Glomeruli Segmentation in Whole-Slide Images: Is Better Local Performance Always Better?

Maria Sánchez[1], Helena Sánchez[1], Carlos Pérez de Arenaza[1], David Ribalta[1], Nerea Arrarte[1], Oscar Cámara[1], Adrian Galdran[1,2]

[1] Universitat Pompeu Fabra, Barcelona, Spain
[2] Computer Vision Center, Universitat Autònoma de Barcelona, Spain.

Abstract. We consider the task of glomeruli segmentation from Whole-Slide Images (WSIs) of pathological kidneys. In particular, we compare the performance of two different encoder-decoder architectures for two tasks: local segmentation of patches extracted from a large WSI, and global segmentation of the entire image. Since segmenting high-resolution WSIs is extremely memory-demanding, a typical approach for this task is to break down these images offline, train a patch-wise segmentation model, and then use a sliding-window inference scheme to stitch back the resulting patch segmentations. Contrary to intuition, we observe in our experiments that a model with higher segmentation accuracy at the patch level can incur in large underperformance gaps at the WSI level, even more so when measuring performance as an instance segmentation problem. This work was carried out in the context of the Kidney Pathology Image Segmentation (KPIs) challenge, which took place jointly with MICCAI 2024, and the best patch-level model we present here ranked second in the final hidden test set of the competition. Code to reproduce our experiments is shared at `github.com/agaldran/kpis`.

Keywords: Kidney Pathology Image segmentation · Whole Slide Image Segmentation

1 Introduction

Glomerular disease is a significant health issue affecting the kidneys, which are essential organs responsible for filtering blood, removing waste, and maintaining electrolyte balance [13]. Each kidney contains approximately one million nephrons, the functional units that filter blood. Within each nephron, the glomerulus acts as a critical filtration barrier, preventing large molecules such as proteins and blood cells from passing into the urine while allowing waste products and excess substances to be excreted [2].

Glomerular diseases, a major cause of chronic kidney disease (CKD) that affects about 10% of the global population [11], can compromise the kidneys' ability to filter blood properly, leading to progressive kidney damage if left untreated [2]. On the other hand, early detection and treatment of glomerular diseases can be crucial to prevent kidney damage and maintain overall health.

In this context, the analysis of kidney pathology images has become increasingly important, particularly segmenting glomeruli from Whole Slide Images (WSIs).

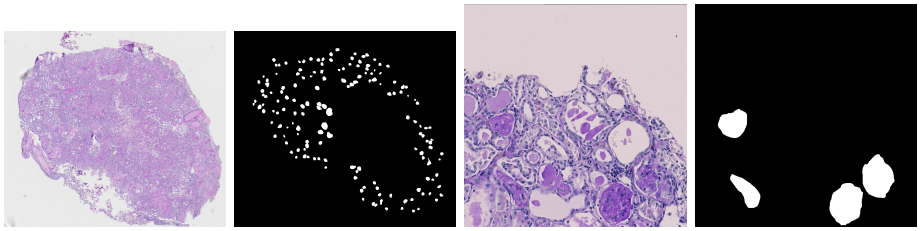
Accurate segmentation is essential for diagnosing glomerular diseases, as it allows clinicians to assess the extent of glomerular damage, evaluate treatment efficacy, and study disease progression [6,4] . Historically, the task has been performed manually by pathologists, a process that is both time-consuming and prone to variability [5]. Early automated methods relied on classical image processing techniques such as thresholding, edge detection, and template matching, but these approaches struggled with the complex variability in glomeruli shapes and tissue appearances. More recent advances in deep learning have revolutionized glomeruli segmentation, offering the ability to learn rich representations from labeled data, providing more precise and scalable solutions [1,10,8].

The KPIs challenge

The **MICCAI 2024 Kidney Pathology Image Segmentation (KPIs) Challenge** was organized to explorethe boundaries of glomeruli segmentation through a competitive framework. Participants were tasked with developing segmentation algorithms to accurately identify glomeruli from WSIs in various CKD contexts. The challenge emphasized the importance of segmenting glomeruli at a pixel level, which is useful for enabling a better understanding of CKD development. The organization made available a dataset with a wide range of image variations such as large differences in glomeruli size, shape, and structural changes due to disease states or tissue preparation techniques [7]. The challenge is divided into two distinct tasks:

1. **Patch-level Segmentation:** This task focused on segmenting glomeruli from smaller, predefined patches of the WSI. The goal was to isolate glomeruli accurately within these constrained regions, often used for training deep learning models due to the reduced computational load and focused analysis.
2. **Whole Slide Image Segmentation:** This task required to segment glomeruli across entire slide images, offering a more comprehensive view of the tissue. WSIs present additional challenges due to their large size and the diversity of tissue types, demanding algorithms that can generalize well across different CKD disease models and tissue conditions.

In the remaining of this paper, we present two segmentation approaches based on encoder-decoder architectures for both patch-level and WSI-level tasks. We will see how local patch-wise high performance does not necessarily translate into global instance-wise accuracy. Indeed, for this specif problem, the model that performed the worse at the local level achieved a widely lower performance at the WSI-level, particularly when considering the task as instance segmentation.



(a) (b) (c) (d)

Fig. 1: (a) A Whole Slide Image from the training set, and (b) the corresponding manual segmentation. (c) and (d) show a local patch extracted from (a) and its segmentation.

2 Methodology

2.1 Segmentation Model Architecture

In our analysis, we employed segmentation models based on encoder-decoder architectures for both patch-level and Whole Slide Image (WSI)-level segmentation, where the final segmentation was constructed via a sliding-window inference approach. We trained various combinations of encoders and decoders, ultimately selecting those models that demonstrated the best performance for each task based on rigorous cross-validation experiments. Our analysis below refers to two segmentation models, both sharing the same image encoder, but with different decoders, which we will refer to as ***model 1*** and ***model 2***.

For patch-level segmentation, the preferred architecture (***model 1***) to experiment with as a Feature Pyramid Network (FPN) as the encoder, and a Mix Vision Transformer (MiT) as the decoder. The FPN is a popular choice for segmentation tasks due to its ability to extract multi-scale features efficiently [9]. It was initialized using pretrained weights from the ImageNet dataset to leverage prior knowledge and improve convergence during training [15]. FPN creates a top-down architecture with lateral connections to build high-level semantic feature maps at multiple scales. The decoder in this architecture was the Mix Vision Transformer (MiT), originally proposed in the SegFormer framework [14]. MiT uses a transformer-based architecture to process and combine features extracted by the encoder. This transformer-based decoder proved effective in our experiments, especially in handling the features of patch-level inputs. The decoder takes the multi-scale features from the FPN and transforms them into a segmentation map, preserving both global and local context.

As a secondary model (***model 2***), we trained a model with a different encoder, a U-Net++ initialized again from ImageNet-pretrained weights [15]. U-Net++ builds upon the classic U-Net architecture by adding nested, dense skip connections, which enhance the model's ability to learn fine-grained details and smooth transitions between different regions. The decoder for the WSI-level segmentation task was a ResNeSt101, chosen from the timm library [15]. ResNeSt

(Split-Attention Networks) extends the traditional ResNet architecture by incorporating split-attention blocks, enabling the network to capture diverse features within a single layer.

2.2 Training Details

For Task 1, we followed a similar training procedure to obtain the training hyperparameters that worked the best and would optimize the cross-validation performance of ***model 1*** and ***model 2***. The following details summarize the key aspects of the training process:

- **Optimizer:** We used the Nadam optimizer [3], which combines the benefits of the Nesterov accelerated gradient and Adam optimizers, offering faster convergence and improved stability during training.
- **Learning Rate:** The learning rate was set to 1×10^{-4}, a value chosen after initial hyperparameter tuning to balance the speed of convergence with model performance.
- **Batch Size:** A batch size of 8 was used to manage the memory requirements while ensuring efficient gradient computation.
- **Image size:** The input images were resized to 1024×1024 pixels for ***model 1*** and 512×512 pixels for ***model 2***.
- **Number of Epochs:** ***model 1*** was trained for 20 epochs, whereas ***model 2*** trained for 60 epochs. These numbers were chosen to ensure sufficient training time and guarantee convergence while monitoring the validation loss and the dice value to avoid overfitting.
- **Loss Function:** The Dice loss function was used to handle the class imbalance in the dataset and improve segmentation performance by focusing on overlap between predicted and true segmentations. This was added to the Cross-Entropy loss, as this combination is known to provide benefits when dealing with overfitting and miscalibration

Other details available on our github repository `github.com/agaldran/kpis`.

2.3 Inference: from Local to Global segmentations

For Task 1, inference was implemented with Test-Time Augmentation (flipping horizontally or/and vertically each patch), and the final segmentation was found by averaging the prediction of the five cross-validation models. For task 2, the code made available as a template by the organizers was employed. This is, an entire WSI is broken down into smaller patches of size 2048×2048, with a stride of 1024 pixels in each direction, and then downsampled to the same resolution at which each model was trained. After this all patches are forwarded through a model to obtain the corresponding local segmentations, which are upsampled back to a 2048×2048 resolution. These patches are then stitched together into a single WSI segmentation.

3 Experimental analysis

3.1 Datasets and performance evaluation

The training and validation data were shared by the organization of the MICCAI 2024 Kidney Pathology Image Segmentation (KPIs) challenge [7]. Tissue sections were stained using Periodic acid-Schiff (PAS) to highlight cellular and structural components. Each image captures nephrons, which contain a glomerulus and a small cluster of blood vessels. The slides were digitized at Vanderbilt University Medical Center, and the digital images were annotated by experienced pathologists. The mouse kidney pathology data were available as whole slide images (WSIs) and segmented patches, provided in TIFF format (.tiff) with corresponding segmentation masks. These data were derived from four groups of mouse models: (1) normal mice, sacrificed at 8 weeks, (2) the 5/6Nx group, where mice underwent 5/6 nephrectomy and were sacrificed 12 weeks post-nephrectomy, (3) the DN group, consisting of double-knockout eNOS-/-/lepr(db/db) mice sacrificed at 18 weeks, and (4) the NEP25 group, consisting of transgenic mice expressing human CD25 selectively in podocytes, sacrificed 3 weeks after immunotoxin-induced glomerular injury.

For the first task focused on patch-level segmentation, requiring segmentation of glomeruli from image patches, 5213 images were provided for training and 1643 for validation. The second, more challenging task, involved Whole Slide Imaging (WSI) segmentation, requiring the segmentation of entire kidney slides. In Task 2, the dataset included 27 images for training and 8 for validation. The test set included an unknown number of WSIs and patches.

For evaluation purposes, the challenge organization chose the Dice Similarity Coefficient to assess the overlap between predicted segmentations and ground truth masks in Task 1. The second task, focused on WSI segmentation, was evaluated using both the Dice metric and the F1 score at the glomeruli level (instance-wise segmentation), which helped measure segmentation performance while addressing false positives and false negatives in the results [12].

3.2 Numerical results

For Task 1, the two segmentation systems were trained following a five-fold cross-validation scheme, resulting in five performance measurements. We trained both **Model 1** and **Model 2** with similar data partitions, leading to comparable experimental results. Average per-fold Dice scores resulting from this process are collected in Table 1, where we also show the mean and standard deviation for each model. It can be observed that **Model 1** appears to be more accurate in terms of segmentation overlap in every validation fold, with a higher average performance and also a lower standard deviation, indicating that training at higher resolution could be a factor in improving patch-wise performance.

Table 1: Five-fold cross-validation performance (Dice similarity score) for patch-wise segmentation, with mean and standard deviation across folds.

	Fold 1	**Fold 2**	**Fold 3**	**Fold 4**	**Fold 5**	$\mu \pm \sigma$
Model 1	**92.37**	**90.54**	**91.47**	**89.65**	**92.50**	**91.31 ± 1.09**
Model 2	**91.96**	**89.75**	**89.94**	**87.95**	**91.87**	**90.29 ± 1.67**

Once the first phase of the challenge was over, the selected models were submitted in the form of a Docker container to the organization, who run it on a hidden test set and returned the final performance to each participant. For Task 1, the first column in Table 2 shows this final score for each analyzed model. We can confirm here the preliminary conclusions drawn from our cross-validation analysis: **Model 1** achieves a higher patch-wise performance than **Model 2**, achieving a second position in the final competition ranking. It is worth noting that both models generalized in an excellent way to the unseen data at the patch level, since segmentation overlap in the test set was even higher than in the cross-validation experiments. Let us also note that the performance difference appears to be relatively substantial (94.28 vs 93.23), and similar to the gap observed on average in Table 1.

Another interesting observation can be made from the performance in the task of glomeruli segmentation from WSI. In this case, we see that **Model 2** noticeably outperforms **Model 1** with a Dice similarity score of 92.74 vs 81.94. Since both models followed the same patch-to-WSI segmentation construction method, this drop in performance appears to indicate that the accuracy of **Model 1** at the patch-level does not translate fully to global performance. The F1 score gives us a hint of what could be the case. In this metric, the performance gap is extremely increased, with **Model 1** underperforming **Model 2** by 35.01 vs 86.81. This seems to inform of a potentially large number of (small in size) false positives being generated by **Model 1**.

Table 2: Test Set Performance on the hidden test set for task 1 (patch segmentation, dice score) and task 2 (WSI segmentation, F1 score for instance-segmentation). Final competition rank in parenthesis.

	Task 1	**Task 2 - F1**	**Task 2 - Dice**
Model 1	**94.28 (2nd/24)**	**35.01 (14th/15)**	**81.94 (11th/15)**
Model 2	**93.23 (8th/24)**	**86.81 (6th/15)**	**92.74 (5th/15)**

4 Conclusions

In this paper we analyzed the performance of two different models for two histopathological image segmentation tasks: patch-wise glomeruli segmentation and its WSI version. We experimentally found that a model achieving higher performance at the patch-level might not keep its advantage when following a sliding-window WSI segmentation strategy. Overall, the ranking of **Model 1** vs **Model 2** was respectively 2nd vs 8th for Task 1, but 11th vs 5th in Task 2 when measured with the Dice score, and 14th vs 6th if we use the F1 score. Given that **Model 2** already achieved a high Dice score at the patch-level (93.23), preferring **Model 1** over **Model 2** would be pointless in this scenario. We can conclude that evaluating segmentation models both at the global and local level should be a requirement for histopathological image segmentation tasks in which WSIs are available.

Acknowledgments

A. Galdran is funded by a Ramon y Cajal fellowship RYC2022-037144-I.

Disclosure of interests

The authors have no competing interests to declare that are relevant to the content of this work.

Contributions

A. Galdran and the P53 team (M. Sánchez, H. Sánchez, C. Pérez de Arenaza, D. Ribalta) developed solutions for the competition, N. Arrarte and O. Cámara supervised the work of the P53 team, all authors contributed to the writing of this article.

References

1. Bouteldja, N., Klinkhammer, B.M., Bülow, R.D., Droste, P., Otten, S.W., Von Stillfried, S.F., Merhof, D.: Deep learning–based segmentation and quantification in experimental kidney histopathology. Journal of the American Society of Nephrology **32**(1), 52–68 (2021) 2
2. Cleveland Clinic: Glomerulonephritis (gn), `https://my.clevelandclinic.org/health/diseases/16167-glomerulonephritis-gn`, accessed: 2024-06-01 1
3. Dozat, T.: Incorporating Nesterov Momentum into Adam. In: Proceedings of the 4th International Conference on Learning Representations. pp. 1–4 4
4. Jha, A., Yang, H., Deng, R., Kapp, M.E., Fogo, A.B., Huo, Y.: Instance segmentation for whole slide imaging: end-to-end or detect-then-segment. Journal of Medical Imaging **8**(1), 014001–014001 (2021) 2

5. Jiang, L., Chen, W., Dong, B., Mei, K., Zhu, C., Liu, J., Cai, M., Yan, Y., Wang, G., Zuo, L., Shi, H.: A Deep Learning-Based Approach for Glomeruli Instance Segmentation from Multistained Renal Biopsy Pathologic Images. The American Journal of Pathology **191**(8), 1431–1441 (Aug 2021). https://doi.org/10.1016/j.ajpath.2021.05.004 2
6. Kaur, G., Garg, M., Gupta, S., Juneja, S., Rashid, J., Gupta, D., Shaikh, A.: Automatic identification of glomerular in whole-slide images using a modified unet model. Diagnostics **13**(19), 3152 (2023) 2
7. KPIs2024 - Task: Kpis2024 - task (2024), `https://sites.google.com/view/kpis2024/task?authuser=0`, accessed: 2024-06-01 2, 5
8. Leng, H., Deng, R., Asad, Z., Womick, R.M., Yang, H., Wan, L., Huo, Y.: An accelerated pipeline for multi-label renal pathology image segmentation at the whole slide image level. In: Medical Imaging 2023: Digital and Computational Pathology. vol. 12471, pp. 174–179. SPIE (Apr 2023). https://doi.org/10.1117/12.2653651 2
9. Lin, T.Y., Dollár, P., Girshick, R., He, K., Hariharan, B., Belongie, S.: Feature pyramid networks for object detection. In: Proceedings of the IEEE conference on computer vision and pattern recognition. pp. 2117–2125 (2017) 3
10. Lutnick, B., Manthey, D., Becker, J.U., Ginley, B., Moos, K., Zuckerman, J.E., Rodrigues, L., Gallan, A.J., Barisoni, L., Alpers, C.E., Wang, X.X., Myakala, K., Jones, B.A., Levi, M., Kopp, J.B., Yoshida, T., Zee, J., Han, S.S., Jain, S., Rosenberg, A.Z., Jen, K.Y., Sarder, P.: A user-friendly tool for cloud-based whole slide image segmentation with examples from renal histopathology. Communications Medicine **2**(1), 1–15 (Aug 2022). https://doi.org/10.1038/s43856-022-00138-z 2
11. Mallamaci, F., Tripepi, G.: Risk factors of chronic kidney disease progression: Between old and new concepts. J. Clin. Med. **13**(3), 678 (2024). https://doi.org/10.3390/jcm13030678, submission received: 20 December 2023 / Revised: 17 January 2024 / Accepted: 22 January 2024 / Published: 24 January 2024 1
12. Reinke, A., et al.: Understanding metric-related pitfalls in image analysis validation. Nature Methods **21**(2), 182–194 (Feb 2024). https://doi.org/10.1038/s41592-023-02150-0 5
13. Thomas, L., Huber, A.R.: Renal function–estimation of glomerular filtration rate. Clinical Chemistry and Laboratory Medicine (CCLM) **44**(11), 1295–1302 (2006) 1
14. Xie, E., Wang, W., Yu, Z., Anandkumar, A., Alvarez, J.M., Luo, P.: Segformer: Simple and efficient design for semantic segmentation with transformers. Advances in Neural Information Processing Systems **34**, 12077–12090 (2021) 3
15. Yakubovskiy, P.: Segmentation models with pytorch. `https://github.com/qubvel-org/segmentation_models.pytorch` (2023) 3