 0.4762, with minor DQ and SQ scores of 0.6178 and 0.7389, respectively. This indicates the model's limited capability in adapting to different domains. A similar results was observed on the CoNSeP dataset, where the model obtained a DQ of 0.2561 and an SQ of 0.7270. Notably, the SQ scores remained relatively stable across domains, whereas the DQ scores declined, suggesting that while the model maintains segmentation quality, its detection accuracy decreases. Additionally, as illustrated in Figure 4.4, the issues with missed or incorrect detections by YOLOv7 are evident both from the model predictions and the decrease in DQ. A major cause of to these inconsistencies is the use of a training domain-specific confidence threshold, which may not be ideal for other domains, reducing the model's performance.

Table 4.4 presents the semantic segmentation results. Compared to UNet, which uses watershed techniques to transition from semantic to instance segmentation, the YOLOv7 instance segmentation model performs significantly worse in semantic segmentation. This reduced performance is expected due to the model's strategy of eliminating non-instance objects from the segmentation mask, which accidentally misses out correctly identified nuclei, although with a small overall impact.

5 YOLO-U: ENSEMBLE OF THE YOLOV7 AND UNET

Instance segmentation combines the techniques of object detection and semantic segmentation. Object detection is the process of determining the precise location and size of objects in an image, while semantic segmentation is the task of assigning each pixel to a specific class. By combining these two approaches, instance segmentation aims to not only locate objects within an image but also to understand the boundaries of each object. Due to the complexity of the instance segmentation task, an extensive dataset is required to achieve accurate results [116]. Nonetheless, obtaining sufficient datasets is a significant challenge, especially in the field of medical applications, and a lack of comprehensive datasets frequently limits the development of robust instance segmentation models [117].

The UNet model, known for its robustness in semantic segmentation tasks, produces precise segmentation results, particularly nuclei borders. Despite its strengths, UNet struggles to distinguish between closely located nuclei (see Figure 3.3). In contrast, the YOLOv7 instance segmentation model performs better at separating close nuclei. However, YOLOv7 does not produce the same precise segmentation masks as UNet. Furthermore, optimizing models such as YOLOv7 for object detection and instance segmentation requires finetuning of numerous threshold hyperparameters in order to accurately differentiate objects.

In this chapter, we introduce an ensemble methodology combining the segmentation capabilities of UNet with the center prediction performance of YOLOv7, referred to as YOLO-U. This approach leverages the strength of YOLOv7 in distinguishing individual nuclei and the robust semantic segmentation performance of UNet. By integrating these capabilities, we propose a confidence threshold-free model for nuclei instance segmentation that optimizes accuracy and efficiency.

5.1 Methodology

Considering the challenges presented by UNet and YOLOv7, we propose YOLO-U, a novel ensemble model that combines the best features of both to enhance performance. The proposed approach of dividing the instance segmentation task into object proposals and semantic segmentation models provides more successful results with fewer data samples, thanks to a reduction in problem complexity. This methodology takes advantage of UNet's accuracy in defining object boundaries and YOLOv7's ability to distinguish near objects. Furthermore, the ensemble methodology eliminates the necessity for manual confidence threshold hyperparameter adjustments in YOLO-U.

Figure 5.1 shows an overview of the YOLO-U architecture, which is divided into three main branches: semantic segmentation, instance segmentation, and decision combination. The instance segmentation branch is specifically designed to improve the accuracy of object center predictions. This is accomplished in an initial step by generating a distance map for all pixels across proposed object masks without applying a confidence threshold. In order to determine the distance value for a given pixel, the shortest distance to the nearest background pixel is computed. Subsequently, the centers of the proposed objects can be determined by identifying the optimal local maxima on the distance map.

Let us define $YOLO_{Ins}(.)$ as the function representing the YOLOv7 instance segmentation model, and $Dist(.)$ denotes the distance function used to calculate the pixel's shortest distance to the nearest background pixel. The process of locating the center of an object is formalized as follows:

$$YOLO_{Ins}(X) = \{(\mathbf{\hat{x}_i}, \mathbf{\hat{y}_i}, \mathbf{\hat{w}_i}, \mathbf{\hat{h}_i}, \mathbf{\hat{m}_i})\}_{i=1}^{N_p} \tag{5.1}$$

where $\mathbf{\hat{x}_i}$, $\mathbf{\hat{y}_i}$, $\mathbf{\hat{w}i}$, and $\mathbf{\hat{h}i}$ represent the bounding box's x-coordinate, y-coordinate, width, and height, respectively, while $\mathbf{\hat{m}i}$ denotes the mask of the ith proposed object. The input image is denoted by $X \in R^{3\times h\times w}$, and N_p represents the number of proposed objects. The distance map D for the input image X is derived using $\mathbf{\hat{m}i}$:

$$D = Dist(\{\mathbf{\hat{m}_i}\}_{i=1}^{N_p}) \tag{5.2}$$

where $D \in R^{h\times w}$ is same size with X. D indicates of the object centers' probability distribution, and D_i refers to the distance map corresponding to the i^{th} proposed object. Figure 5.2 illustrates the relation between the center and the distance map.

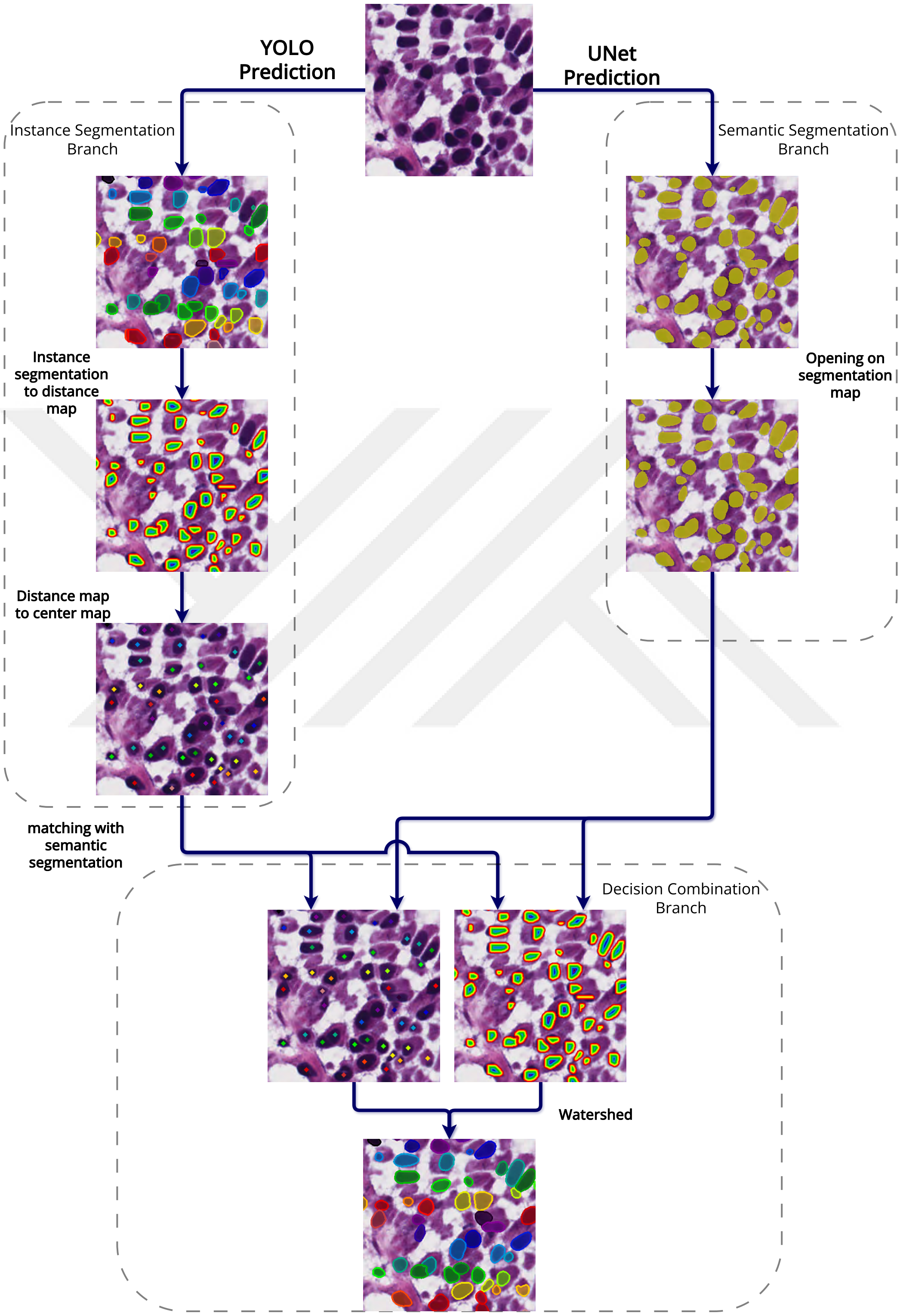


Figure 5.1 YOLO-U architecture

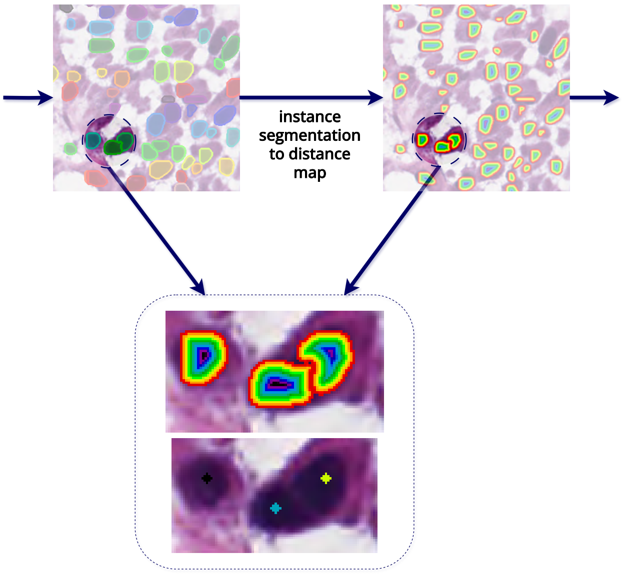


Figure 5.2 Distance and center map

The center map C is defined as:

$$(x_i, y_i) = \underset{x_i, y_i}{\arg\max}(D_i) \quad \text{for} \quad i \in \{1, \ldots, N_p\} \tag{5.3}$$

$$C[x_i, y_i] = i \tag{5.4}$$

where the coordinates (x_i, y_i) represent the center of the i^{th} proposed object having the maximum distance in $D[x_i, y_i]$. Considering there are multiple potential center points, our method proposes determining a singular center point by taking the median of all candidate points. Figure 5.3 shows a 3D visualization of the distance map alongside the center points, demonstrating the effectiveness of our methodology in separating closely spaced nuclei. It is noteworthy that the confidence threshold-free approach may result in numerous false positives regarding proposed objects. To address this issue, the semantic segmentation branch is used.

The semantic segmentation branch aims to define the boundaries of proposed objects and reduce the number of false positives. This branch is primarily based on UNet segmentation results, which are assumed to be mostly accurate, despite their inability to differentiate closely situated nuclei. However, these results frequently contain artifacts, especially when models are trained with small amounts of data. To address these issues, an opening operation is applied to the segmentation map, resulting in a cleaner and more refined output.

The decision combination branch combines the semantic segmentation, distance map, and center map results. If a proposed object from the instance segmentation branch has no significant overlap with the semantic map, it is excluded from further

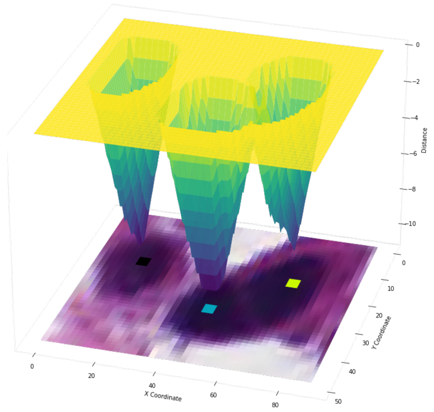


Figure 5.3 Distance and center map 3D

processing in this branch. Let $S \in R^{2 \times h \times w}$ is the semantic segmentation map, and the final distance map $\mathbf{D}$ is adjusted as follows:

$$\mathbf{D}_i = \begin{cases} D_i & intersect(D_i, S) \geq \mathtt{th} \\ 0 & intersect(D_i, S) < \mathtt{th} \end{cases} \quad \text{for} \quad i \in \{1, \ldots, N_p\} \quad \text{and} \quad 0 < \mathtt{th} < 1 \tag{5.5}$$

The function $intersect(.)$ calculates the ratio of intersection over the area of D_i with $\mathtt{th}$ being a threshold close to 1. The center map $\mathbf{C}$ is processed in a similar manner. The instance segmentation model does not always predict the blobs identified by semantic segmentation. these blobs are also included in $\mathbf{D}$ and $\mathbf{C}$. The ensemble model result is then refined using the watershed algorithm:

$$\mathbf{P} = \text{Watershed}(S, \mathbf{D}, \mathbf{C}) \tag{5.6}$$

where $\mathbf{P} \in R^{h \times w}$ denotes the final instance segmentation map, including object ID masks. The complete architecture can be summarized as follows:

$$\mathbf{P} = \text{YOLO-U}(X) = Ens(YOLO_{Ins}(X), UNet(X)) \tag{5.7}$$

Ens denotes proposed ensemble strategy, combining the instance segmentation model with the semantic segmentation model to yield a more successful and confidence threshold-free model.

5.2 Examination of Confidence Threshold

The YOLOv7 model generates object proposals, and only those aligned with UNet predictions are included, effectively eliminating low-confidence false positives. Figure 5.4 illustrates the performance of the YOLOv7 model across various

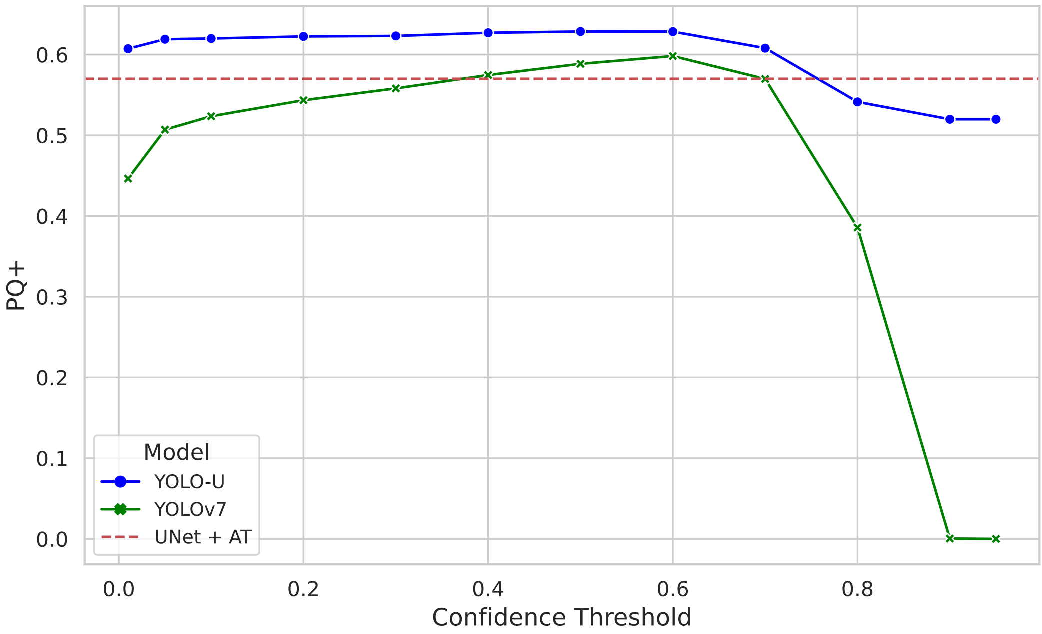


Figure 5.4 Examination of confidence threshold on MoNuSeg

confidence thresholds, ranging from 0.01 to 0.95. The YOLOv7 model achieves peak performance at a confidence threshold of 0.6. However, the performance dramatically decreases at thresholds above 0.7. Additionally, thresholds below 0.4 result in lower performance compared to the UNet with adaptive thresholding methodology.

The performance of the proposed YOLO-U model is remarkably stable across different confidence threshold settings. It maintains this stability up to the point where the YOLOv7 component achieves its maximum number of true positives. As the count of true positives decreases, the performance gradually aligns with that of the UNet model's instance segmentation results. This clearly shows that our ensemble methodology not only improves success but also performs efficiently without relying on confidence thresholds.

5.3 Results

Table 5.1 compares the YOLO-U model's instance segmentation results to those of all previously developed models. With a PQ^+ of 0.6199 on the MoNuSeg dataset, 0.5852 on the CMP17 dataset, and 0.3674 on the CoNSeP dataset, YOLO-U has emerged as the most successful model. It outperforms its closest competitor, the UNet with adaptive thresholding, by margins of 4.99% on MoNuSeg, 2.5% on CMP17, and 8.54% on CoNSeP. This superior performance is attributable to the ensemble methodology, which not only preserves the SQ performance of the

Table 5.1 Instance segmentation results of YOLO-U

	Model	PQ^+	PQ	SQ	DQ	AJI^+	AJI
MoNuSeg	SwinUNet	0.4623	0.4602	0.7487	0.6134	0.5462	0.5136
	Mask-RCNN	0.4882	0.4936	0.7333	0.6727	0.4344	0.4304
	TransUNet	0.4964	0.4916	0.7599	0.6455	0.5496	0.5137
	SwinUNet + AT	0.5259	0.5262	0.7487	0.7020	0.5473	0.5139
	UNet	0.5414	0.5392	**0.7631**	0.7056	0.5819	0.5530
	TransUNet + AT	0.5437	0.5381	0.7599	0.7071	0.5507	0.5143
	YOLOv7	0.5698	0.5548	0.7522	0.7363	0.5284	0.5190
	UNet + AT	0.5700	0.5701	**0.7631**	0.7464	0.5824	0.5532
	YOLO-U	**0.6199**	**0.6121**	0.7608	**0.8037**	**0.6291**	**0.6215**
CPM17	SwinUNet	0.4495	0.4544	0.7563	0.5960	0.5747	0.5604
	TransUNet	0.4759	0.4673	0.7702	0.6013	0.5596	0.5227
	YOLOv7	0.4762	0.4580	0.7389	0.6178	0.4167	0.4110
	Mask-RCNN	0.5235	0.5104	0.7145	0.7080	0.5015	0.4919
	SwinUNet + AT	0.5335	0.5340	0.7568	0.7009	0.5750	0.5603
	UNet	0.5301	0.5282	**0.7765**	0.6758	0.5899	0.5782
	TransUNet + AT	0.5358	0.5244	0.7704	0.6756	0.5605	0.5232
	UNet + AT	0.5602	0.5567	**0.7765**	0.7126	0.5896	0.5777
	YOLO-U	**0.5852**	**0.5787**	0.7722	**0.7464**	**0.6078**	**0.6018**
CoNSeP	SwinUNet	0.1677	0.1626	0.7130	0.2270	0.2347	0.1445
	TransUNet	0.1972	0.1926	0.7327	0.2612	0.2618	0.1916
	YOLOv7	0.1992	0.1860	0.7270	0.2561	0.1655	0.1639
	SwinUNet + AT	0.2214	0.2206	0.7135	0.3091	0.2342	0.1435
	TransUNet + AT	0.2327	0.2282	0.7332	0.3098	0.2609	0.1900
	UNet	0.2608	0.2497	0.7448	0.3334	0.2978	0.2120
	Mask-RCNN	0.2635	0.2764	**0.7498**	0.3682	0.2478	0.2431
	UNet + AT	0.2820	0.2721	0.7449	0.3637	0.2972	0.2111
	YOLO-U	**0.3674**	**0.3526**	0.7341	**0.4773**	**0.4024**	**0.3623**

UNet but also significantly boosts the DQ performance, thanks to its confidence threshold-free approach.

Particularly in the CPM17 and CoNSeP datasets, which are different from the training dataset, the YOLOv7 model exhibits DQ performances of 0.6178 and 0.2561, respectively. However, the YOLO-U ensemble methodology, leveraging YOLOv7 for nuclei center prediction, significantly improves performances to 0.7464 and 0.4773 DQ on CPM17 and CoNSeP, respectively. This significant improvement is made possible by the confidence threshold-free approach. The confidence threshold determined empirically for the MoNuSeg dataset domain does not suit other domains, reducing the number of true positives detected by the YOLOv7 model. As shown in Figure 5.5, the YOLO-U model successfully segments and distinguishes instances that UNet can only segment, and that

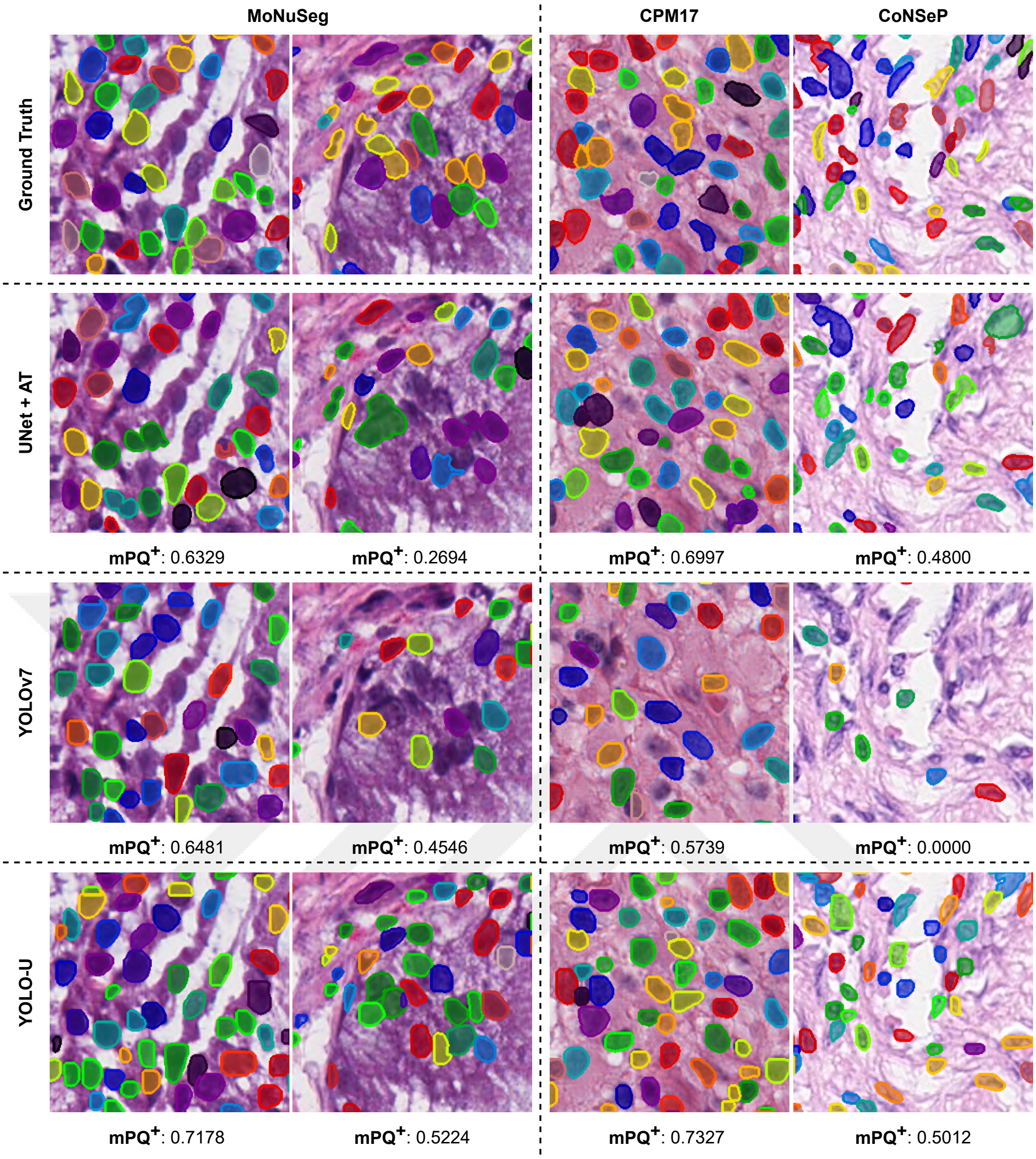


Figure 5.5 YOLO-U model prediction

YOLOv7 entirely misses.

Due to its superior semantic segmentation performance, UNet is employed within the YOLO-U model. An opening filter is applied to the UNet semantic segmentation output in the YOLO-U model to discard instances that do not match the ground truth by at least 50%. This process, similar to the adaptive thresholding method, enhances instance segmentation performance. However, it is true that some correctly classified pixels are also removed. The semantic segmentation results are presented in Table 5.2, demonstrating the expected effectiveness of the UNet model in semantic segmentation tasks.

Table 5.2 Mean semantic segmentation results of YOLO-U

	Model	μ IoU	μ Dice	μ Precision	μ Recall
MoNuSeg	SwinUNet	0.7850	0.8750	0.8787	0.8715
	Mask-RCNN	0.6465	0.7648	**0.8912**	0.7206
	TransUNet	0.7699	0.8650	0.8601	0.8702
	SwinUNet + AT	0.7856	0.8754	0.8796	0.8714
	UNet	0.7946	0.8814	0.8843	**0.8786**
	TransUNet + AT	0.7707	0.8655	0.8611	0.8702
	YOLOv7	0.7162	0.8240	0.8874	0.7882
	UNet + AT	**0.7948**	**0.8816**	0.8848	0.8785
	YOLO-U	0.7932	0.8805	0.8863	0.8750
CPM17	SwinUNet	0.7936	0.8795	0.9298	0.8454
	TransUNet	0.7813	0.8713	0.8982	**0.8502**
	YOLOv7	0.6494	0.7650	0.9094	0.7159
	Mask-RCNN	0.7035	0.8122	0.9005	0.7683
	SwinUNet + AT	0.7927	0.8788	0.9304	0.8442
	UNet	**0.7948**	**0.8802**	0.9335	0.8448
	TransUNet + AT	0.7814	0.8714	0.8993	0.8496
	UNet + AT	0.7942	0.8798	0.9335	0.8442
	YOLO-U	0.7881	0.8756	**0.9338**	0.8381
CoNSeP	SwinUNet	0.7296	0.8347	0.8417	**0.8283**
	TransUNet	0.6695	0.7861	0.8184	0.7637
	YOLOv7	0.5069	0.6099	0.8291	0.5920
	SwinUNet + AT	0.7290	0.8342	0.8425	0.8266
	TransUNet + AT	0.6685	0.7852	0.8191	0.7620
	UNet	**0.7391**	**0.8412**	0.8656	0.8220
	Mask-RCNN	0.5536	0.6664	**0.8724**	0.6325
	UNet + AT	0.7387	0.8408	0.8659	0.8212
	YOLO-U	0.7345	0.8375	0.8687	0.8143

5.4 Limitations

In the YOLO-U approach, it is presupposed that the UNet model executes semantic segmentation effectively. As a result, the efficacy of the YOLO-U model is limited by the capabilities of the UNet model, which means that an object not segmented by UNet cannot be predicted by YOLO-U. Moreover, since the outputs of both models are utilized, the training and testing durations exceed the sum of the individual times required for each model.

6
MEDICAL DOMAIN-SPECIFIC MODELS FOR NUCLEI INSTANCE SEGMENTATION

In the literature, nuclei instance segmentation tasks have been addressed using UNet and its derivatives, general-purpose instance segmentation models, and models designed specifically for nuclei instance segmentation. One of the most successful models specifically designed for nuclei instance segmentation is HoVer-Net [20]. In the HoVer-Net study, a PQ value of 0.597 is reported for the MoNuSeg dataset, which is 1.2$\%$ lower than that of the YOLO-U model. However, various studies indicate that different experimental setups can lead to either more successful or less successful outcomes [50]. Consequently, we have evaluated the HoVer-Net model within our specifically designed experimental framework.

In the domain of medical imaging, StarDist [118, 119] is another model known for its state-of-the-art results. There are several implementations of StarDist for different nuclei instance segmentation datasets in the literature [21, 120]. However, naturally, no study mirrors our experimental setup exactly. Due to its reported success in the literature, the StarDist model has also been incorporated into our study.

This chapter examines HoVer-Net and StarDist, two prominent models extensively used for instance segmentation in medical imaging. Both models were trained on the MoNuSeg dataset specifically for nuclei instance segmentation. The strengths and weaknesses of these models have been analyzed using the MoNuSeg, CoNSeP, and CPM17 datasets.

6.1 HoVer-Net

HoVer-Net [20] is a model specifically developed for the nuclei instance segmentation task. It approaches the problem by dividing it into three distinct tasks: segmenting nuclei pixels, identifying and separating nuclei centers, and classifying

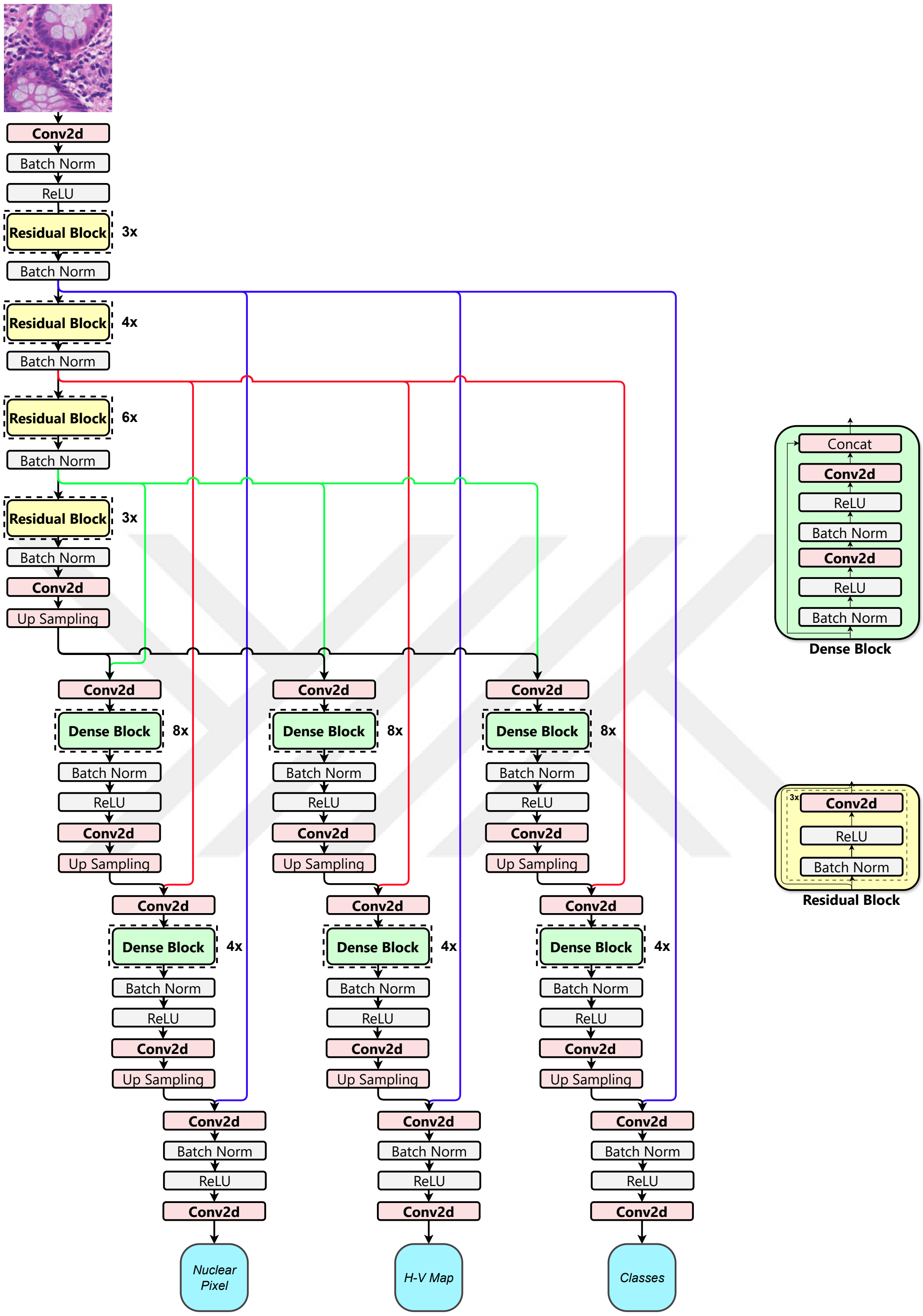


Figure 6.1 HoVEr-Net model architecture

the types of nuclei. To tackle each of these problems, it employs specialized head architectures that share common structural features.

The HoVer-Net model for nuclei instance segmentation fundamentally consists of four main components: a backbone and three heads. Figure 6.1 details the architecture of HoVer-Net. The backbone architecture is composed of consecutive residual blocks and batch normalization layers and is designed to process input images of 270×270 pixels. Unlike the classic ResNet architecture, each residual block in HoVer-Net is uniquely structured, starting with batch normalization, followed by a ReLU activation function, and a convolution filter. This sequence is repeated three times.

The head architecture in HoVer-Net incorporates skip connections from the backbone, similar to other models we have analyzed. The principal idea, as observed in previously discussed models, is to combine coarse and fine feature vectors and enhance the gradient flow. The head's main components are successive dense blocks, which are similar to residual blocks but organized differently. Each dense block sequentially applies batch normalization, ReLU activation, and convolution filters. Unlike residual blocks that add input to output, dense blocks concatenate these vectors [115].

Each of the three heads in HoVer-Net is trained for a distinct task. The first head performs semantic segmentation to differentiate between nuclei and background, producing an 80×80 pixel binary output classifying it as either nucleus or background. The primary objective of this head is to identify the pixels belonging to nuclei as precisely as possible. The overall performance of the model is dependent on the accuracy of this head, so its effectiveness is critical to the model's success.

Another head within the HoVer-Net architecture focuses on determining the distance of each pixel to the nearest nucleus center, outputting a $2 \times 80 \times 80$ matrix that represents vertical and horizontal distances. This feature is particularly crucial in post-processing, where it aids in resolving which nucleus each pixel belongs to, especially in cases where nuclei are in close proximity.

The last head in the architecture is responsible for classifying the types of nuclei. It produces a traditional semantic segmentation output. However, the performance of this head does not impact the overall model performance as significantly as the other heads. This is because, during post-processing, the classification of nuclei is refined by averaging the class pixels within the instances identified by the two other heads, which focus on separating nuclei pixels and calculating their proximity.

6.1.1 HoVer-Net Experimental Setup

The model training for this chapter utilized the MoNuSeg dataset, consistent with other chapters. For testing, cross-domain tests were conducted using the MoNuSeg, CPM17, and CoNSeP datasets. The hyperparameters strictly adhere to those

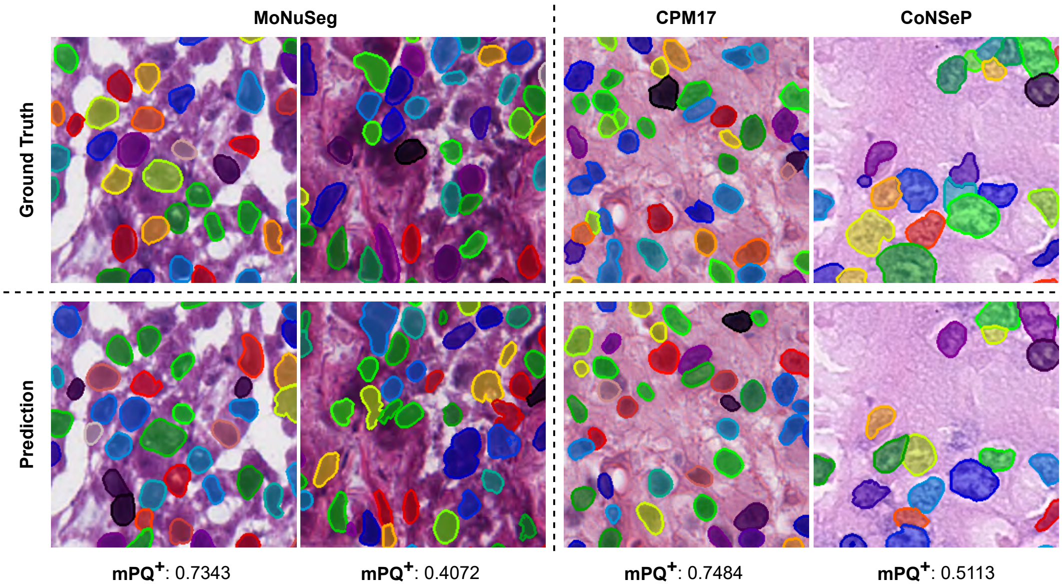


Figure 6.2 HoVer-Net model prediction

Table 6.1 Instance segmentation results of HoVer-Net

Test Dataset	PQ^+	PQ	SQ	DQ	AJI^+	AJI
MoNuSeg	0.6406	0.6370	0.7648	0.8316	0.6677	0.6611
CPM17	0.6357	0.6258	0.7802	0.7984	0.6566	0.6516
CoNSeP	0.4019	0.3859	0.7480	0.5129	0.3859	0.3778

specified in the original HoVer-Net study. The training process utilized overlapping patches of 280x280. To ensure a fair comparison with other models, the testing protocol was carried out on RoI. The patch merging process strictly followed the protocols established in the HoVer-Net study, without introducing any new methodologies.

6.1.2 HoVer-Net Results

Table 6.1 presents the instance segmentation results obtained with the HoVer-Net model. With a PQ^+ score of 0.6406, HoVer-Net demonstrates robust performance in instance segmentation. However, as observed in other experiments, it exhibits lower performance on the CoNSeP dataset compared to the MoNuSeg and CPM17 datasets. On the CPM17 dataset, it achieves a higher SQ than on MoNuSeg. Comparing the PQ^+ scores of 0.6406 on the MoNuSeg dataset and 0.6357 on the CPM17 dataset, it can be inferred that the HoVer-Net model, specifically developed for nuclei instance segmentation tasks, possesses strong generalization capabilities.

Given that one of the heads of the HoVer-Net model is dedicated to semantic segmentation, it is expected to produce better results compared to detection-based

Table 6.2 Semantic segmentation results of HoVer-Net

	Test Dataset	μ	Background	Nuclei
Dice	**MoNuSeg**	0.8879	0.9500	0.8258
	CPM17	0.8985	0.9638	0.8332
	CoNSeP	0.8180	0.9390	0.6970
IoU	**MoNuSeg**	0.8040	0.9047	0.7032
	CPM17	0.8221	0.9300	0.7141
	CoNSeP	0.7100	0.8851	0.5349
Recall	**MoNuSeg**	0.8960	0.9424	0.8497
	CPM17	0.8681	0.9882	0.7481
	CoNSeP	0.7840	0.9723	0.5956
Precision	**MoNuSeg**	0.8804	0.9577	0.8031
	CPM17	0.9404	0.9405	0.9402
	CoNSeP	0.8739	0.9080	0.8399

models. The semantic segmentation results for the HoVer-Net model are shown in Table 6.2, where it achieves remarkably high results across all evaluated datasets. Notably, on the CPM17 dataset, a precision of 0.9402 is particularly impressive. As evidenced by the instance segmentation predictions displayed in Figure 6.2, the HoVer-Net model effectively delineates the boundaries of nuclei.

6.2 StarDist

The classical approach to nuclei instance segmentation typically involves first classifying pixels as either nuclei or not, followed by grouping to form nuclei instances. However, StarDist [118, 119] introduces a novel perspective by representing nuclei as star-convex polygons. This method solves the problem in a single step, making it both efficient and robust to noise such as misclassifications in semantic segmentation.

Figure 6.3 illustrates the architecture of the StarDist model. As depicted, the model possesses a straightforward architecture, closely resembling the UNet model described in Chapter 3.1, except for a modification in the decoder block. Like the classical UNet, the encoder in the StarDist model progresses through five levels. However, in StarDist's adapted UNet architecture, there is no skip connection between the decoder block and the level 0 Conv Block, meaning the decoder has one fewer level. Consequently, for an input size of $w \times h$, the output size is reduced to $w/2 \times h/2$.

A convolution filter is applied to the output of the decoder block to generate input matrices for two distinct heads, each consisting of a convolution block. The first

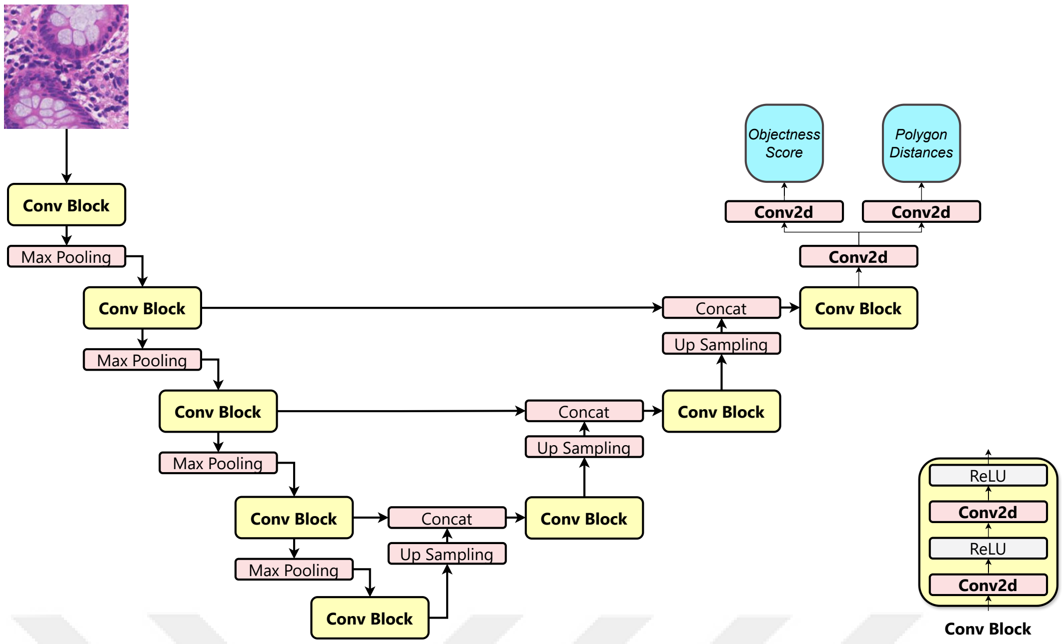


Figure 6.3 StarDist model architecture

head determines whether a pixel is in the center of an object, similar to the function used in object detection models. The second head predicts the distances to the boundaries of the nuclei, using the specified pixel as the center point.

The key feature of the model lies in its representation of distances. It conceptualizes nuclei boundaries as star-convex polygons and computes distances from the center to these boundaries. Instead of using polygon points derived directly from the ground truth instance segmentation map, this model generates points by projecting radii from the center at fixed angular intervals. This method ensures that all instances are represented with the same number of polygon points, independently from the segmentation map in the ground truth.

During the prediction process, objects with an objectness score below a certain threshold are excluded. Non-maximum suppression is then applied to the selected objects to remove any overlaps. The remaining objects constitute the final predictions of the model. To maximize performance, hyperparameters applied in post-processing are optimized for a given dataset.

StarDist includes task-specific designs for tackling the nuclei segmentation problem, most notably representing object borders as convex polygons. This approach is not suitable for segmenting concave objects and does not address occlusions between overlapping objects. While these design choices enhance its effectiveness for nuclei instance segmentation, they limit the model's applicability to other segmentation tasks.

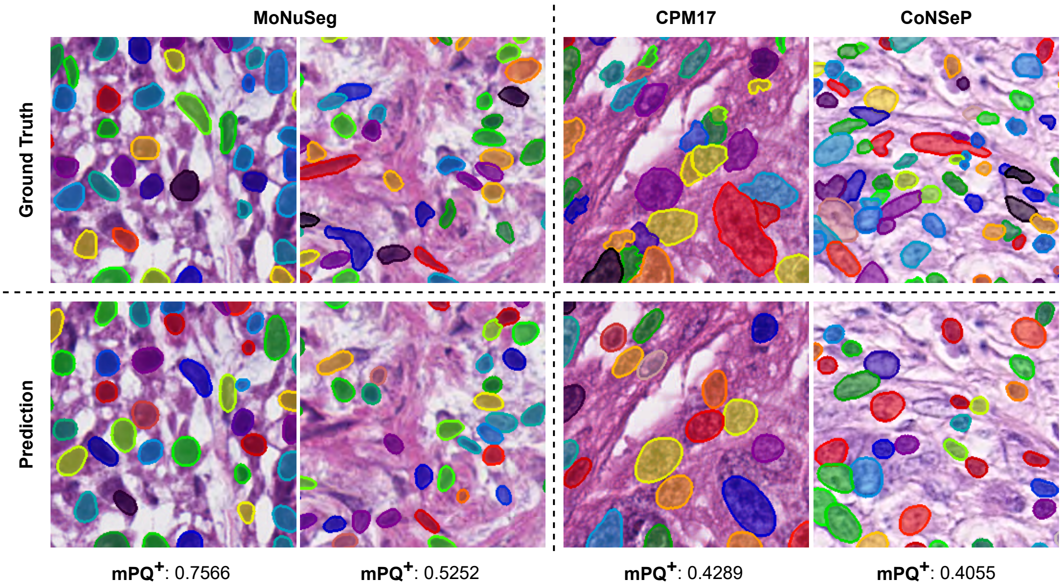


Figure 6.4 StarDist model prediction

Table 6.3 Instance segmentation results of StarDist

Test Dataset	PQ^+	PQ	SQ	DQ	AJI^+	AJI
MoNuSeg	0.6878	0.6839	0.7746	0.8822	0.6835	0.6727
CPM17	0.6632	0.6547	0.7862	0.8302	0.6604	0.6499
CoNSeP	0.4023	0.3963	0.7415	0.5334	0.4272	0.3806

6.2.1 StarDist Experimental Setup

StarDist adjusts multiple hyperparameters according to the specific dataset used, which is why this study strictly follows the methodology recommended for StarDist. In this study, consistent with previous implementations, the model was trained using patches of size 256×256 and tested on RoI. For testing, the MoNuSeg, CPM17, and CoNSeP datasets were used, as with other models.

6.2.2 StarDist Results

The StarDist model, trained using the MoNuSeg dataset, was evaluated on the MoNuSeg, CPM17, and CoNSeP datasets. Its instance segmentation performance is detailed in Table 6.3. Like its performance on other datasets, the StarDist demonstrates high performance on the MoNuSeg and CPM17 datasets while showing relatively lower performance on the CoNSeP dataset. Notably, it achieved impressive PQ^+ scores of 0.6867 on MoNuSeg and 0.6632 on CPM17. In the MoNuSeg dataset, the SQ score is consistent with other models, ranging between 75-80, but the DQ score notably approaches the 90th percentile.

As evident from Figure 6.4, the StarDist model draws very precise oval masks for

Table 6.4 Semantic segmentation results of StarDist

	Test Dataset	μ	Background	Nuclei
Dice	**MoNuSeg**	0.8893	0.9521	0.8265
	CPM17	0.8864	0.9591	0.8137
	CoNSeP	0.8039	0.9334	0.6744
IoU	**MoNuSeg**	0.8065	0.9086	0.7043
	CPM17	0.8037	0.9215	0.6859
	CoNSeP	0.6919	0.8751	0.5087
Recall	**MoNuSeg**	0.8886	0.9528	0.8497
	CPM17	0.8590	0.9818	0.7362
	CoNSeP	0.7740	0.9638	0.5841
Precision	**MoNuSeg**	0.8900	0.9515	0.8285
	CPM17	0.9235	0.9375	0.9094
	CoNSeP	0.8512	0.9048	0.7975

convex objects in the MoNuSeg dataset. However, it is noteworthy that it fails to generate successful masks for some non-convex objects and also misses detecting many nuclei that are non-convex. Despite these limitations, the model is generally effective in performing instance segmentation.

Table 6.4 presents the semantic segmentation performance of the StarDist model on the MoNuSeg, CPM17, and CoNSeP datasets. The recall values obtained for these datasets are 0.8497, 0.7362, and 0.5841, respectively. Assuming that the StarDist model is closer in operation to detection-based models like YOLOv7 rather than semantic segmentation models like HoVer-Net, it demonstrates an unexpectedly high performance in recall. The effectiveness of the star-convex polygon approach in simplifying the problem and enhancing model performance is evident from these results.

7 HR-YOLO: IMPROVED YOLO FOR NUCLEI INSTANCE SEGMENTATION

In all evaluated models, the architectures involve reducing the dimensions of the input images through a series of downsampling blocks to decrease the size of feature vectors. Subsequently, these dimensions are restored to those of the input images using a series of upsampling blocks. For instance, in the YOLOv7 model, the output feature vector produced by the initial convolution block is half the size of the input dimension. Although this feature may yield excellent results on datasets like COCO [121], we believe it has a negative impact on tasks such as nuclei instance segmentation, where accurate detection and segmentation of small objects are crucial.

In this chapter, we introduce the HR-YOLO, specifically designed for the task of nuclei instance segmentation. This model is developed by integrating a backbone derived from HRNet [122] with the neck and head components of YOLOv7.

7.1 Methodology

Traditional networks typically downsample the input image to a lower resolution early in the process, then attempt to reconstruct high-resolution details in subsequent layers. However, HRNet [122] introduces an alternative approach by maintaining parallel branches at different resolutions. This method allows the network to preserve high-resolution details while simultaneously capturing low-resolution contextual information. Interconnections between these branches facilitate the integration of contextual data across different resolutions.

Figure 7.1 illustrates the backbone used in the HR-YOLO model. This backbone is a 3-branch HRNet configuration that produces feature vectors at resolutions of 1/4, 1/8, and 1/16 of the input size. The distinctiveness of the model lies in the transitions between branches, resulting in a complex structure. Initially, the

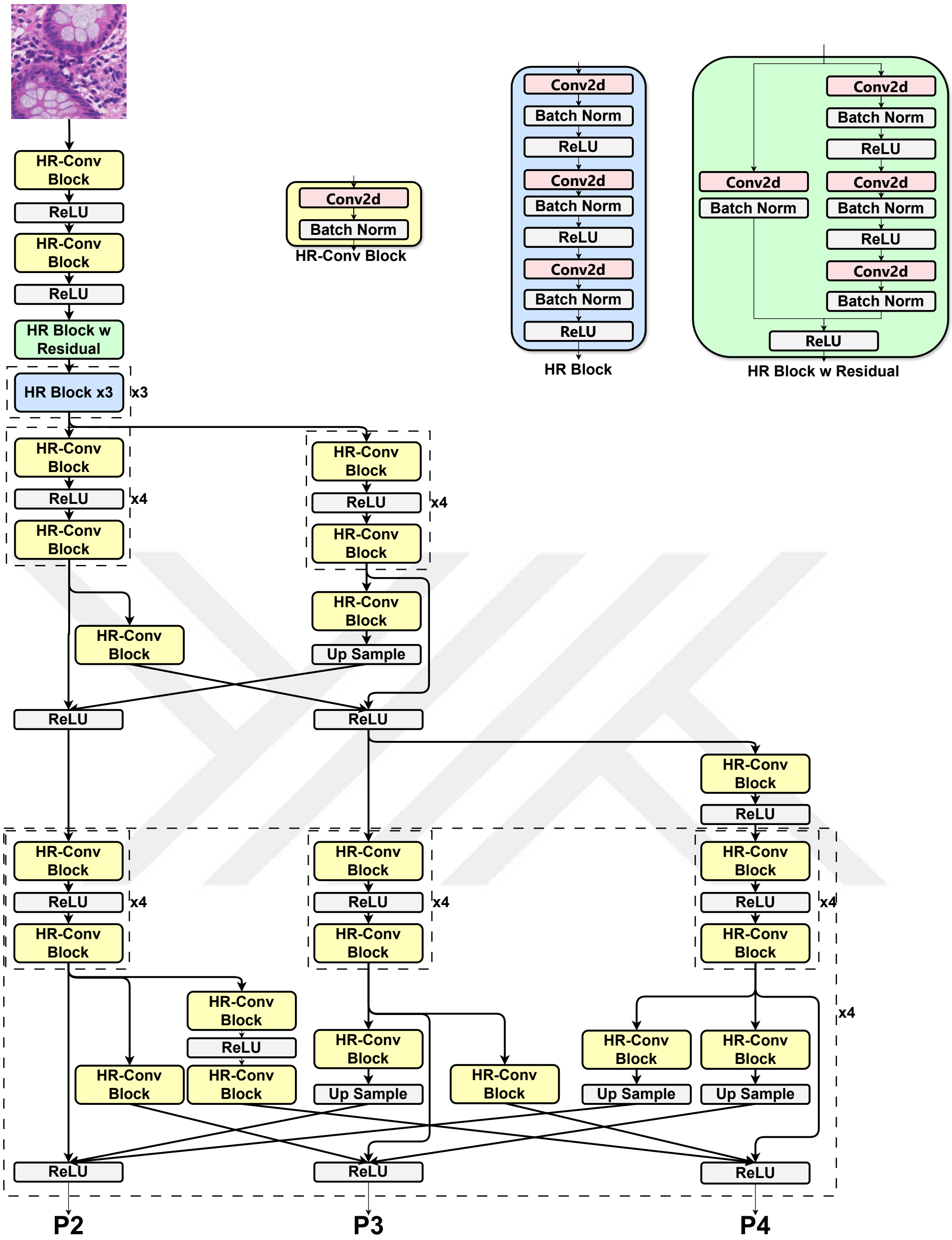


Figure 7.1 HR-YOLO model backbone

input image is processed through an HR-Conv Block, followed by HR Blocks with residual connections.

The HR-Conv Block is a foundational element within the HR-YOLO model's architecture, consisting of convolution filters and batch normalization layers. The HR Block is formed by repeating a sequence of convolution filters, batch normalization, and ReLU activation layers three times. Additionally, the HR Block with Residual Connection mirrors the HR Block's structure but

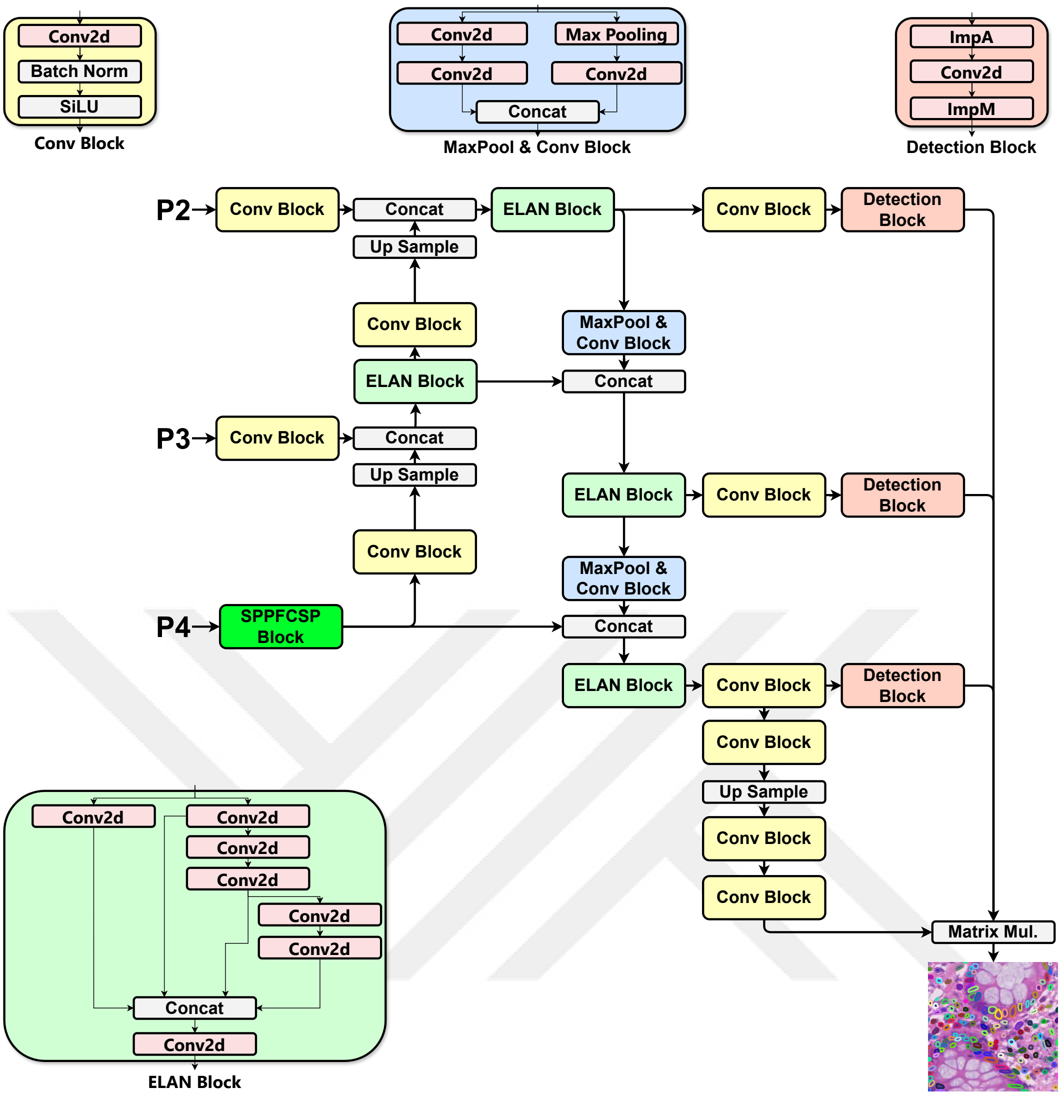


Figure 7.2 HR-YOLO neck and head

integrates a secondary connection that includes extra convolutional filters and batch normalization, improving gradient flow across the network.

In the HR-YOLO model architecture, a stride of 2 convolution filters is applied to the main branch, generating new branches that produce even lower-resolution feature vectors at 1/8 scale. Sequential applications of HR-Conv blocks extract feature vectors from both branches.Then, the lower-resolution branch is upsampled to match the output of the high-resolution branch, while a 2 stride convolution filter is applied to the high-resolution branch to merge it with the output of the lower-resolution branch using ReLU activation. This process continues across three different levels (P2, P3, and P4) until feature vector outputs are generated.

The neck and head of the HR-YOLO model retain the same architectural structure used in the YOLOv7 model. The neck combines coarse and fine feature vectors, processing inputs at 1/4, 1/8, and 1/16 scales of the original image size and producing feature vectors at corresponding resolutions. For feature extraction, the model utilizes Conv Blocks and ELAN Blocks. Similar to YOLOv7, it employs MaxPool & Conv Blocks to reduce the dimensions of the feature vectors by half. The head utilizes a simple YOLOv7 detection block that includes an additive implicit layer, convolution filters, and a multiplicative implicit layer. Crucially, the detection block inputs high-resolution feature vectors, resulting in a high number of proposed bounding boxes. For instance, for a feature map that is 1/4 the size of the input image, a bounding box is proposed for every 4-pixel grid. Non-maximum suppression is then applied to remove overlapping bounding boxes.

For the mask output, the HR-YOLO model adopts the same structure as used in YOLOv7. However, to enhance the segmentation accuracy, the model is fed with feature vectors scaled to 1/16 of the input image size instead of 1/32, aiming to achieve more refined segmentation results.

7.2 HR-YOLO Experimental Setup

The SGD optimizer was used to train 300 epochs with patches sized at 256×256. The initial learning rate was set to 0.01, without employing a learning rate scheduler. Scale augmentation was not applied. However, flip augmentations for both up-down and left-right were implemented with a 50% probability to enhance model robustness against orientation variations. The batch size was set to 16, with a gradient accumulation parameter of 4. The training was conducted without the aid of pretrained models.

While the initial training was conducted on 256x256 image patches, the testing phase utilized RoI. Patches of the same size were extracted from these RoI and fed into the model. The results from these patches were then combined to produce complete predictions for the entire RoI.

The model allows a pixel to be assigned to multiple instance masks. However, this multiplicity is not suitable for nuclei instance segmentation tasks. To improve performance and accuracy, a confidence score-based method has been used, like our proposed methodology in the YOLOv7 instance segmentation model. This method prioritizes masks with the highest confidence scores during placement on the RoI. Subsequent predictions are only added if a significant portion of the area remains unassigned. Additionally, predictions were generated on 256×256 patches with a step size of 64. Predictions located on the boundaries of the patches were excluded when compiling the RoI. This exclusion aims to avoid the negative boundary effects

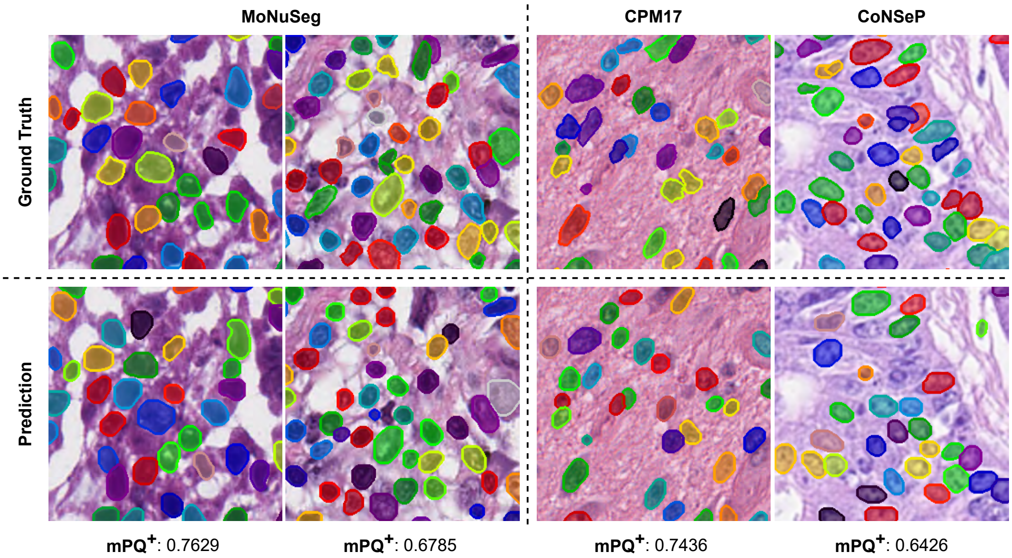


Figure 7.3 HR-YOLO model prediction

Table 7.1 Instance segmentation results of HR-YOLO

Test Dataset	PQ$^+$	PQ	SQ	DQ	AJI$^+$	AJI
MoNuSeg	0.6864	0.6800	0.7787	0.8726	0.6661	0.6561
CPM17	0.6661	0.6609	0.7975	0.8266	0.6476	0.6398
CoNSeP	0.3893	0.3626	0.7741	0.4667	0.3281	0.3024

of the convolution operation on nuclei located at the edges of patches.

7.3 Results

Table 7.1 presents the instance segmentation results of the HR-YOLO model trained on the MoNuSeg dataset. The model has demonstrated exceptional performance, with a PQ^+ of 0.6864 on the MoNuSeg dataset and 0.6661 on the CPM17 dataset. As shown in Figure 7.3, it also achieves significant segmentation performance with an SQ of 0.7975 on the CPM17 dataset and 0.7787 on the MoNuSeg dataset. However, as HR-YOLO is a detection-based approach, its performance in the AJI and AJI^+ metrics does not reach the same high levels.

The YOLO-U model demonstrates high performance on the MoNuSeg and CPM17 datasets; however, it does not perform as well on the CoNSeP dataset. Despite achieving a high SQ of 0.7741, the DQ remains considerably low at 0.4667. As previously discussed, detection-based models tend to exhibit reduced true positives in different domains due to the influence of the confidence threshold. Table 5.2 provides the semantic segmentation results for the HR-YOLO model, with a recall

Table 7.2 Semantic segmentation results of HR-YOLO

	Test Dataset	μ	Background	Nuclei
Dice	**MoNuSeg**	0.8830	0.9496	0.8163
	CPM17	0.8784	0.9577	0.7991
	CoNSeP	0.7406	0.9226	0.5587
IoU	**MoNuSeg**	0.7969	0.9041	0.6897
	CPM17	0.7921	0.9188	0.6654
	CoNSeP	0.6220	0.8563	0.3876
Recall	**MoNuSeg**	0.8811	0.9514	0.8107
	CPM17	0.8434	0.9873	0.6995
	CoNSeP	0.7008	0.9762	0.4255
Precision	**MoNuSeg**	0.8849	0.9478	0.8220
	CPM17	0.9307	0.9298	0.9317
	CoNSeP	0.8439	0.8745	0.8132

of 0.4255 on the CoNSeP dataset, supporting this observation.

8
DISCUSSIONS

In this study, we have endeavored to tackle the task of nuclei instance segmentation, which is an important undertaking in real-world histopathology image analysis. Quantifying nuclear density in specific tissue areas and analyzing nuclear morphology have significant diagnostic importance. Therefore, our research focuses on the PQ^+ metric, which best measures these aspects. This chapter provided a comprehensive comparison and analysis of the performance of the various models discussed and proposed throughout this study, examining the reasons for their effectiveness or shortcomings.

As previously discussed, there is a notable absence of a universally accepted training and testing protocol in the literature for nuclei instance segmentation. To address this gap, this study established a standardized protocol using the widely adopted MoNuSeg dataset for model training, with its test subset utilized for validation. The primary test dataset employed was CPM17, with the CoNSeP dataset used for cross-domain testing. Model training was executed on patches derived from the MoNuSeg dataset, whereas the tests were carried out on RoI. This method facilitated the development of 12 distinct models. By maintaining consistent training and testing conditions, this study not only enables a fair comparison of model performances across domains but also establishes an experimental setup that could potentially set a benchmark for future research in this area.

This study conducted a comprehensive comparative analysis for all models analyzed, focusing on their performance across the MoNuSeg, CPM17, and CoNSeP datasets, as outlined in Table 8.1, with results ranked according to the PQ^+ metric. In the MoNuSeg dataset, the HR-YOLO and StarDist models showcased nearly identical performances. StarDist achieved a PQ^+ score of 0.6878, while HR-YOLO closely followed with a score of 0.6864, differing only by 0.0014. HR-YOLO outperformed in SQ, indicating better segmentation quality, while StarDist performed in DQ, implying more accurate detection capabilities. Remarkably, in the AJI metric, HoVer-Net, HR-YOLO, and StarDist outperformed

Table 8.1 Comparison of instance segmentation results

	Model	PQ^+	PQ	SQ	DQ	AJI^+	AJI
MoNuSeg	SwinUNet	0.4623	0.4602	0.7487	0.6134	0.5462	0.5136
	Mask-RCNN	0.4882	0.4936	0.7333	0.6727	0.4344	0.4304
	TransUNet	0.4964	0.4916	0.7599	0.6455	0.5496	0.5137
	SwinUNet + AT	0.5259	0.5262	0.7487	0.7020	0.5473	0.5139
	UNet	0.5414	0.5392	0.7631	0.7056	0.5819	0.5530
	TransUNet + AT	0.5437	0.5381	0.7599	0.7071	0.5507	0.5143
	YOLOv7	0.5698	0.5548	0.7522	0.7363	0.5284	0.5190
	UNet + AT	0.5700	0.5701	0.7631	0.7464	0.5824	0.5532
	YOLO-U	0.6199	0.6121	0.7608	0.8037	0.6291	0.6215
	HoVer-Net	0.6406	0.6370	0.7648	0.8316	0.6677	0.6611
	HR-YOLO	0.6864	0.6800	**0.7787**	0.8726	0.6661	0.6561
	StarDist	**0.6878**	**0.6839**	0.7746	**0.8822**	**0.6835**	**0.6727**
CPM17	SwinUNet	0.4495	0.4544	0.7563	0.5960	0.5747	0.5604
	TransUNet	0.4759	0.4673	0.7702	0.6013	0.5596	0.5227
	YOLOv7	0.4762	0.4580	0.7389	0.6178	0.4167	0.4110
	Mask-RCNN	0.5235	0.5104	0.7145	0.7080	0.5015	0.4919
	SwinUNet + AT	0.5335	0.5340	0.7568	0.7009	0.5750	0.5603
	UNet	0.5301	0.5282	0.7765	0.6758	0.5899	0.5782
	TransUNet + AT	0.5358	0.5244	0.7704	0.6756	0.5605	0.5232
	UNet + AT	0.5602	0.5567	0.7765	0.7126	0.5896	0.5777
	YOLO-U	0.5852	0.5787	0.7722	0.7464	0.6078	0.6018
	HoVer-Net	0.6357	0.6258	0.7802	0.7984	0.6566	**0.6516**
	StarDist	0.6632	0.6547	0.7862	**0.8302**	**0.6604**	0.6499
	HR-YOLO	**0.6661**	**0.6609**	**0.7975**	0.8266	0.6476	0.6398
CoNSeP	SwinUNet	0.1677	0.1626	0.7130	0.2270	0.2347	0.1445
	TransUNet	0.1972	0.1926	0.7327	0.2612	0.2618	0.1916
	YOLOv7	0.1992	0.1860	0.7270	0.2561	0.1655	0.1639
	SwinUNet + AT	0.2214	0.2206	0.7135	0.3091	0.2342	0.1435
	TransUNet + AT	0.2327	0.2282	0.7332	0.3098	0.2609	0.1900
	UNet	0.2608	0.2497	0.7448	0.3334	0.2978	0.2120
	Mask-RCNN	0.2635	0.2764	0.7498	0.3682	0.2478	0.2431
	UNet + AT	0.2820	0.2721	0.7449	0.3637	0.2972	0.2111
	YOLO-U	0.3674	0.3526	0.7341	0.4773	0.4024	0.3623
	HR-YOLO	0.3893	0.3626	**0.7741**	0.4667	0.3281	0.3024
	HoVer-Net	0.4019	0.3859	0.7480	0.5129	0.3859	0.3778
	StarDist	**0.4023**	**0.3963**	0.7415	**0.5334**	**0.4272**	**0.3806**

other models, demonstrating their robustness in nuclei instance segmentation tasks.

In the CPM17 dataset, the HR-YOLO model emerged as the top performer, achieving a PQ^+ of 0.6661, distinguishing itself as the most effective at instance segmentation among the evaluated models. Moreover, in terms of the AJI metric, HoVer-Net demonstrated the highest scores, while StarDist excelled in the AJI^+

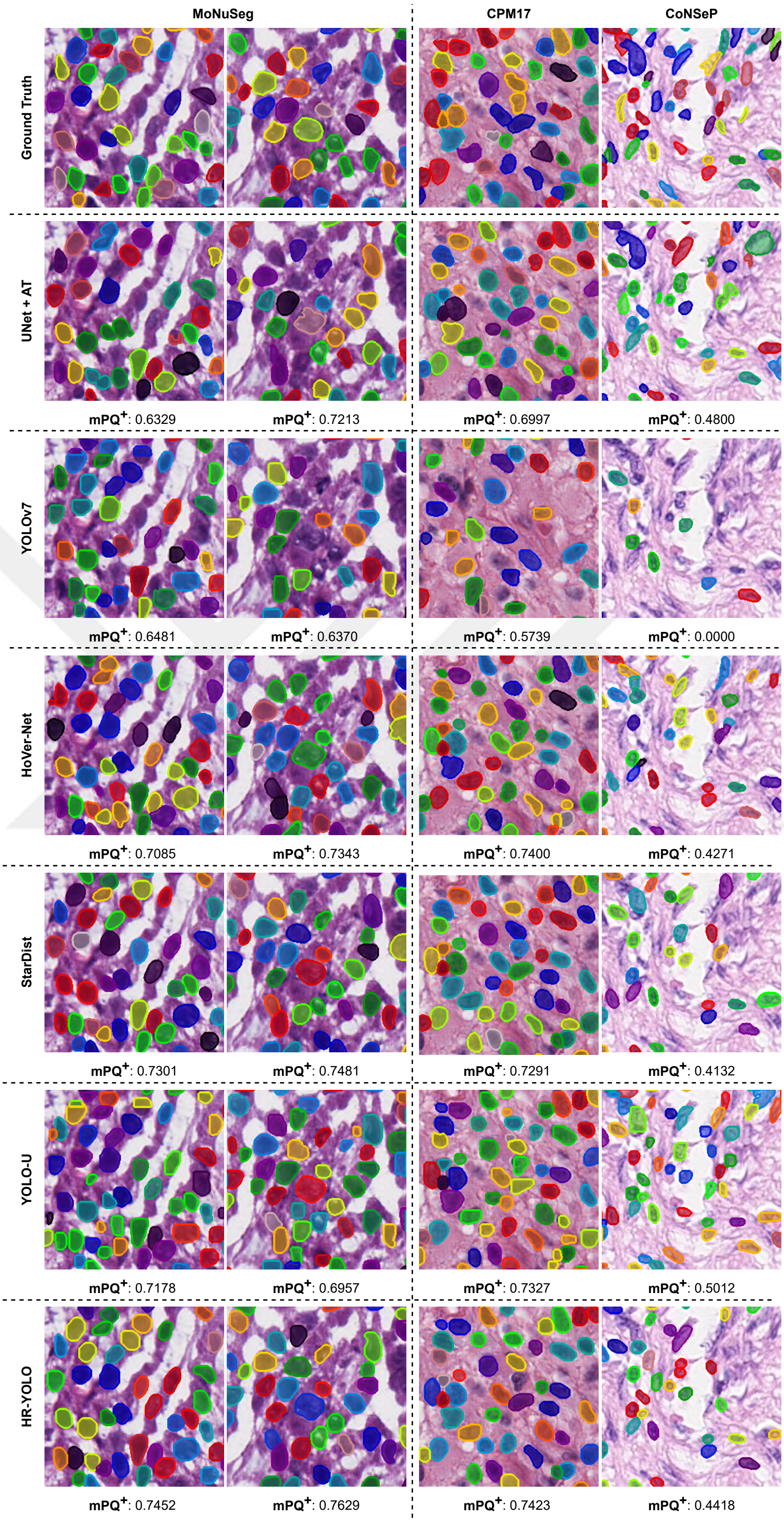


Figure 8.1 Comparison of models predictions

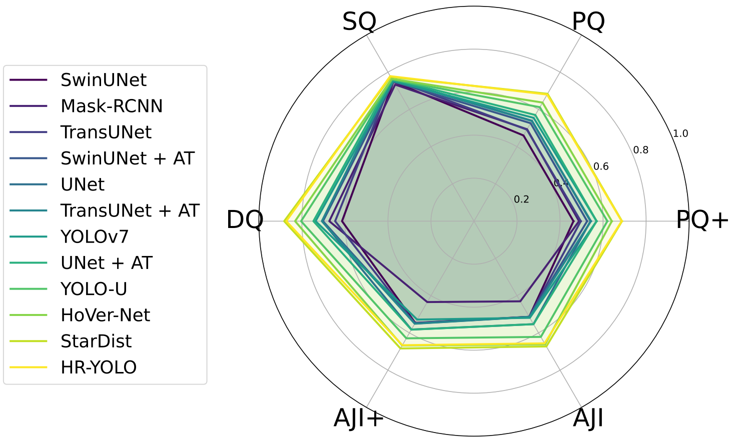
metric. These outcomes underscore the crucial importance of choosing appropriate metrics for model comparison, as they can significantly influence the interpretation of a model's performance in nuclei instance segmentation.

The HR-YOLO model's impressive performance in the PQ metric is a direct result of its effective segmentation capability. This outcome highlights the model's architectural advantage in preserving high-resolution details alongside capturing essential low-resolution contextual information, which enhances its segmentation effectiveness. However, its lesser performance in the AJI and AJI$^+$ metrics is primarily due to its failure to detect certain objects, pointing to limitations in its object detection capabilities.

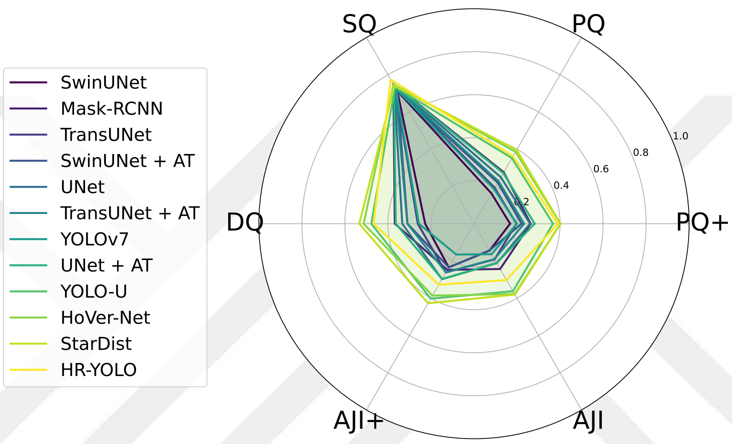
In the CoNSeP dataset, the three most successful models are HR-YOLO, HoVer-Net, and StarDist, with StarDist leading in performance with a PQ$^+$ score of 0.4023. The success of these models, specifically designed for medical applications, in a domain-specific dataset like CoNSeP is not unexpected. Despite this, HR-YOLO continues to exhibit the highest SQ across different datasets, underscoring its superior segmentation quality. However, its relatively low DQ score of 0.4667 in the CoNSeP dataset indicates a decrease in detection performance across different domains, highlighting challenges in generalizing the detection capabilities of the model.

In summary, the results across all three datasets clearly indicate that HR-YOLO, HoVer-Net, and StarDist are the most successful models. HR-YOLO and StarDist demonstrate closely matched performance on the MoNuSeg dataset used for validation. In the CPM17 dataset, designated as the primary test set, HR-YOLO achieves the best results. On the CoNSeP dataset, used for cross-domain testing, both HoVer-Net and StarDist produce noteworthy outcomes. However, HR-YOLO shows relatively weaker performance in this cross-domain test. Visual representations of model predictions are provided in Figure 8.1.

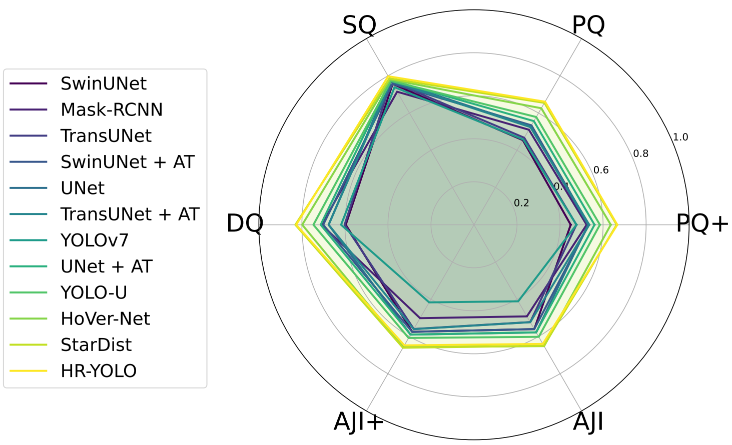
Figure 8.2 presents comparative analyses of models across six different metrics. Similar distributions are observed in the MoNuSeg and CPM17 datasets, while the CoNSeP dataset consistently demonstrates lower results. As detailed in Chapter 1, nuclei instance segmentation models in the literature can be classified into three main categories. It is evident that models based on semantic segmentation yield the least successful outcomes, as indicated by their proximity to the center in the figures. In contrast, models based on object detection approaches tend to yield more favorable results. While semantic quality (SQ) scores are similar across different datasets, detection quality (DQ) of models is significantly influenced by variations



(a) Comparison of Instance Segmentation Results on MoNuSeg



(b) Comparison of Instance Segmentation Results on CoNSeP



(c) Comparison of Instance Segmentation Results on CPM17

Figure 8.2 Comparison of instance segmentation results

in the domain, underscoring the challenges in achieving robust cross-domain performance.

In Table 8.2, the semantic segmentation results of various models are presented in the same order as their instance segmentation performances. Contrary to

Table 8.2 Comparison of mean semantic segmentation results

	Model	μ IoU	μ Dice	μ Precision	μ Recall
MoNuSeg	SwinUNet	0.7850	0.8750	0.8787	0.8715
	Mask-RCNN	0.6465	0.7648	**0.8912**	0.7206
	TransUNet	0.7699	0.8650	0.8601	0.8702
	SwinUNet + AT	0.7856	0.8754	0.8796	0.8714
	UNet	0.7946	0.8814	0.8843	0.8786
	TransUNet + AT	0.7707	0.8655	0.8611	0.8702
	YOLOv7	0.7162	0.8240	0.8874	0.7882
	UNet + AT	0.7948	0.8816	0.8848	0.8785
	YOLO-U	0.7932	0.8805	0.8863	0.8750
	HoVer-Net	0.8040	0.8879	0.8804	**0.8960**
	HR-YOLO	0.7969	0.8830	0.8849	0.8811
	StarDist	**0.8065**	**0.8893**	0.8900	0.8886
CPM17	SwinUNet	0.7936	0.8795	0.9298	0.8454
	TransUNet	0.7813	0.8713	0.8982	0.8502
	YOLOv7	0.6494	0.7650	0.9094	0.7159
	Mask-RCNN	0.7035	0.8122	0.9005	0.7683
	SwinUNet + AT	0.7927	0.8788	0.9304	0.8442
	UNet	0.7948	0.8802	0.9335	0.8448
	TransUNet + AT	0.7814	0.8714	0.8993	0.8496
	UNet + AT	0.7942	0.8798	0.9335	0.8442
	YOLO-U	0.7881	0.8756	0.9338	0.8381
	HoVer-Net	**0.8221**	**0.8985**	**0.9404**	**0.8681**
	StarDist	0.8037	0.8864	0.9235	0.8590
	HR-YOLO	0.7921	0.8784	0.9307	0.8434
CoNSeP	SwinUNet	0.7296	0.8347	0.8417	**0.8283**
	TransUNet	0.6695	0.7861	0.8184	0.7637
	YOLOv7	0.5069	0.6099	0.8291	0.5920
	SwinUNet + AT	0.7290	0.8342	0.8425	0.8266
	TransUNet + AT	0.6685	0.7852	0.8191	0.7620
	UNet	**0.7391**	**0.8412**	0.8656	0.8220
	Mask-RCNN	0.5536	0.6664	0.8724	0.6325
	UNet + AT	0.7387	0.8408	0.8659	0.8212
	YOLO-U	0.7345	0.8375	0.8687	0.8143
	HR-YOLO	0.6220	0.7406	0.8439	0.7008
	HoVer-Net	0.7100	0.8180	**0.8739**	0.7840
	StarDist	0.6919	0.8039	0.8512	0.7740

expectations that models based on semantic segmentation would outperform others, StarDist and HoVer-Net emerge as the top performers in the MoNuSeg and CPM17 datasets. Specifically, StarDist achieves a μIoU of 0.8065 in the MoNuSeg dataset, with HoVer-Net close behind at 0.8040. In the CPM17 dataset, HoVer-Net leads with a μIoU of 0.8221, followed by StarDist at 0.8037. Interestingly, in the CoNSeP dataset, the UNet model breaks from this trend and achieves the highest performance with a μIoU of 0.7391. Subsequently, the UNet with adaptive thresholding model and YOLO-U both perform well in this dataset.

Considering the results observed, it can be argued that models based on object detection produce more successful outcomes. Additionally, as seen with the StarDist model, enhancements in object representation, even when using a UNet backbone, can yield state-of-the-art results. This suggests that enhancing models like HR-YOLO, which currently lack specific representation capabilities, with advanced geometric representations such as convex polygons, could significantly improve their performance. We consider this a promising direction for future research.

9
CONCLUSION

This thesis analyzed various models and methodologies for nuclei instance segmentation and provided a comprehensive analysis of their performance on cross-domain datasets. This study has implemented and evaluated a variety of models, including UNet-based semantic segmentation approaches and modern detection-based models such as the YOLOv7 instance segmentation model. The study identified the strengths and limitations of existing techniques while also introducing novel methods for improving segmentation and detection accuracy. The proposed HR-YOLO model is one of the most important contributions. It improves the YOLO architecture for nuclei instance segmentation by making it better at finding small objects at high resolution. The model has exhibited outstanding performance, achieving a PQ^+ score of 0.6864 on the MoNuSeg dataset and 0.6661 on the CPM17 dataset. In addition, the YOLO-U model, an ensemble technique that combines the precision of UNet semantic segmentation capabilities with the detection capabilities of YOLOv7, reduces the dependency on confidence thresholds, aiding scenarios with limited expert review. The YOLO-U not only preserves the SQ performance of the UNet but also significantly boosts the DQ performance, thanks to its confidence threshold-free approach. For semantic segmentation based approaches, we also proposed the size-based adaptive thresholding technique, which dynamically adjusts the threshold, improving accuracy by effectively distinguishing between true objects and noise. For the UNet model, which is trained using the CoNIC dataset, a performance of 0.5090 mPQ^+ was achieved across 5 folds, which is 5.9% higher compared to results without any thresholding applied. Moreover, standardized experimental setup was established using popular datasets like MoNuSeg, CMP17, and CoNSeP, which allows for fair and reproducible comparisons of model performance across studies.

The analysis confirmed that while semantic segmentation-based models outperform in recall, they often struggle with separating overlapping nuclei. On the other hand, detection-based approaches, while less precise in boundary definition, provide better individual detection in dense cluster scenarios. Models that effectively balance segmentation and the separation of individual nuclei demonstrate superior performance in the task of nuclei instance segmentation.

In conclusion, this thesis has not only advanced the field of nuclei instance segmentation by proposing new models and methodologies, but it has also set a benchmark for future research by establishing a standardized testing protocol. Continued efforts in this direction will undoubtedly aid in early disease detection and treatment, potentially saving lives through improved diagnostic techniques.

REFERENCES

[1] A. L. Kierszenbaum, L. Tres, *Histology and Cell Biology: an introduction to pathology E-Book*. Elsevier Health Sciences, 2015.

[2] J. Lammerding, "Mechanics of the nucleus," *Comprehensive physiology*, vol. 1, no. 2, p. 783, 2011.

[3] C.-J. Kim *et al.*, "Nuclear morphology predicts cell survival to cisplatin chemotherapy," *Neoplasia*, vol. 42, p. 100 906, 2023.

[4] M. F. Santana, L. C. L. Ferreira, "Diagnostic errors in surgical pathology," *Jornal Brasileiro de Patologia e Medicina Laboratorial*, vol. 53, pp. 124–129, 2017.

[5] S. A. Boppart, R. Richards-Kortum, "Point-of-care and point-of-procedure optical imaging technologies for primary care and global health," *Science translational medicine*, vol. 6, no. 253, 253rv2–253rv2, 2014.

[6] R. Hollandi, N. Moshkov, L. Paavolainen, E. Tasnadi, F. Piccinini, P. Horvath, "Nucleus segmentation: Towards automated solutions," *Trends in Cell Biology*, vol. 32, no. 4, pp. 295–310, 2022.

[7] M. Aubreville *et al.*, "Mitosis domain generalization in histopathology images—the midog challenge," *Medical Image Analysis*, vol. 84, p. 102 699, 2023.

[8] E. S. Nasir, A. Parvaiz, M. M. Fraz, "Nuclei and glands instance segmentation in histology images: A narrative review," *Artificial Intelligence Review*, vol. 56, no. 8, pp. 7909–7964, 2023.

[9] O. Martos *et al.*, "Optimized detection and segmentation of nuclei in gastric cancer images using stain normalization and blurred artifact removal," *Pathology-Research and Practice*, vol. 248, p. 154 694, 2023.

[10] S. Nam, J. Jeong, M. Luna, P. Chikontwe, S. H. Park, "Pronet: Point refinement using shape-guided offset map for nuclei instance segmentation," in *International Conference on Medical Image Computing and Computer-Assisted Intervention*, Springer, 2023, pp. 528–538.

[11] S. Chen, C. Ding, D. Tao, H. Chen, "Darc: Distribution-aware re-coloring model for generalizable nucleus segmentation," in *International Conference on Medical Image Computing and Computer-Assisted Intervention*, Springer, 2023, pp. 591–601.

[12] A. Mahbod, G. Dorffner, I. Ellinger, R. Woitek, S. Hatamikia, “Improving generalization capability of deep learning-based nuclei instance segmentation by non-deterministic train time and deterministic test time stain normalization,” *Computational and Structural Biotechnology Journal*, vol. 23, pp. 669–678, 2024.

[13] A. Sharma, P. K. Mishra, “Dri-unet: Dense residual-inception unet for nuclei identification in microscopy cell images,” *Neural Computing and Applications*, vol. 35, no. 26, pp. 19 187–19 220, 2023.

[14] M. Kadaskar, N. Patil, “Anet: Nuclei instance segmentation and classification with attention-based network,” *SN Computer Science*, vol. 5, no. 4, p. 348, 2024.

[15] L. Pan, L. Wang, Z. Feng, Z. Xu, L. Xu, S. Peng, “Mgtunet: An new unet for colon nuclei instance segmentation and quantification,” in *2022 IEEE International Conference on Bioinformatics and Biomedicine (BIBM)*, IEEE, 2022, pp. 1121–1125.

[16] F. Meyer, “Topographic distance and watershed lines,” *Signal processing*, vol. 38, no. 1, pp. 113–125, 1994.

[17] V. T.-T. Vo, S.-H. Kim, T. Lee, “Mf-hovernet: An extension of hovernet for colon nuclei identification and counting (conic) challenge,” *arXiv preprint arXiv:2203.02161*, 2022.

[18] J. Wang *et al.*, “An improved hover-net for nuclear segmentation and classification in histopathology images,” *Neural Computing and Applications*, vol. 35, no. 19, pp. 14 403–14 417, 2023.

[19] J. Li, C. Wang, B. Huang, Z. Zhou, “Convnext-backbone hovernet for nuclei segmentation and classification,” *arXiv preprint arXiv:2202.13560*, 2022.

[20] S. Graham *et al.*, “Hover-net: Simultaneous segmentation and classification of nuclei in multi-tissue histology images,” *Medical image analysis*, vol. 58, p. 101 563, 2019.

[21] M. Weigert, U. Schmidt, “Nuclei instance segmentation and classification in histopathology images with stardist,” in *2022 IEEE International Symposium on Biomedical Imaging Challenges (ISBIC)*, IEEE, 2022, pp. 1–4.

[22] M. Kang, C.-M. Ting, F. F. Ting, R. C.-W. Phan, “Asf-yolo: A novel yolo model with attentional scale sequence fusion for cell instance segmentation,” *Image and Vision Computing*, p. 105 057, 2024.

[23] N. Kumar *et al.*, “A multi-organ nucleus segmentation challenge,” *IEEE transactions on medical imaging*, vol. 39, no. 5, pp. 1380–1391, 2019.

[24] Q. D. Vu *et al.*, “Methods for segmentation and classification of digital microscopy tissue images,” *Frontiers in bioengineering and biotechnology*, vol. 7, p. 433 738, 2019.

[25] D. Zhang, Y. Song, S. Liu, D. Feng, Y. Wang, W. Cai, “Nuclei instance segmentation with dual contour-enhanced adversarial network,” in *2018 IEEE 15th International Symposium on Biomedical Imaging (ISBI 2018)*, IEEE, 2018, pp. 409–412.

[26] H. Qu, Z. Yan, G. M. Riedlinger, S. De, D. N. Metaxas, "Improving nuclei/gland instance segmentation in histopathology images by full resolution neural network and spatial constrained loss," in *Medical Image Computing and Computer Assisted Intervention–MICCAI 2019: 22nd International Conference, Shenzhen, China, October 13–17, 2019, Proceedings, Part I 22*, Springer, 2019, pp. 378–386.

[27] G. Lv, K. Wen, Z. Wu, X. Jin, H. An, J. He, "Nuclei r-cnn: Improve mask r-cnn for nuclei segmentation," in *2019 IEEE 2nd International Conference on Information Communication and Signal Processing (ICICSP)*, IEEE, 2019, pp. 357–362.

[28] Z. Cheng, A. Qu, "A fast and accurate algorithm for nuclei instance segmentation in microscopy images," *IEEE Access*, vol. 8, pp. 158 679–158 689, 2020.

[29] A. Qu, Z. Cheng, X. He, Y. Li, "An integration convolutional neural network for nuclei instance segmentation," in *2020 IEEE International Conference on Bioinformatics and Biomedicine (BIBM)*, IEEE, 2020, pp. 1104–1109.

[30] H. Wang, M. Xian, A. Vakanski, "Bending loss regularized network for nuclei segmentation in histopathology images," in *2020 IEEE 17th International Symposium on Biomedical Imaging (ISBI)*, IEEE, 2020, pp. 1–5.

[31] W. Hu *et al.*, "Generative adversarial training for weakly supervised nuclei instance segmentation," in *2020 IEEE International Conference on Systems, Man, and Cybernetics (SMC)*, IEEE, 2020, pp. 3649–3654.

[32] X. Liu, Z. Guo, J. Cao, J. Tang, "Mdc-net: A new convolutional neural network for nucleus segmentation in histopathology images with distance maps and contour information," *Computers in Biology and Medicine*, vol. 135, p. 104 543, 2021.

[33] B. Bancher, A. Mahbod, I. Ellinger, R. Ecker, G. Dorffner, "Improving mask r-cnn for nuclei instance segmentation in hematoxylin & eosin-stained histological images," in *MICCAI workshop on computational pathology*, PMLR, 2021, pp. 20–35.

[34] G. Rahmon, I. E. Toubal, K. Palaniappan, "Extending u-net network for improved nuclei instance segmentation accuracy in histopathology images," in *2021 IEEE Applied Imagery Pattern Recognition Workshop (AIPR)*, IEEE, 2021, pp. 1–7.

[35] H. He *et al.*, "Cdnet: Centripetal direction network for nuclear instance segmentation," in *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2021, pp. 4026–4035.

[36] M. M. Haq, J. Huang, "Self-supervised pre-training for nuclei segmentation," in *International Conference on Medical Image Computing and Computer-Assisted Intervention*, Springer, 2022, pp. 303–313.

[37] E. H. Nguyen, H. Yang, Z. Asad, R. Deng, A. B. Fogo, Y. Huo, "Circlesnake: Instance segmentation with circle representation," in *International Workshop on Machine Learning in Medical Imaging*, Springer, 2022, pp. 298–306.

[38] E. H. Nguyen *et al.*, "Circle representation for medical object detection," *IEEE transactions on medical imaging*, vol. 41, no. 3, pp. 746–754, 2021.

[39] M. B¨ohland *et al.*, "Ciscnet-a single-branch cell nucleus instance segmentation and classification network," in *2022 IEEE International Symposium on Biomedical Imaging Challenges (ISBIC)*, IEEE, 2022, pp. 1–5.

[40] A. Mahbod, G. Schaefer, G. Dorffner, S. Hatamikia, R. Ecker, I. Ellinger, "A dual decoder u-net-based model for nuclei instance segmentation in hematoxylin and eosin-stained histological images," *Frontiers in Medicine*, vol. 9, p. 978 146, 2022.

[41] V. Magoulianitis, Y. Yang, C.-C. J. Kuo, "Hunis: High-performance unsupervised nuclei instance segmentation," in *2022 IEEE 14th Image, Video, and Multidimensional Signal Processing Workshop (IVMSP)*, IEEE, 2022, pp. 1–5.

[42] A. K. Chanchal, S. Lal, J. Kini, "Deep structured residual encoder-decoder network with a novel loss function for nuclei segmentation of kidney and breast histopathology images," *Multimedia Tools and Applications*, vol. 81, no. 7, pp. 9201–9224, 2022.

[43] W. Liu, Q. He, X. He, "Weakly supervised nuclei segmentation via instance learning," in *2022 IEEE 19th International Symposium on Biomedical Imaging (ISBI)*, 2022, pp. 1–5. doi: 10.1109/ISBI52829.2022.9761644.

[44] T. Imtiaz, S. A. Fattah, S.-Y. Kung, "Bawgnet: Boundary aware wavelet guided network for the nuclei segmentation in histopathology images," *Computers in Biology and Medicine*, vol. 165, p. 107 378, 2023.

[45] L. Rettenberger, F. Rieken M¨unke, R. Bruch, M. Reischl, "Mask r-cnn outperforms u-net in instance segmentation for overlapping cells," in *Current Directions in Biomedical Engineering*, De Gruyter, vol. 9, 2023, pp. 335–338.

[46] L. Chen, H. Li, S. K. Zhou, "Label-free nuclei segmentation using intra-image self similarity," in *International Conference on Medical Image Computing and Computer-Assisted Intervention*, Springer, 2023, pp. 673–682.

[47] P. Chen *et al.*, "Exploring unsupervised cell recognition with prior self-activation maps," in *International Conference on Medical Image Computing and Computer-Assisted Intervention*, Springer, 2023, pp. 559–568.

[48] I. Ahmad, S. M. Israr, Z. U. Islam, "A three in one bottom-up framework for simultaneous semantic segmentation, instance segmentation and classification of multi-organ nuclei in digital cancer histology," *Image and Vision Computing*, vol. 146, p. 105 047, 2024.

[49] M. Fiaz, R. M. Anwer, H. Cholakkal, "Sa2-net: Scale-aware attention network for microscopic image segmentation," *arXiv preprint arXiv:2309.16661*, 2023.

[50] A. Parulekar *et al.*, "Combining datasets with different label sets for improved nucleus segmentation and classification," *arXiv preprint arXiv:2310.03346*, 2023.

[51] Z. Li, H. Lyu, J. Wang, "Fusionu-net: U-net with enhanced skip connection for pathology image segmentation," in *Asian Conference on Machine Learning*, PMLR, 2024, pp. 694–706.

[52] A. B. Silva *et al.*, "Cnn ensembles for nuclei segmentation on histological images of oed," in *2023 IEEE 36th International Symposium on Computer-Based Medical Systems (CBMS)*, IEEE, 2023, pp. 601–604.

[53] T. Imtiaz, S. A. Fattah, M. Saquib, "Condanet: Contourlet driven attention network for automatic nuclei segmentation in histopathology images," *IEEE Access*, 2023.

[54] S. Chen, C. Ding, M. Liu, J. Cheng, D. Tao, "Cpp-net: Context-aware polygon proposal network for nucleus segmentation," *IEEE Transactions on Image Processing*, vol. 32, pp. 980–994, 2023.

[55] H. He *et al.*, "Toposeg: Topology-aware nuclear instance segmentation," in *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2023, pp. 21 307–21 316.

[56] M. Salvi *et al.*, "Cyto-knet: An instance segmentation approach for multiple myeloma plasma cells using conditional kernels," *International Journal of Imaging Systems and Technology*, vol. 34, no. 1, e22984, 2024.

[57] S. N. Rashid, M. M. Fraz, "Nuclei probability and centroid map network for nuclei instance segmentation in histology images," *Neural Computing and Applications*, vol. 35, no. 21, pp. 15 447–15 460, 2023.

[58] P. Senapati, A. Basu, M. Deb, K. G. Dhal, "Sharp dense u-net: An enhanced dense u-net architecture for nucleus segmentation," *International Journal of Machine Learning and Cybernetics*, pp. 1–16, 2023.

[59] J. Li, X. Li, M. Li, P. Yan, "A dual-path instance segmentation network based on nuclei contour in histology image," *Discover Artificial Intelligence*, vol. 3, no. 1, p. 35, 2023.

[60] Y. Zhao, X. Shao, C. Chen, J. Song, C. Tian, W. Li, "The contrastive network with convolution and self-attention mechanisms for unsupervised cell segmentation," *IEEE Journal of Biomedical and Health Informatics*, vol. 27, no. 12, pp. 5837–5847, 2023. doi: 10.1109/JBHI.2023.3310507.

[61] E. Song, B. Zhan, H. Liu, "Combining external-latent attention for medical image segmentation," *Neural Networks*, vol. 170, pp. 468–477, 2024.

[62] Q. Xu, W. Kuang, Z. Zhang, X. Bao, H. Chen, W. Duan, "Sppnet: A single-point prompt network for nuclei image segmentation," in *International Workshop on Machine Learning in Medical Imaging*, Springer, 2023, pp. 227–236.

[63] Z. Song *et al.*, "Nucleus-aware self-supervised pretraining using unpaired image-to-image translation for histopathology images," *IEEE Transactions on Medical Imaging*, vol. 43, no. 1, pp. 459–472, 2024. doi: 10.1109/TMI.2023.3309971.

[64] S. Graham *et al.*, "Conic: Colon nuclei identification and counting challenge 2022," *ArXiv preprint arXiv:2111.14485*, 2021.

[65] T. A. Tuan, T. A. Tuan, P. T. Bao, "Brain tumor segmentation using bit-plane and unet," in *Brainlesion: Glioma, Multiple Sclerosis, Stroke and Traumatic Brain Injuries: 4th International Workshop, BrainLes 2018, Held in Conjunction with MICCAI 2018, Granada, Spain, September 16, 2018, Revised Selected Papers, Part II 4*, Springer, 2019, pp. 466–475.

[66] M. Krithika alias AnbuDevi, K. Suganthi, "Review of semantic segmentation of medical images using modified architectures of unet," *Diagnostics*, vol. 12, no. 12, 2022, issn: 2075-4418. [Online]. Available: https://www.mdpi.com/2075-4418/12/12/3064.

[67] Y. Lin, Z. Wang, D. Zhang, K.-T. Cheng, H. Chen, "Bonus: Boundary mining for nuclei segmentation with partial point labels," *IEEE Transactions on Medical Imaging*, 2024.

[68] E. Baumann, B. Dislich, J. L. Rumberger, I. D. Nagtegaal, M. R. Martinez, I. Zlobec, "Hover-next: A fast nuclei segmentation and classification pipeline for next generation histopathology," in *Submitted to Medical Imaging with Deep Learning*, under review, 2024. [Online]. Available: https://openreview.net/forum?id=3vmB43oqIO.

[69] O. Ronneberger, P. Fischer, T. Brox, "U-net: Convolutional networks for biomedical image segmentation," in *Medical image computing and computer-assisted intervention–MICCAI 2015: 18th international conference, Munich, Germany, October 5-9, 2015, proceedings, part III 18*, Springer, 2015, pp. 234–241.

[70] N. Siddique, S. Paheding, C. P. Elkin, V. Devabhaktuni, "U-net and its variants for medical image segmentation: A review of theory and applications," *Ieee Access*, vol. 9, pp. 82 031–82 057, 2021.

[71] S. Ioffe, C. Szegedy, "Batch normalization: Accelerating deep network training by reducing internal covariate shift," in *International conference on machine learning*, pmlr, 2015, pp. 448–456.

[72] A. F. Agarap, "Deep learning using rectified linear units (relu)," *arXiv preprint arXiv:1803.08375*, 2018.

[73] N. Bjorck, C. P. Gomes, B. Selman, K. Q. Weinberger, "Understanding batch normalization," *Advances in neural information processing systems*, vol. 31, 2018.

[74] I. Loshchilov, F. Hutter, "Decoupled weight decay regularization," *arXiv preprint arXiv:1711.05101*, 2017.

[75] L. N. Smith, "Cyclical learning rates for training neural networks," in *2017 IEEE winter conference on applications of computer vision (WACV)*, IEEE, 2017, pp. 464–472.

[76] M. Drozdzal, E. Vorontsov, G. Chartrand, S. Kadoury, C. Pal, "The importance of skip connections in biomedical image segmentation," in *International Workshop on Deep Learning in Medical Image Analysis, International Workshop on Large-Scale Annotation of Biomedical Data and Expert Label Synthesis*, Springer, 2016, pp. 179–187.

[77] T.-Y. Lin, P. Goyal, R. Girshick, K. He, P. Dollár, "Focal loss for dense object detection," in *Proceedings of the IEEE international conference on computer vision*, 2017, pp. 2980–2988.

[78] S. S. M. Salehi, D. Erdogmus, A. Gholipour, "Tversky loss function for image segmentation using 3D fully convolutional deep networks," in *Machine Learning in Medical Imaging: 8th International Workshop, MLMI 2017, Held in Conjunction with MICCAI 2017, Quebec City, QC, Canada, September 10, 2017, Proceedings 8*, Springer, 2017, pp. 379–387.

[79] N. Abraham, N. M. Khan, "A novel focal Tversky loss function with improved attention U-net for lesion segmentation," in *2019 IEEE 16th international symposium on biomedical imaging (ISBI 2019)*, IEEE, 2019, pp. 683–687.

[80] S. Jadon, "A survey of loss functions for semantic segmentation," in *2020 IEEE conference on computational intelligence in bioinformatics and computational biology (CIBCB)*, IEEE, 2020, pp. 1–7.

[81] J. Ma *et al.*, "Loss odyssey in medical image segmentation," *Medical Image Analysis*, vol. 71, p. 102 035, 2021.

[82] A. Vaswani *et al.*, "Attention is all you need," *Advances in neural information processing systems*, vol. 30, 2017.

[83] H. Touvron *et al.*, "Llama 2: Open foundation and fine-tuned chat models," *arXiv preprint arXiv:2307.09288*, 2023.

[84] H. Touvron *et al.*, "Llama: Open and efficient foundation language models," *arXiv preprint arXiv:2302.13971*, 2023.

[85] S. Khan, M. Naseer, M. Hayat, S. W. Zamir, F. S. Khan, M. Shah, "Transformers in vision: A survey," *ACM computing surveys (CSUR)*, vol. 54, no. 10s, pp. 1–41, 2022.

[86] K. Han *et al.*, "A survey on vision transformer," *IEEE transactions on pattern analysis and machine intelligence*, vol. 45, no. 1, pp. 87–110, 2022.

[87] A. Dosovitskiy *et al.*, "An image is worth 16x16 words: Transformers for image recognition at scale," in *International Conference on Learning Representations*, 2021. [Online]. Available: `https://openreview.net/forum?id=YicbFdNTTy`.

[88] M.-H. Guo *et al.*, "Attention mechanisms in computer vision: A survey," *Computational visual media*, vol. 8, no. 3, pp. 331–368, 2022.

[89] J. Chen *et al.*, "Transunet: Transformers make strong encoders for medical image segmentation," *arXiv preprint arXiv:2102.04306*, 2021.

[90] Z. Liu *et al.*, "Swin transformer: Hierarchical vision transformer using shifted windows," in *Proceedings of the IEEE/CVF international conference on computer vision*, 2021, pp. 10 012–10 022.

[91] Y. Li, H. Mao, R. Girshick, K. He, "Exploring plain vision transformer backbones for object detection," in *European Conference on Computer Vision*, Springer, 2022, pp. 280–296.

[92] B. Chen *et al.*, "Backbone is all your need: A simplified architecture for visual object tracking," in *European Conference on Computer Vision*, Springer, 2022, pp. 375–392.

[93] H. Cao *et al.*, "Swin-unet: Unet-like pure transformer for medical image segmentation," in *European conference on computer vision*, Springer, 2022, pp. 205–218.

[94] M. Caron *et al.*, "Emerging properties in self-supervised vision transformers," in *Proceedings of the IEEE/CVF international conference on computer vision*, 2021, pp. 9650–9660.

[95] K. He, X. Chen, S. Xie, Y. Li, P. Dollár, R. Girshick, "Masked autoencoders are scalable vision learners," in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 2022, pp. 16 000–16 009.

[96] C. Wen, D. Jayaraman, Y. Gao, "Can transformers capture spatial relations between objects?" In *The Twelfth International Conference on Learning Representations*, 2023.

[97] R. Sharma, M. Saqib, C.-T. Lin, M. Blumenstein, "A survey on object instance segmentation," *SN Computer Science*, vol. 3, no. 6, p. 499, 2022.

[98] K. He, G. Gkioxari, P. Dollár, R. Girshick, "Mask r-cnn," in *Proceedings of the IEEE international conference on computer vision*, 2017, pp. 2961–2969.

[99] W. Jia, Y. Tian, R. Luo, Z. Zhang, J. Lian, Y. Zheng, "Detection and segmentation of overlapped fruits based on optimized mask r-cnn application in apple harvesting robot," *Computers and Electronics in Agriculture*, vol. 172, p. 105 380, 2020.

[100] X. Cao *et al.*, "Application of generated mask method based on mask r-cnn in classification and detection of melanoma," *Computer Methods and Programs in Biomedicine*, vol. 207, p. 106 174, 2021.

[101] R. Anantharaman, M. Velazquez, Y. Lee, "Utilizing mask r-cnn for detection and segmentation of oral diseases," in *2018 IEEE international conference on bioinformatics and biomedicine (BIBM)*, IEEE, 2018, pp. 2197–2204.

[102] R. Girshick, "Fast r-cnn," in *Proceedings of the IEEE international conference on computer vision*, 2015, pp. 1440–1448.

[103] C.-Y. Wang, A. Bochkovskiy, H.-Y. M. Liao, "Yolov7: Trainable bag-of-freebies sets new state-of-the-art for real-time object detectors," in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 2023, pp. 7464–7475.

[104] M. Carranza-García, J. Torres-Mateo, P. Lara-Benítez, J. García-Gutiérrez, "On the performance of one-stage and two-stage object detectors in autonomous vehicles using camera data," *Remote Sensing*, vol. 13, no. 1, p. 89, 2020.

[105] S. Ren, K. He, R. Girshick, J. Sun, "Faster r-cnn: Towards real-time object detection with region proposal networks," *Advances in neural information processing systems*, vol. 28, 2015.

[106] K. He, X. Zhang, S. Ren, J. Sun, “Deep residual learning for image recognition,” in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2016, pp. 770–778.

[107] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, L. Fei-Fei, “Imagenet: A large-scale hierarchical image database,” in *2009 IEEE conference on computer vision and pattern recognition*, Ieee, 2009, pp. 248–255.

[108] J. Terven, D. Cordova-Esparza, “A comprehensive review of yolo: From yolov1 to yolov8 and beyond,” *arXiv preprint arXiv:2304.00501*, 2023.

[109] Y. Zhong, J. Wang, J. Peng, L. Zhang, “Anchor box optimization for object detection,” in *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision*, 2020, pp. 1286–1294.

[110] S. Ahmad, J.-S. Kim, D. K. Park, T. Whangbo, “Automated detection of gastric lesions in endoscopic images by leveraging attention-based yolov7,” *IEEE Access*, 2023.

[111] K. Liu, Q. Sun, D. Sun, L. Peng, M. Yang, N. Wang, “Underwater target detection based on improved yolov7,” *Journal of Marine Science and Engineering*, vol. 11, no. 3, p. 677, 2023.

[112] C.-Y. Wang, H.-Y. M. Liao, I.-H. Yeh, “Designing network design strategies through gradient path analysis,” *arXiv preprint arXiv:2211.04800*, 2022.

[113] R. Zhang, “Making convolutional networks shift-invariant again,” in *International conference on machine learning*, PMLR, 2019, pp. 7324–7334.

[114] A. Chaman, I. Dokmanic, “Truly shift-invariant convolutional neural networks,” in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2021, pp. 3773–3783.

[115] G. Huang, Z. Liu, L. Van Der Maaten, K. Q. Weinberger, “Densely connected convolutional networks,” in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2017, pp. 4700–4708.

[116] D. Tian, Y. Han, B. Wang, T. Guan, H. Gu, W. Wei, “Review of object instance segmentation based on deep learning,” *Journal of Electronic Imaging*, vol. 31, no. 4, pp. 041 205–041 205, 2022.

[117] J. Li *et al.*, “A systematic collection of medical image datasets for deep learning,” *ACM Computing Surveys*, vol. 56, no. 5, pp. 1–51, 2023.

[118] U. Schmidt, M. Weigert, C. Broaddus, G. Myers, “Cell detection with star-convex polygons,” in *Medical Image Computing and Computer Assisted Intervention–MICCAI 2018: 21st International Conference, Granada, Spain, September 16-20, 2018, Proceedings, Part II 11*, Springer, 2018, pp. 265–273.

[119] M. Weigert, U. Schmidt, R. Haase, K. Sugawara, G. Myers, “Star-convex polyhedra for 3d object detection and segmentation in microscopy,” in *Proceedings of the IEEE/CVF winter conference on applications of computer vision*, 2020, pp. 3666–3673.

[120] M. Stevens, A. Nanou, L. W. Terstappen, C. Driemel, N. H. Stoecklein, F. A. Coumans, “Stardist image segmentation improves circulating tumor cell detection,” *Cancers*, vol. 14, no. 12, p. 2916, 2022.

[121] T.-Y. Lin *et al.*, “Microsoft coco: Common objects in context,” in *Computer Vision–ECCV 2014: 13th European Conference, Zurich, Switzerland, September 6-12, 2014, Proceedings, Part V 13*, Springer, 2014, pp. 740–755.

[122] K. Sun, B. Xiao, D. Liu, J. Wang, “Deep high-resolution representation learning for human pose estimation,” in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 2019, pp. 5693–5703.

PUBLICATIONS FROM THE THESIS

Conference Papers

1. Yildiz, Serdar, Abbas Memiş, and Songül Varlı. "Nuclei segmentation in colon histology images by using the deep CNNs: a U-net based multi-class segmentation analysis." 2022 Medical Technologies Congress (TIPTEKNO). IEEE, 2022.

2. Yıldız, Serdar, Abbas Memiş, and Songül Varlı. "Nuclei Instance Segmentation in Colon Histology Images with YOLOv7." International Conference on Advanced Engineering, Technology and Applications. Cham: Springer Nature Switzerland, 2023.

3. Yıldız, Serdar, Abbas Memiş, and Songül Varlı. "Size-based adaptive instance pruning for refined segmentation of cell nuclei in histology images." 2023 31st Signal Processing and Communications Applications Conference (SIU). IEEE, 2023.

4. Yıldız, Serdar, Abbas Memiş, and Songül Varlı. "Segmentation of Cell Nuclei in Histology Images with Vision Transformer Based U-Net Models" 2024 32nd Signal Processing and Communications Applications Conference (SIU). IEEE, 2024.

5. Yıldız, Serdar, Abbas Memiş, and Songül Varlı. "Semantic and Instance Segmentation of Multi-organ Cell Nuclei Using Deep Learning Based Methods" 2024 32nd Signal Processing and Communications Applications Conference (SIU). IEEE, 2024.