
**Nanyang Technological University**

School of Computer Science and Engineering

# Segmenting Glomeruli Functional Tissue Units in Human Kidneys

*CX4041 - Machine Learning - G61*

Authors:

Xiao Lingao - U2023253B

Dong Luojie - U2022869B

Sirin Nadir Kaan - N2304005H

Jose Antonio Franca - N2304058L

Heng Wei Jie - U2023731J

November 2023

# 1 Introduction

With its remarkable 37 trillion cells, the human body offers a vast and complex landscape that invites cellular-level mapping. Such an understanding is not only a scientific curiosity but also a potential catalyst for groundbreaking advancements in medical and biological research. At the start of machine learning for medical usage, one standout example is the application of artificial intelligence to enhance cancer care. Among these pioneering efforts stands the IBM Watson for Oncology, a cognitive computing system developed in collaboration with Memorial Sloan Kettering Cancer Center (MSK) introduced in 2013 [1]. This system harnesses the power of machine learning and natural language processing to revolutionize the way oncologists approach treatment decisions for cancer patients.

In a similar spirit to the Human Genome Project, which decoded our DNA [2], the Human BioMolecular Atlas Program (HuBMAP) has emerged as a major project. Backed by the National Institutes of Health (NIH), its goal is to facilitate a structure to study the human body at the cellular scale. HuBMAP is poised to create a paradigm shift in how we understand the relationship between cellular and tissue structures and their functions. It promises to deliver invaluable insights, benefiting a wide spectrum of stakeholders, including academic researchers, and pharmaceutical developers, among others.

The HuBMAP project involves several detailed case studies focusing on different parts of the human body. In particular, our project will concentrate on the "Hacking the Kidney" program, aiming to create a detailed map of the human kidney. This program challenges participants to identify and characterize functional tissue units (FTUs) across various tissue samples. An FTU is a cluster of cells surrounding a capillary where each cell is within a reachable diffusion distance from its neighbors. The primary goal here is to devise an effective method to detect glomeruli FTUs. This will be instrumental in shaping the tools, resources, and maps required to understand how cellular interconnections impact individual health.

# 2 Experiment with U-Net

In our methodological approach, we started with a comprehensive set of 15 high-resolution TIFF images. These images were meticulously processed into smaller, manageable tiles of 512x512 pixels. A thorough analysis of these segmented tiles revealed a subset of tiles that did not contribute to our objectives, labeled as 'invalid tiles'. In our commitment to efficiency and precision, we strategically eliminated these non-essential tiles, focusing exclusively on those containing relevant sections of the target object, as verified by the ground truth mask. This careful refinement resulted in a streamlined dataset , each of 3x512x512 dimensions, optimizing our training process by reducing the image count by over 40%. This step not only ensured more focused training but also significantly enhanced the overall efficiency of our experiment.

## 2.1 Preliminaries

Grasping the nuances of the glomerulus segmentation task needs a clear understanding of its specific category within segmentation tasks. In our case, we are engaged in instance segmentation, a task focused on identifying and reporting the foreground object without the need for further classification. The U-Net architecture [1] emerges as the most prominent network choice for this specific kind of task, renowned for its effectiveness in precisely delineating the object of interest from the background. This makes U-Net not just a typical choice but a strategically optimal one for the intricacies of instance segmentation in our study to unravel nuanced patterns within the medical imagery of glomeruli with unprecedented clarity and accuracy.

U-Net is distinguished by its use of multi-scale intermediate feature maps, a core aspect of its architecture. As a fully convolutional neural network (FCN), U-Net's standout feature is its sophisticated integration of these intermediate feature maps. They play a crucial role in capturing information across multiple scales and retaining detailed nuances during segmentation.

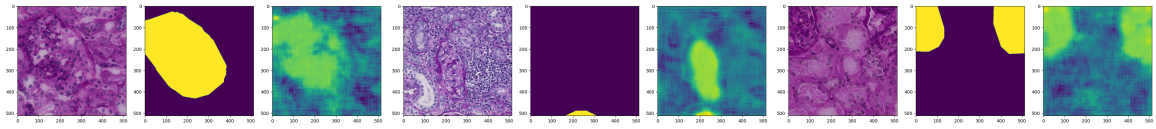
The architecture of U-Net is divided into an encoder and a decoder. The encoder is responsible for extracting high-level features, while the decoder reconstructs a pixel-wise segmentation map that aligns with the spatial resolution of the input image. Linking these two segments are skip connections, which facilitate the blending of low-level and high-level features for more accurate segmentation.

These skip connections are integral to embedding the intermediate feature maps within the U-Net architecture. As the encoder processes the image, it generates feature maps at varying levels of abstraction and spatial resolution during the downsampling phase. These maps are then concatenated with corresponding layers in the decoder. This process effectively merges broader contextual information with high-resolution details, significantly enhancing the

model's accuracy in producing precise segmentations. This blend of depth and detail is key to the effectiveness of the U-Net architecture in complex segmentation tasks.

A notable reference is the work of Taneshi *et al.* [3], who applied the U-Net architecture in their study on glomerulus segmentation.

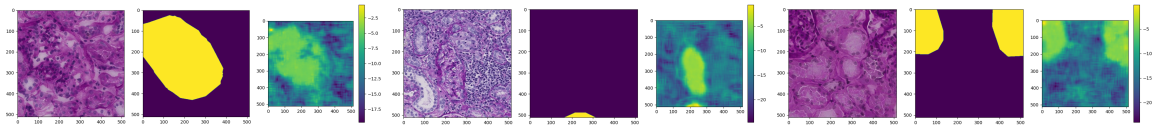
## 2.2 Basic U-Net training and Challenges



(a) Validation Image 1 (b) Validation Image 2 (c) Validation Image 3

Figure 1: Comparative Analysis of Basic U-Net Training.

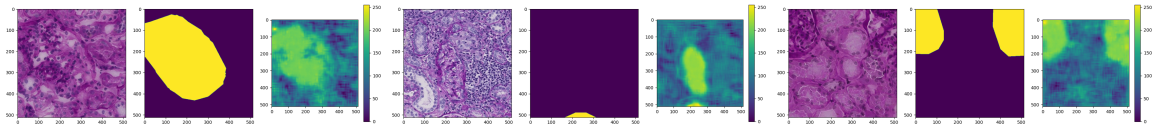
Figure 1 displays the U-Net model's segmentation results. This figure comprises three distinct validation images, each broken down into three parts. On the left of each subfigure is the original image, the middle shows the actual mask (ground truth), and the right part depicts the mask as predicted by the model. From these images, it's evident that the model's predictions align with parts of the actual mask to some degree. However, as illustrated in Figure 1 (b), the mis-segmentation of certain cells suggests an overfitting issue within the model.



(a) Validation Image 1 (b) Validation Image 2 (c) Validation Image 3

Figure 2: Unnormalized Mask Visualization: Intensity Variations Across Validation Images.

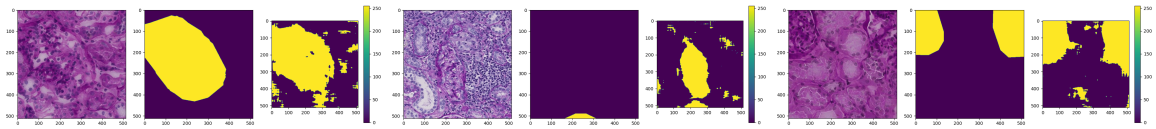
Figure 2 displays images alongside a color bar that reflects the intensity of the mask. It's observed that the value ranges differ across each validation image.



(a) Validation Image 1 (b) Validation Image 2 (c) Validation Image 3

Figure 3: Normalized Mask Visualization: Intensity Variations Across Validation Images.

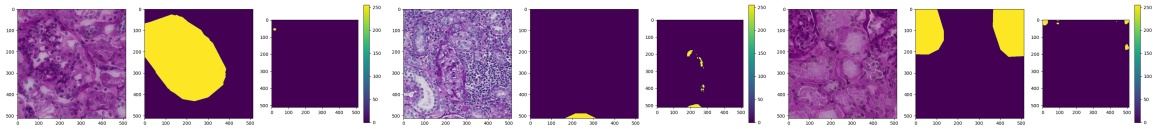
Figure 3 demonstrates that after normalization, the predicted masks appear visually consistent across images, confirming the effectiveness of our normalization process.



(a) Validation Image 1 (b) Validation Image 2 (c) Validation Image 3

Figure 4: Binary Mask with a Threshold of 128.

Figures 1-3 present the predicted masks as "attention maps," where each pixel value represents a probability. To convert these into binary segmentation masks, we must set a threshold to distinguish mask values. Deciding on the appropriate threshold value poses a challenge. Figure 4, which sets the threshold at a 50% ratio (256*0.5 = 128), shows significant noise, highlighting the complexity of this decision.



(a) Validation Image 1 (b) Validation Image 2 (c) Validation Image 3

Figure 5: Binary Mask with a Threshold of 220.

To minimize noise, we raised the threshold to 220, as illustrated in Fig. 5. This, however, led to unsatisfactory segmentation, with many targets not accurately segmented. This underscores the criticality of selecting an appropriate threshold. A key challenge here is avoiding overfitting, as using a fixed threshold value may inadvertently lead to overfitting to that specific value. To tackle this, we explore four distinct approaches to address overfitting in the model training process:

1. *Learnable Threshold* — **Hyper-parameter Perspective**
2. *Data Augmentation* — **Dataset Perspective**
3. *Mixup* — **Training Perspective**
4. *Deep Fourier Up-Sampling* — **Model Perspective**

### 2.2.1 Learnable Threshold

As discussed in the previous section, the choice of threshold value is crucial for effective segmentation. Manual selection of this threshold may lead to overfitting with respect to the test sets. In response, we embrace a strategy akin to Meta-Learning, as described in Hospedales et al. [4], which is fundamentally about learning how to learn. An example of this can be seen in the work of Cazenavette et al. [5], where the learning rate itself is treated as a learnable parameter. Following this principle, we also designate the threshold as a learnable parameter, optimizing it in tandem with the model throughout the training process.

To render the threshold learnable, it's essential to configure the parameter to accept gradients. This necessitates an optimizer distinct yet similar to the one used for the model to facilitate its optimization. Our experiments revealed that the initial value of this threshold significantly influences performance. Setting it carefully is crucial; an inappropriate initial value can lead to a challenging learning process. Moreover, the learning rate assigned to this learnable threshold must be considerably low to ensure smooth convergence.

Upon integrating the learnable threshold, we observed a slight improvement in performance, yet overfitting persisted. This led us to consider whether the root of the problem might lie within the dataset itself.

### 2.2.2 Data Augmentation

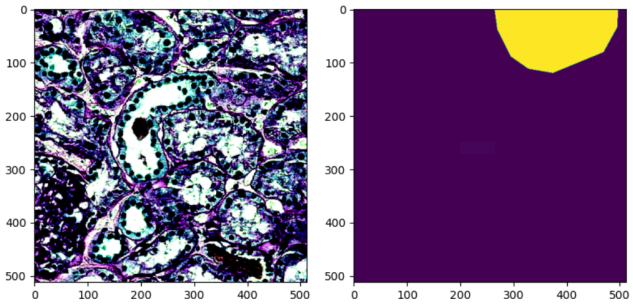



Figure 6: Visual Depiction of Misalignment Between Image and Corresponding Mask.

In the course of our development, we encountered a substantial technical challenge related to image augmentations, particularly in ensuring uniform transformations between images and their respective masks. This challenge was most pronounced in semantic segmentation tasks, where precise pixel-to-label alignment is critical. Our initial attempts using standard PyTorch transformations, such as flips and rotations, resulted in discrepancies when applied separately to images and masks, leading to misalignments (See Fig. 6). Such inconsistencies are detrimental as they compromise the essential pixel-level correspondence between the image and its mask, a factor vital for the accuracy and reliability of our model training process.

To resolve this alignment challenge, we adopted the `Albumentations` library, a sophisticated tool tailored for image augmentation. Its standout feature is the simultaneous application of identical transformations to both the image and its corresponding mask, thus preserving their spatial alignment. This means, for instance, when an image undergoes a horizontal flip, the `Albumentations` library ensures the mask undergoes a precisely matched flip. This capability of synchronized transformations is a critical advantage that elevates `Albumentations` above standard PyTorch transforms, which do not inherently support this feature.

Incorporating the Albumentations library into our workflow effectively resolved the alignment issues we faced. With this integration, transformations such as resizing, flipping, or normalizing are applied to both images and masks, ensuring alignment between the pairs. This augmentation process is vital for the precision of our model, particularly in tasks that demand an intricate spatial analysis. The reliability and consistency achieved through Albumentations have been instrumental in enhancing the overall accuracy and performance of our model in complex imaging tasks.

We also encountered an issue regarding efficiency. While advanced transformations like Affine Transformation and Color Jittering are available, they significantly increase CPU load and slow down image processing. These transformations, therefore, become bottlenecks in the training process. To maintain minimal computational resource usage, we opted for basic but effective transformations, specifically flipping and rotating. Flipping images horizontally or vertically, combined with rotation, exposes our model to various orientations of the same object, aiding in its generalization capabilities. This method is especially useful in image recognition tasks, as it trains the model to identify objects regardless of their orientation, thereby enhancing its performance on novel, unseen data.

### 2.2.3 Mixup Methodology


Figure 7: Mixup Technique Visualization.

In addition to data augmentation, we utilized the training mixup technique [6]. This involves blending different images and their corresponding labels to create new, composite training examples. The idea involves mixing both images and their corresponding labels, aiming to produce less confident predictions and consequently reduce overfitting. For instance, given two images $A$ and $B$, we create a new image $C$ by combining $A$ and $B$ in a certain ratio, defined

by a blending factor $\alpha$. The equation for this combination is $C = \alpha \cdot A + (1 - \alpha) \cdot B$. This method not only expands the variety of our training data but also introduces complexity to the learning process. Figure 7 offers a visual representation of this concept.

For our specific task, rather than blending labels, it's necessary to blend the masks within a batch. This involves a blending factor, denoted as $\alpha$, which must be uniformly applied to both the image and its corresponding mask. Implementing this mixup technique enables the training of the model on a more diverse array of data inputs. This diversity aids in refining the model's decision boundaries and acts as a form of regularization.

### 2.2.4 Deep Fourier Up-Sampling

We observed that much of the feature engineering is performed using large networks like ResNet-101 [7], which are resource-intensive for training. Even fine-tuning these networks poses challenges due to high-resolution images. Consequently, we are exploring techniques that can enhance performance while minimizing training expenses.

Upon analyzing the U-Net structure, we identified that its key elements are the upsampling and downsampling processes, which facilitate the generation of multi-scale intermediate features. Consequently, a logical method to enhance U-Net's performance would be to refine the upsampling process. As suggested by Zhou et al. [8], the conventional method of bi-linear image interpolation can be replaced with a learnable upsampling module, albeit at the expense of additional parameters. This advanced upsampling module, named Deep Fourier Up-Sampling [8], conducts the upsampling operation in the frequency domain.

Integrating U-Net with Deep Fourier Up-Sampling, we significantly improve the quality of low-resolution images, a process known as image super-resolution. This approach begins with creating pairs of low and high-resolution images for training purposes. U-Net is trained to extract features from these low-resolution images. These extracted features are then fed into the Deep Fourier Up-Sampling module, which enhances their spatial resolution and brings in finer details. The fusion of U-Net's feature extraction efficiency and the spatial refinement provided by Deep Fourier Up-Sampling results in high-resolution images that are not only visually superior but also rich in essential details. This makes them highly effective for further stages in the image processing workflow.

We highlight that the Fourier upsampling layers include learnable parameters, which are refined during the model's training process. This optimization entails extra computational resources, yet, in comparison to transitioning to a larger network, it demands significantly fewer training resources to enhance model performance. Essentially, this approach balances efficiency with performance, representing a strategic trade-off.

## 2.3 Insight: Bad Ground Truth

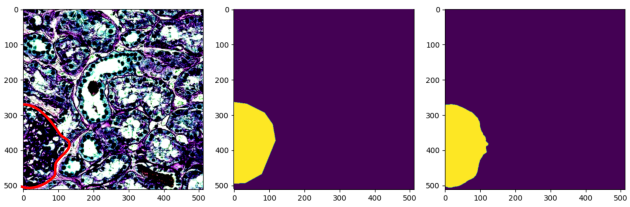
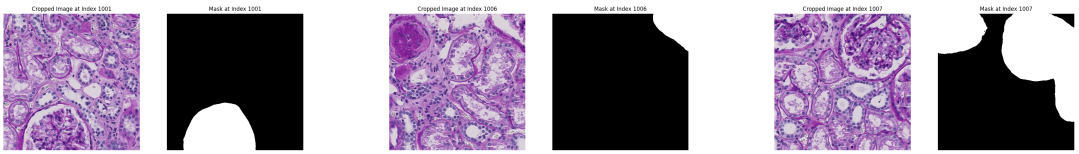


Figure 8: Enhanced Performance through Integrated Techniques: This composite showcases the superior accuracy of the predicted mask over the ground truth. Notably, the red curve serves as a visual indicator, distinctly outlining the segmentation boundary.

Figure 8 showcases the outcome when applying all the techniques discussed. The leftmost image is the original one, characterized by a blend of vivid colors and intricate patterns, a result of normalization applied exclusively to valid tiles. Here, 'valid tiles' refer to those containing the target of interest. The middle image represents the ground truth mask from the training dataset. This mask is formed by polygon indices, yet its detail is somewhat coarse, as evidenced by the noticeable presence of many straight lines. The right panel displays the prediction made by our model. Here, the shape's outline is noticeably sharper and more accurately defined than the ground truth shown in the middle panel. This highlights our model's proficiency in capturing finer details, suggesting its effectiveness in delineating object boundaries within the image.

## 2.4 Evaluating U-Net

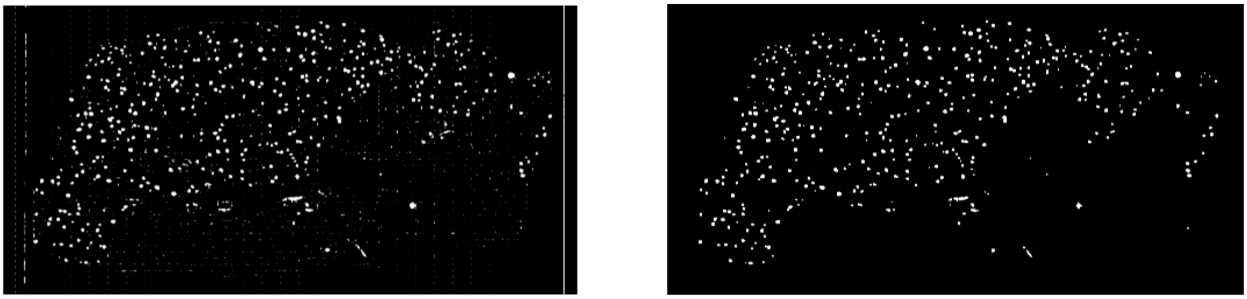


(a) Test Image Crop 1001 (b) Test Image Crop 1006 (c) Test Image Crop 1007

Figure 9: Series of Test Image Crops from Test Image: '2ec3f1bb9'

An intriguing aspect of our test set is its normalization process. For the training set, we initially focus on image crops that contain targets as indicated by the ground truth mask, discarding all crops devoid of valid targets. The normalization statistics are then calculated based on such a dataset. In contrast, for the test set, where many crops might be invalid, we employ the training dataset statistics for normalization. This approach is advisable only when the training and test images are closely aligned and free from domain gaps, a condition fortunately met by our dataset. The resulting visual representation of the test set is quite satisfactory, as can be seen in Figure 9.

## 2.5 Denoising Final Prediction



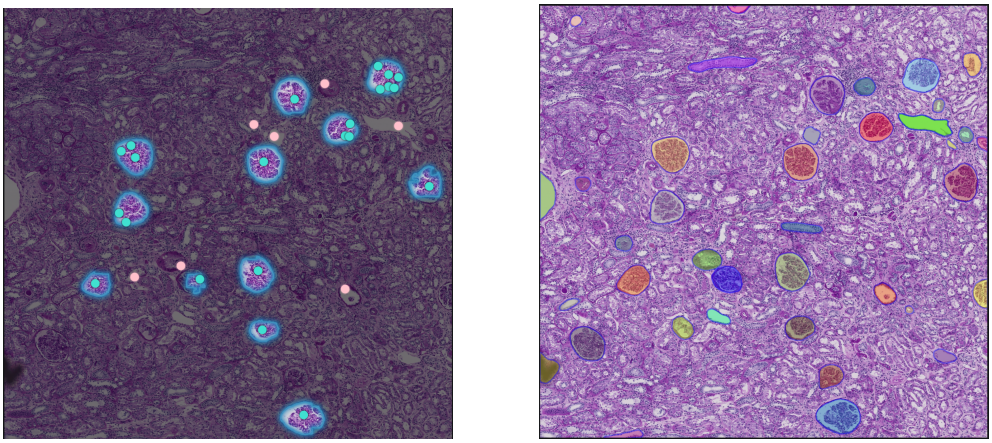
(a) No denoising process. (b) Denoised image.

Figure 10: Illustration of the effect of denoising.

Figure 10 illustrates the contrast in segmentation outcomes with and without employing a denoising technique. Image (a) depicts the initial segmentation, marred by numerous incorrectly segmented specks—these are the bright dots mistakenly marked as relevant entities. Such misidentifications can result in over-segmentation, falsely flagging minuscule cells or even insignificant noise as key features. In image (b), the denoising has been applied, evidently rectifying much of the problem. The marked reduction in these superfluous specks suggests that the denoising has effectively diminished false positives. Consequently, the segmentation becomes more precise, reducing the chance of erroneously segmented tiny or irrelevant objects, thereby enhancing the model's accuracy in discerning actual objects of interest.

# 3 Attempt on SAM

In the standard workflow for this type of task, the sequence is detection followed by segmentation as outlined by Jha et al. [9]. We propose an alternative method that utilizes the robust capabilities of pre-trained foundational models. The recent introduction of Segment Anything (SAM) [10] provides an innovative opportunity. Our approach modifies the traditional **detection + segmentation** paradigm into a **segmentation + classification** pipeline. Essentially, we aim to harness the zero-shot capabilities of SAM, thereby transforming the task into a more straightforward image classification problem.



(a) 'Segment Everything' Mode (b) Result of the Online Demo.

Figure 11: Original vs. Demo Output. Access the interactive demo here.

In our preliminary investigations, we observed that SAM effectively identifies the cellular structures. Figure 11 demonstrates that the online demo of SAM achieves high-quality cell segmentation using a **zero-shot** approach.

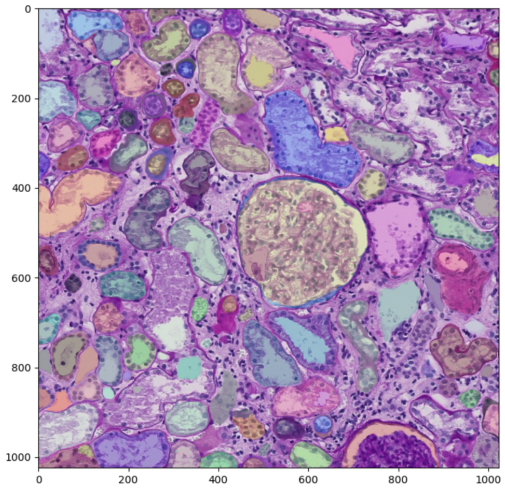



Figure 12: Segmentation by the Local SAM Model.

We replicated the SAM model, selecting the most extensive version available, which employs a ViT-Huge backbone. Yet, the outcomes did not quite match the quality of the online demo, a concern echoed by others in the community as per discussions found at GitHub issue 612. Despite this, the local model successfully detects all targeted cells, even if the delineation isn't as precise as the demo's. The subsequent step involves image preparation and the development of a streamlined classification model. In the inference stage, we leverage SAM to segment the cells, followed by our classifier to identify the targets. The primary obstacle we encounter is the constrained computational resources, which the following sections will address as we explore an efficient solution.

## 3.1 Saving Computational Power

Confronted with the challenge of limited resources for training or fine-tuning the SAM model, we adopt two practical strategies: one focuses on the model architecture, and the other on image preprocessing. To address the resource constraints from the image perspective, we introduce an image tiling approach. The rationale for segmenting the images into smaller tiles is twofold: firstly, it helps mitigate the computational load by reducing the image size that the model needs to process; secondly, it allows us to bypass the irrelevant areas in the scans.

The irregular contours of kidney samples result in original scan images laden with expansive white spaces, occasionally accounting for about 50% of the area, as evidenced in Figure 13. Black edges are a common additional feature in these scans. Such regions are categorized as "blank" for their lack of pertinent data. Segmenting the full-size images into $1024 \times 1024$ pixel tiles has facilitated a reduction in computational demand; tiles overlaying these blank spaces are identified and eliminated. Further, tiles predominantly white or black are excluded, recognizing that upon close examination, isolated elements—potentially solitary cells or sample impurities—may populate these zones. These sparse elements are often insufficient to constitute a FUT or are extraneous and can be disregarded.

We employed an algorithm to assess a tile's content value: it computes the proportion of white and black pixels and their relative ratio within each tile. A tile is classified as unproductive and thus omitted from subsequent segmentation processes if these metrics surpass predefined thresholds. These thresholds are delicately selected to balance computational efficiency and precision. While more lenient thresholds effectively reduce resource usage, they risk excluding tiles containing valuable FUTs.

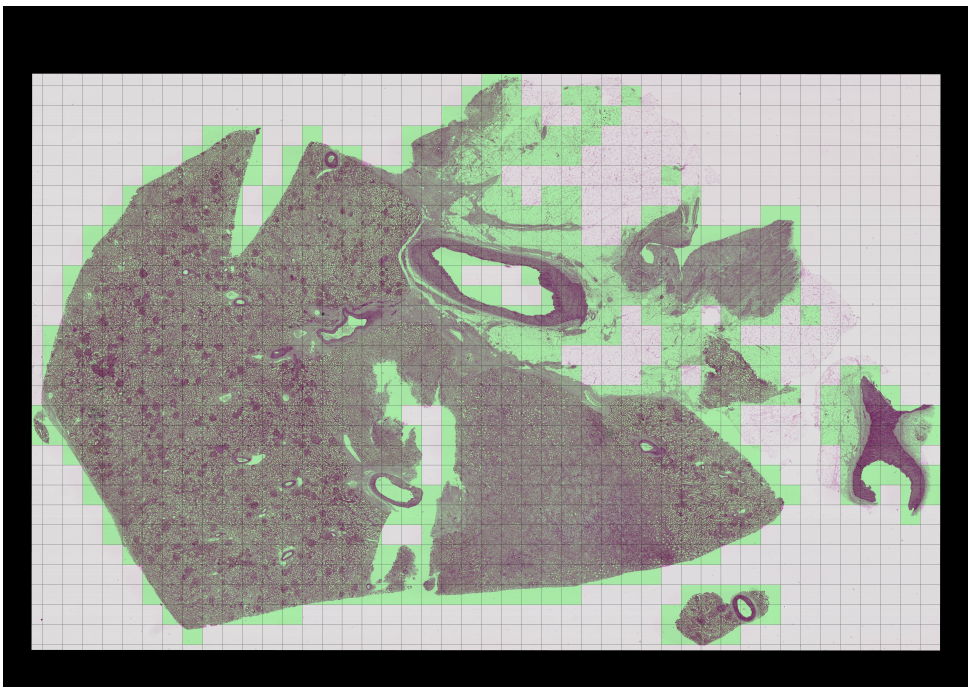


Figure 13: The original large tiff image is segmented into $1024 \times 1024$ pixel tiles, with only 880 of 1800 tiles identified as containing pertinent data, termed 'valid tiles' and indicated by green masking. Exclusion of 'invalid' tiles yields a computational resource saving of over 50%.

Even with a 40% reduction in images thanks to pre-processing, SAM's inference remains slow-moving, attributed to its inability to process batches simultaneously. Consequently, preparing the training set takes an estimated 40 days, with an additional 10 days required for processing the test images.

## 3.2 Segment with Text Prompt

In addressing the above-mentioned problem, our investigation into SAM variants led us to discover Fast-SAM, a notably efficient alternative. Fast-SAM incorporates YOLO-v7 as its core network, significantly reducing inference time. Additionally, the implementation of batch inferencing by the author markedly enhances processing speed. However, a notable discrepancy in performance between SAM and Fast-SAM is observed, particularly with medical images. As depicted in Fig. 14, the straightforward application of Fast-SAM yields subpar results. A critical observation is the failure to segment the prominent Glomerulion the right side of the image. Contrarily, the image illustrates the segmentation of the entire kidney slice. This presents a significant challenge, as accurate detection of at least the Glomeruliis essential for classification, regardless of segmentation quality. To address this, we propose two potential solutions.

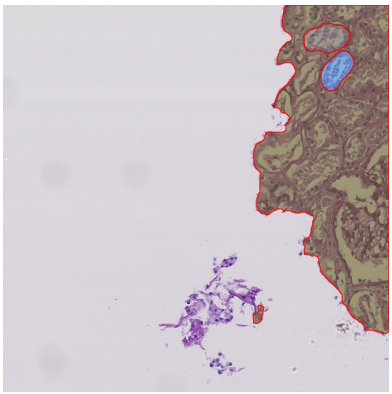
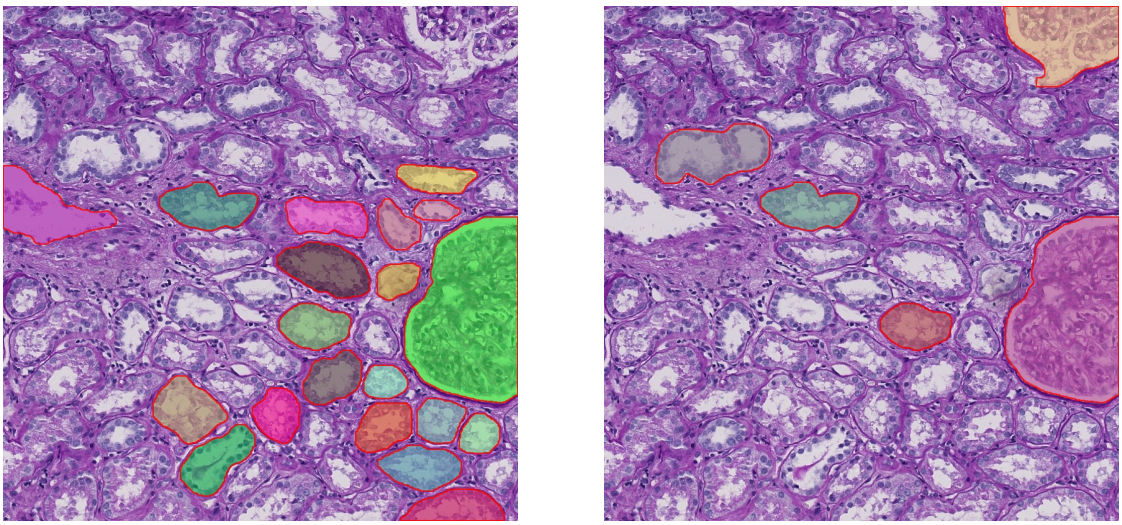


Figure 14: Default Fast-SAM model.



(a) After hyper-parameter search. (b) Segment with aids of text prompt.

Figure 15: Results on Fast-SAM. Our version with batching text prompt.

The first approach is hyper-parameter searching, especially the `confidence` and `iou` values. Adjusting `confidence` affects how certain the model must be about its predictions, impacting precision and recall balance. A higher `confidence` threshold reduces false positives but may miss true positives. `iou` (Intersection over Union) gauges the accuracy of object localization. A higher `iou` value ensures that only predictions with a strong overlap with ground truth are considered correct, enhancing the model's localization accuracy. Optimal tuning of these parameters is vital for maximizing segmentation performance. Figure 15 (a) delivers the result of hyper-parameter tuning, which captures the most cells but still missing the Glomerulion the top right corner. This one is harder to capture since it only has a small part of showing in the image.

The second approach is about tweaking Fast-SAM with detailed text prompts, for instance, "large glomerulus cells (kidney), which cells contain dark materials in the middle part." We're facing a bottleneck with how the model processes these prompts – it's doing them one by one, which isn't efficient. To tackle this, we're shifting towards batch processing. This means the model will handle a bunch of prompts and images all at once, instead of taking them on one at a time. This shift is a big deal because it speeds things up a lot. It's especially handy when we have to work with many images or when the prompts get complex. By doing this, Fast-SAM becomes quicker and gets better at figuring out what's in the images based on our prompts. It's like making it multitask smartly, so we get the results we need faster and more accurately. Fig. 15 (b) provides the visualization of using text-prompt with batching inference (5 masks). In this case, both Glomeruliare successfully captured.

### 3.3 Converting to Classification Task

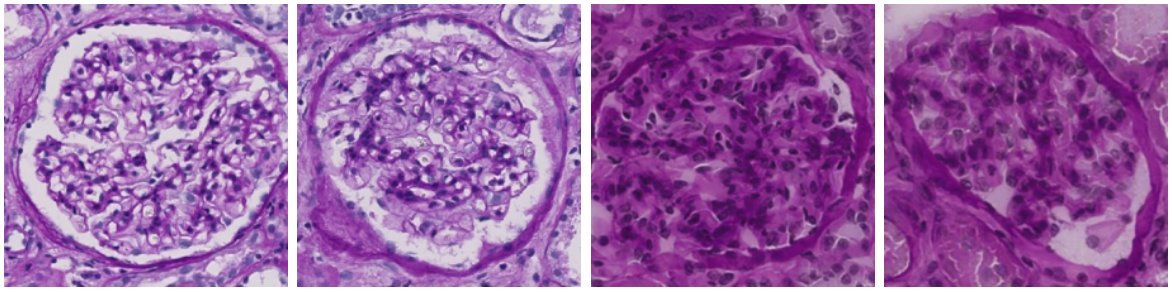


Figure 16: Visualization of training images (Positive ones).

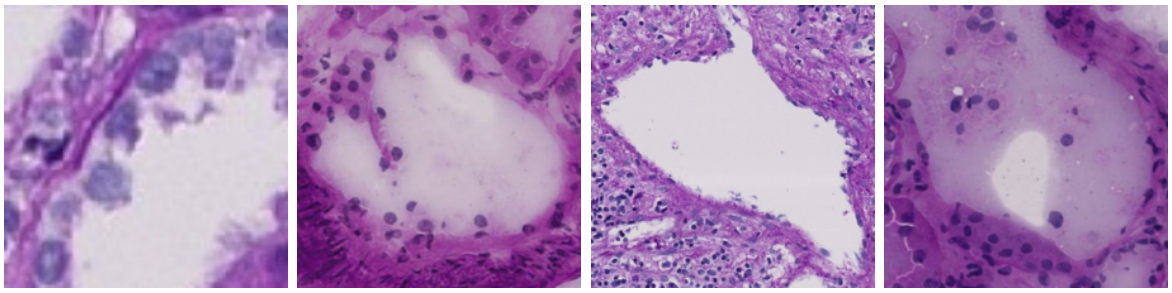


Figure 17: Visualization of training images (Negative ones).

After segmenting the cells, we individually prepare each one as training images. We center each cell within the image frame and scale it to a resolution of 224x224 pixels, aligning with the common practice for ImageNet dataset images. This process is facilitated by the knowledge that the cells of interest, *glomerulus cells*, are typically large. As a result, when they are centered and resized to 224x224, they appear normal-sized (as shown in Figure 16). This is in contrast to the first image in Figure 17, which depicts a small cell that was incorrectly detected and segmented by Fast-SAM. Consequently, when this small cell is scaled up to a higher resolution, it becomes much more blurred.

With the assembly of these positive and negative sample pairs, we effectively compile a dataset labeled with two categories: 0 for negative and 1 for positive. Training a binary classification network on such a dataset enables us to isolate images containing true glomerulus cells. Once trained, we can then determine the validity of the segmented masks and retain only those that are accurate. This strategy simplifies the complex task of segmentation into a more straightforward task of classification. This pipeline is feasible particularly when the pretrained model exhibits robust zero-shot segmentation capabilities, operating under the presumption that the segmented masks are accurate. Given the rapid advancements in large-scale foundational models, we are confident that this pipeline can yield outstanding outcomes.

# 4 Conclusion

In this study, we tackled the complex task of segmenting glomeruli in human kidneys, navigating challenges in machine learning and image processing. By employing advanced techniques like U-Net, learnable thresholds, data augmentation, mixup, and Deep Fourier Up-Sampling, we overcame issues such as overfitting and enhanced model performance. A key innovation was adapting the Segment Anything Model (SAM) and its variant, Fast-SAM, for efficient image processing, transforming the segmentation task into a simpler classification problem. This approach, leveraging the zero-shot capabilities of SAM, demonstrated a novel and effective method in medical imaging, highlighting the intersection of creativity and technical proficiency in overcoming computational limitations.

# References


[1] O. Ronneberger, P. Fischer, and T. Brox, "U-net: Convolutional networks for biomedical image segmentation," 2015.

[2] F. Collins and L. Fink, "The human genome project," Alcohol Health Res World, vol. 19, no. 3, pp. 190–195, 1995.

[3] Taneshi, "Glomerular segmentation of kidney images with u-net." `https://github.com/taneishi/UNet_Glomerulus`, 2020.

[4] T. Hospedales, A. Antoniou, P. Micaelli, and A. Storkey, "Meta-learning in neural networks: A survey," IEEE transactions on pattern analysis and machine intelligence, vol. 44, no. 9, pp. 5149–5169, 2021.

[5] G. Cazenavette, T. Wang, A. Torralba, A. A. Efros, and J.-Y. Zhu, "Dataset distillation by matching training trajectories," in Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, 2022.

[6] H. Zhang, M. Cisse, Y. N. Dauphin, and D. Lopez-Paz, "mixup: Beyond empirical risk minimization," arXiv preprint arXiv:1710.09412, 2017.

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

[8] M. Zhou, H. Yu, J. Huang, F. Zhao, J. Gu, C. C. Loy, D. Meng, and C. Li, "Deep fourier up-sampling," arXiv preprint arXiv:2210.05171, 2022.

[9] A. Jha, H. Yang, R. Deng, M. E. Kapp, A. B. Fogo, and Y. Huo, "Instance segmentation for whole slide imaging: end-to-end or detect-then-segment," Journal of Medical Imaging, vol. 8, no. 1, pp. 014001–014001, 2021.

[10] A. Kirillov, E. Mintun, N. Ravi, H. Mao, C. Rolland, L. Gustafson, T. Xiao, S. Whitehead, A. C. Berg, W.-Y. Lo, P. Dollár, and R. Girshick, "Segment anything," 2023.