Proceedings of the 20th Conference on Computer Science and Intelligence Systems (FedCSIS) pp. 363–368
DOI: 10.15439/2025F5576
ISSN 2300-5963 ACSIS, Vol. 43

Hybrid U-Net segmentation of vessels in fundus eye images

Lesław M. Pawlaczyk
WSB Merito University
in Chorzow, Poland
ul. Sportowa 29, 41-506 Chorzów, Poland
Email: leslaw.pawlaczyk@chorzow.merito.pl

***Abstract*—We present a novel multi-stage method for colour image segmentation, with a primary focus on vessels segmentation in retinal fundus images, using a U-Net based architecture. Our approach tackles challenges posed by varying image resolutions through a coarse-to-fine segmentation pipeline. It begins with a rough segmentation at varying scales, guided by a traditional CNN and progressively refines results to find a target resolution. It culminates with detailed segmentation at a target scale with a smaller window sliding step, compared to previous stages. We train and validate our method using four publicly available datasets FIVES, DRHAGIS, HRF, and STARE - and demonstrate superior performance compared to traditional sliding window techniques. Notably, our model achieves high accuracy with relatively few training images. The entire framework is open-sourced and adaptable to a wide range of image segmentation tasks.**

***Index Terms*—U-Net, segmentation, deep learning, vessel segmentation, medical imaging, fundus images**

I. Introduction

THE process of image segmentation has been for many years a challenge, that was only partially solveable. This changed with an advent of deep learning and introduction of convoutional neural networks. In the year 2012 Krizhevsky et al [1] introduced ImageNet, which achieved a new level of image recognition accuracy. It won the ImageNet competition that year, and inspired a plethora of other researchers to improve upon it. This milestone in image classification allowed a development of new class of segmentation algorithms.

One of the challenges in research was to segment biomedical images. It was partially solved by Ronneberger et al. [2] by introduction of *U-Net* architecture. The *U-Net* is designed to work with very few training images and to yield more precise segmentations. It consists of a contracting path to capture context and a symmetric expanding path that enables precise localization. However it comes with drawbacks, such as sensitivity to input image resolution and segmentation window position, and high demand for GPU memory which is limiting the segmentation window size (*WS*).

In this article, we address challenges in image segmentation by proposing a multi-step framework for training a *U-Net* adaptable to varying image resolutions. We evaluate segmentation results at multiple scales to select the most effective output and analyze how resolution affects performance when the model is applied at different scales. We also assess the annotation effort required to achieve satisfactory results. Experiments on four public vessel segmentation datasets enable us to test how well a *U-Net* trained on one dataset generalizes to others.

The article has six chapters. The first introduces the topic. The second reviews recent work on *U-Net* segmentation in medical imaging. The third explains our approach with illustrations. The fourth describes our experiments and results. The fifth highlights key findings. The final chapter states there are no competing interests.

The main contributions of this work are:

- A multi-scale rough segmentation step that processes images at multiple predefined scales using a sliding window approach.
- A *CNN*-based scoring mechanism that selects the optimal segmentation scale for each image based on predicted segmentation quality.
- Experimental validation of cross-dataset generalization, demonstrating that models trained on one dataset can perform well on others.
- An analysis of how the number of training images affects segmentation quality, providing insight into the annotation effort required for satisfactory performance.
- An open-source implementation of the method, available on GitHub [3].

II. Related works

Ren et al. [4] proposed an improved *U-Net*-based method for retinal vessel image segmentation, enhancing the original architecture to achieve greater accuracy and robustness. Their improvements include modifications to both the network structure and the training process. Liu et al. [5] introduced a three-path *U-Net* model for retina image segmentation, which leverages multiple pathways to capture features at different levels, resulting in improved segmentation performance. Das et al. [6] evaluated the performance of the standard *U-Net* for retinal blood vessel segmentation, confirming its effectiveness for this task. Yun et al. [7] proposed a Multi-Path Recurrent *U-Net* for segmenting retinal fundus images, incorporating both multiple paths and recurrent units to enhance feature extraction. Similarly, Huang et al. [8] developed an improved *U-Net* architecture based on residual modules, achieving increased

Topical area: Advanced Artificial Intelligence in Applications

robustness and better segmentation results in retinal vascular images.

Many existing studies aim to enhance the performance of *U-Net* across a range of applications. A common theme among these efforts is the focus on segmenting disconnected or localized objects [9], often overlooking use cases like vessel segmentation, where structures are continuous and span the entire image. Additionally, relatively few works address the challenge of handling images with varying resolutions - a critical factor in real-world medical imaging scenarios. Notably, many proposed improvements to *U-Net* result in only marginal performance gains, typically in the range of 1–2% absolute accuracy, as the original *U-Net* already performs strongly, often achieving over 95% accuracy under favorable conditions.

III. Methodology

In recent years, numerous high-quality datasets for fndus eye images (*FEI*) were published. We selected four publicly available datasets for our experiments: *FIVES* [10], consisting of 1,200 images with a resolution of 2048×2048 pixels; *DRHAGIS* [11], containing 80 images at 4752×3168 pixels; *HRF* [12], which includes 45 images at 3504×2336 pixels; and *STARE* [13], from which we used 20 images at 700×605 pixels resolution.

A. Training data preparation

In preparing data for *U-Net* training, it is essential to extract square windows (ideally with dimensions of 2^n) from the input images. These windows are then fed into the training pipeline to iteratively refine the model's performance. Although *U-Net* is known for its efficiency and accuracy under favorable conditions, a significant challenge lies in determining which parts of the image should be selected for training.To address this challenge, several key questions must be considered:

- Which locations should be chosen to ensure that windows cover both object and background regions effectively?
- How many windows should be sampled from each image?
- How many of these windows should predominantly contain the object of interest, and how many background?

Below, we present our approach to answering the forementioned questions by constructing a mechanism for automatic data generation. We use the *FIVES* dataset [10] as the primary input for training the *U-Net* model. The original dataset, consisting of 1,200 images, is divided into two subsets. The first subset, referred to as the *U-Net* Main Training Set (*UNETMTS*), includes 900 images and is used for model training. The second subset, called the *U-Net* Segmentation Validation Set (*UNETSVS*), comprises the remaining 300 images and is used to validate segmentation performance.

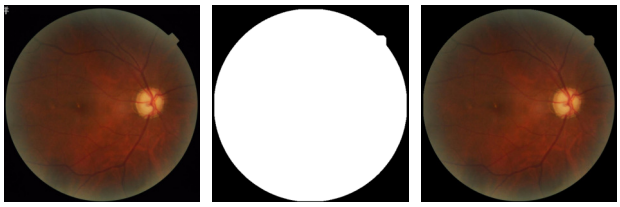
1) Mask finding: Each image is processed as follows:

1) Convert from RGB to grayscale.
2) Compute a threshold:

$$\text{threshold} = \frac{\text{mean}(\text{grayscale\_image})}{3} - 5$$

and binarize to generate a mask.
3) Apply flood fill from all four corners to produce an additional mask.
4) Combine the two masks by pixel-wise summation.
5) Apply erosion and dilation with a 20×20 kernel.

Finally, the mask is applied via logical AND operation. Pixels within its outer layer are blacked out; others remain unchanged (see Fig. 1).



(a) Input image (b) Mask (c) Masked image

Fig. 1. Image Masking Results

After the mask is generated, the image is trimmed from the left, right, top, and bottom edges to ensure that no completely black vertical or horizontal lines remain.

2) Random windows: The initial set, *UNETMTS*, is reduced to a specified value referred to as *TRDTSZ* (training data size), which is initially set to 900. However, this value is varied and analyzed in later sections of this article (see Section IV) to investigate whether the full set of 900 images is necessary, or if smaller subsets can still yield high segmentation accuracy. The goal is to investigate how much training data the *U-Net* model requires to achieve optimal results. Once the value of *TRDTSZ* is selected, *UNETMTS* is randomly divided into two subsets: a training set (*DPSTR*) and a test set (*DPSTS*), using a 75% to 25% split.

From each image, a set of training windows is randomly sampled. Two types of windows are defined: object-type windows (*OBTW*) and background-type windows (*BKTW*). A window is classified as *OBTW* if the number of pixels corresponding to the ground truth label exceeds a minimum threshold (e.g., 15%). Conversely, a window is labeled as *BKTW* if the proportion of object pixels does not exceed a maximum threshold (e.g., 5%). We also define target ratio of object to background windows per image (e.g., 50%), to ensure a representative sampling that reflects typical pixel distribution between object and background regions in *FEI*.

The underlying motivation for this windowing strategy is to balance object and background representation during training. This balance is reinforced by employing a weighted binary cross-entropy loss function in the *U-Net* implementation, which helps to address class imbalance. The size of each sampled window (*WS*) is typically set to either 64×64 or 128×128 pixels. Input images are rescaled after mask generation to 512×512 pixels. Larger values of *WS* and image resolution significantly increase GPU memory consumption - an important consideration given that our hardware setup was a GPU with only 8GB of memory.

The percentage values used in this step are based on intuition and can be adjusted depending on the type of images, particularly when the ratio of object pixels to background pixels varies. For instance, these values would differ when segmenting structures such as the eye cup (*EC*) or optic disc (*ED*) in other *FEI* datasets. The goal of this strategy is to capture a wide range of spatial arrangements of the target object within randomly selected windows, thereby enriching the training dataset for the *U-Net* model used in later stages of the system.

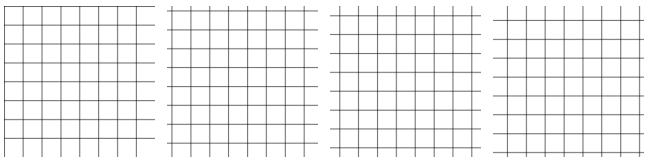
B. U-Net network training

The windows extracted in Section III-A are grouped into training batches, with a predefined number of epochs and a selected (*WS*) that matches the input dimensions of the *U-Net* model. The input windows, referred to as *X* windows, consist of RGB channels, with pixel values normalized to the range [0, 1]. The corresponding output windows *Y* windows, represent the segmentation masks, and are converted into binary image.

C. U-Net preliminary multi resolution segmentation

This step begins with a rough segmentation step (*RSS*). Input image is scaled to a resolution of 512×512, then segmented using a sliding window technique. Its primary goal is to produce preliminary segmentation results which are passed to the next stage Section III-D, where a dedicated *CNN* model is trained how to evaluate segmentation result.

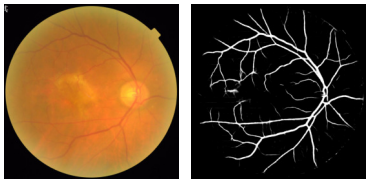
1) Windows mesh creation: Once the image has been prepared for segmentation - by generating a mask as described in Section III-A1 - it typically contains a black border surrounding the scene, with the retina centered. This preprocessed image is then analyzed across three color channels: red, green, and blue. Each color channel is divided into a collection of meshes (i.e., image patches) by sliding a window across the image. Mesh extraction begins with an offset of 0 and proceeds up to $WS - 1$, using a predefined sliding step denoted as *SS*. While it is not required for $WS \bmod SS = 0$, it is essential that $SS < WS$ to ensure overlapping patches and continuous coverage of the entire image. The idea of meshes is shown in Fig. 2



(a) Offset 0 (b) Offset 32 (c) Offset 64 (d) Offset 96

Fig. 2. Sample meshes generated for an image of size 1024×1024 pixels, *WS* of 128×128 pixels, and *WSS* set to 32

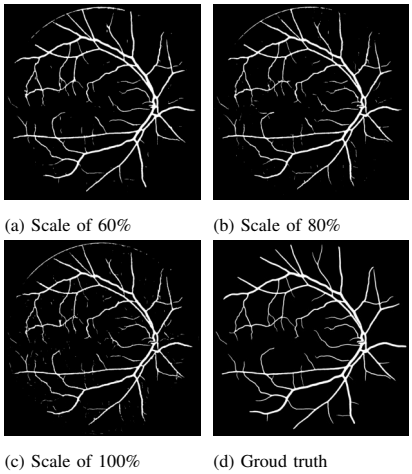
Each mesh allows the input image to be segmented using a different window configuration. This approach is crucial for capturing diverse spatial variations within the image, as each mesh is processed independently by the trained *U-Net* model. For each mesh, the *U-Net* generates a probability map, assigning to each pixel the likelihood of belonging to the object or background. Due to overlapping meshes, a single pixel may appear in multiple segmentations, resulting in multiple probability estimates. These values are averaged to generate a heatmap, which represents the overall likelihood of each pixel being part of the object. In this heatmap, brighter pixels correspond to higher confidence in object presence. Finally, the heatmap is binarized using a fixed threshold of 127, producing a black-and-white segmentation mask. An example of this process is shown in Fig. 2, where four meshes contribute to the heatmap visualized in Fig. 3. This method is referred to as the Sliding Window Algorithm (*SWA*).



(a) Input image (b) Pre binarization heatmap

Fig. 3. Input image from [10] along with generated heatmap

2) Data preparation for CNN training: As described in Section III-A, the main dataset [10] was partitioned to include a subset named *UNETSVS*, which is used for training the *CNN* in the subsequent evaluation stage. All images in this subset are processed through the (*RSS*). We perform segmentation at multiple scales, ranging from 20% to 100% of the original image size, in increments of 10%. This produces a series of black-and-white segmentation outputs for each image at varying resolutions. Figure 4 presents the segmentation results for three selected scales: 60%, 80%, and 100%. It is apparent from the visual comparison that the quality of the segmentations varies significantly across different scales.



(a) Scale of 60% (b) Scale of 80%

(c) Scale of 100% (d) Groud truth

Fig. 4. Different scales of input image for *CNN* segmentation assessment training

To objectively assess which of the segmentation results is most optimal, we design and train a simple *CNN*-based scoring network. This network consists of a single output neuron that produces a score between 0 and 1, where 0 indicates the

poorest segmentation quality and 1 indicates the best (see Fig. 7). The trained *CNN* is employed in the final segmentation stage, as described in Section III-E.

D. CNN training for segmentation results assessment

Using the *UNETSVS* set, we segment each image into nine black-and-white binarizations, producing a total of $300 \times (9 + 1) = 3000$ samples for training the *CNN*. The additional term in the formula accounts for the ground truth image (*GT*), which serves as the expected output from the segmentation algorithm. This ground truth image is also used to compute the Jaccard index [14], which measures the similarity between the segmented output and its corresponding *GT*. The index yields a value between 0 and 1, where 0 indicates a completely incorrect segmentation and 1 indicates a perfect match. The *CNN* architecture used for evaluation consists of a simple stack of convolutional layers, followed by batch normalization and pooling layers. Once the *CNN* scorer is trained, it can be used to evaluate and select the most accurate segmentation result. This final decision-making step is carried out as described in Section III-E.

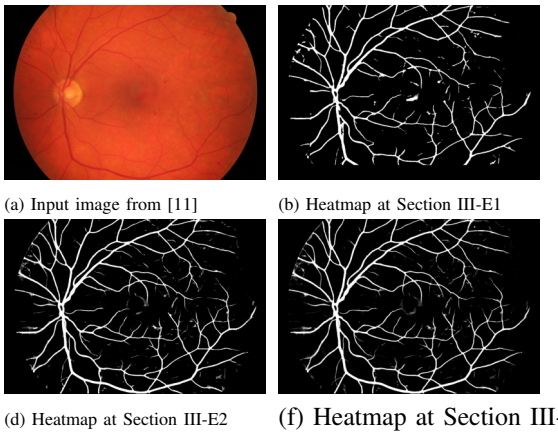
E. Optimal segmentation step

The final segmentation comprises of three stages.

1) Rough stage of final segmentation: The final segmentation algorithm builds upon the approach described in Section III-C. We continue to perform the Rough Segmentation Step (*RSS*), but in this phase, segmentation is carried out at scales ranging from 20% to 100%, with a step size of 20%. From the five resulting segmentation candidates, the one deemed most accurate by the trained *CNN* (introduced in Section III-D) is selected as the final output. The scale at which this optimal segmentation occurs is referred to as the *RSOR*. It is important to note that for smaller images (up to 800×800 pixels), we increase the maximum scale to 150%. This adjustment accounts for the fact that *U-Net* may underperform when operating on very low-resolution inputs.

2) Detailed stage of final segmentation: The *RSOR* ratio plays a crucial role, as it serves as the starting point for the subsequent stage, known as the Detailed Segmentation Step (*DSS*). This step further refines the search for the optimal segmentation scale. Specifically, segmentation is performed at scales ranging from $RSOR - 10\%$ to $RSOR + 10\%$, using a step size of 4%. As before, the trained *CNN* from Section III-D is used to evaluate the quality of each segmentation. The scale identified as producing the best result is referred to as the *DSOR*.

3) Most detailed stage of final segmentation: With the *DSOR* ratio now established, we proceed to the Most Detailed Segmentation Step (*MDSS*). In this final stage, the input image is rescaled according to the selected *DSOR* value. Unlike in Section III-C, where the window sliding step (*SS*) was set to 64, we now adopt a finer step size of 32 to enhance segmentation precision. The results of this final segmentation step, applied to a sample image from the *DRHAGIS* dataset [11], are shown in Figure 5. It is important to note that both the binarized segmentation outputs and the probability heatmaps, originally generated at the scaled resolution, are rescaled back to the original image dimensions for consistency and visualization.



(a) Input image from [11] (b) Heatmap at Section III-E1

(d) Heatmap at Section III-E2 (f) Heatmap at Section III-E3

Fig. 5. Input image from [11] along with generated heatmaps at 3 different stages with *SS* at 128, 64 and 32

All configuration parameters associated with window size (*WS*), sliding step (*SS*), and the scale percentages used for segmentation are fully adjustable by the user of the implementation provided in [3]. In Section IV, we present a series of experiments evaluating segmentation accuracy using the Jaccard Index across a dataset composed of three sources: *DRHAGIS* [11], *HRF* [12], and *STARE* [13]. This dataset is referred as the Combined Validation Dataset (*CVDS*).

IV. Experiments

In our experiments, we aimed to evaluate how well the proposed approach generalizes when a *U-Net* model trained on one dataset is applied to a completely different dataset.

We conducted a total of three experiments:

1) In the first experiment, both the *U-Net* and *CNN* models were trained on 100 randomly selected images from the *FIVES* dataset [10]. The trained models were then tested on the Combined Validation Dataset (*CVDS*), which consists of 150 images.
2) In the second experiment, 900 randomly selected images from the *FIVES* dataset were used to train both the *U-Net* and *CNN* models.
3) In the third experiment, we used the same *U-Net* model trained in the first experiment, but applied a naive Sliding Window Algorithm (*SWA*), as described in Section III-C.

The purpose of these experiments was to assess whether the full pipeline, outlined in Sections III-A through III-E, achieves superior segmentation performance compared to the traditional Sliding Window Algorithm (*SWA*). The central hypothesis is that identifying an optimal resolution, at which the *U-Net* model performs best for each individual image, leads to more accurate segmentation results then applying the model directly, at a fixed resolution in a brute-force manner. The *U-Net*

architecture used for segmentation is shown in Figure 6. As illustrated, the structure is simplified compared to the original version proposed in [2]. The accompanying *CNN* (Figure 7) was designed to be lightweight.

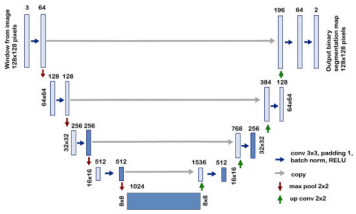


Fig. 6. *U-Net* used across all segmentation stages

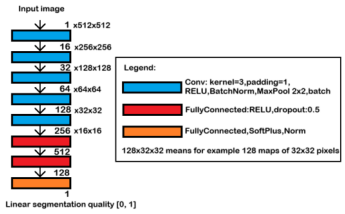


Fig. 7. *CNN* used for segmentation quality assessment

A. First experiment

In the first experiment we randomly selected 100 images from *FIVES* dataset. Results are shown in Table I.

TABLE I
EXPERIMENT 1 WITH JACCARD METRICS AND *TTE* FOR VARIOUS PARAMETER SETTINGS OF *WSS*.

RWSS	DWSS	OWSS	TTE	AJI	MNJI	MXJI
128	128	16	3316	0.6114	0.2722	0.7705
128	16	8	53107	0.6123	0.2823	0.7672
128	16	16	5641	0.6145	0.274	0.7650
128	32	8	10527	0.6148	0.2713	0.7672
128	64	8	3822	0.6147	0.2722	0.7705
128	**64**	**32**	**2704**	**0.6125**	**0.2714**	**0.7694**
128	**64**	**8**	**12877**	**0.6153**	**0.2713**	**0.7713**
16	16	8	35212	0.6078	0.2823	0.7570
16	32	8	18842	0.6089	0.2823	0.7570
16	64	8	7653	0.6097	0.2823	0.7516
32	32	8	14048	0.6082	0.2823	0.7436
32	64	8	11784	0.6099	0.2823	0.7516
64	16	8	16839	0.6137	0.2823	0.7649
64	16	16	5627	0.6147	0.2829	0.7634
64	32	8	9680	0.6151	0.2819	0.7665
64	32	16	4091	0.6153	0.2819	0.7668
64	64	8	7938	0.6143	0.2823	0.7601

The common parameters selected for the segmentation process are as follows:

- Minimum rough segmentation scale: 20%,
- Maximum rough segmentation scale: 100%,
- Rough segmentation scale step: 20%,
- Rough segmentation range for optimal resolution: $\pm$10%,
- Detailed segmentation range for optimal resolution: $\pm$4%.

The meanings of the column headings used in the evaluation tables are:

- *RWSS*: Rough Segmentation Window Step,
- *DWSS*: Detailed Segmentation Window Step,
- *FOWSS*: Final Optimal Window Step for Segmentation,
- *TTE*: Total Time Elapsed (in seconds),
- *AJI*: Average Jaccard Index,
- *MNJI*: Minimum Jaccard Index,
- *MXJI*: Maximum Jaccard Index.

The highest achieved *AJI* was 0.6153; however, this result came at the cost of a total time elapsed (*TTE*) of 12,877 seconds. In contrast, the fastest configuration, with a *TTE* of just 2,704 seconds, was obtained using the parameters *RWSS* = 128 pixels, *DWSS* = 64 pixels, and *FOWSS* = 32 pixels. This setup yielded an *AJI* of 0.6125, which is only marginally lower than the best result. As such, it represents a significantly more efficient trade-off between segmentation accuracy and computational performance.

B. Second experiment

In the second experiment we randomly selected 900 images from *FIVES* dataset. Exactly the same experiments in terms of values for **RWSS**, **DWSS** and **OWSS** were conducted and can be previewed in details in [3].

The best achieved *AJI* in this experiment was 0.5957, obtained with a total time elapsed (*TTE*) of 8,320 seconds. The fastest configuration, with a *TTE* of just 6,208 seconds, was achieved using the parameters *RWSS* = 128 pixels, *DWSS* = 64 pixels, and *FOWSS* = 32 pixels. However, this setup yielded an *AJI* of only 0.5813, which is noticeably lower than the optimal value. This indicates a trade-off between speed and accuracy, with the faster configuration sacrificing some segmentation performance.

C. Third experiment

In the third experiment we decided to try and apply only a part of the Section III-E3. We specified the *SS* to values of 8, 16, 32, 64 and 128. You can see the segmentation results in Table II. The results in this experiment are notably inferior to those presented in Section IV-A. The best outcome observed corresponds to *SS* of 8, resulting in *TTE* of 24,227 seconds and *AJI* of 0.5639. While this approach demonstrates high execution speed for *SS* values ranging between 32 and 128 pixels, the resulting *AJI* scores remain unsatisfactory. In contrast, the fastest configuration from Section IV-A achieved an *AJI* of 0.6125, clearly demonstrating the superiority of the hybrid approach described in this article over the traditional *SWA*-based method.

V. CONCLUSIONS

The experiments conducted in this article reveal several notable observations regarding the proposed *HUS* algorithm:

- When using only 100 training images from the *FIVES* dataset [10], we achieved a higher accuracy of 0.6153 compared to 0.5957 obtained with 900 training images. This suggests that in Section IV-B, the *U-Net* and *CNN*

TABLE II
EXPERIMENT 3 WITH JACCARD METRICS AND *TTE* FOR VARIOUS PARAMETER SETTINGS OF *WSS*.

WSS	TTE	AJI	MNJI	MXJI
128	454	0.4878	0.1824	0.6490
64	591	0.5226	0.1920	0.6918
32	1354	0.5591	0.2582	0.7131
16	4504	0.5626	0.2626	0.7153
8	24727	0.5639	0.2642	0.71705

models may have become overly specialized to the training data, resulting in decreased generalization to the test set.

- In Experiments IV-A and IV-B, we demonstrated that the hybrid method outperforms the classic *SWA* approach by nearly 0.05 in *AJI*, while maintaining comparable execution speeds.
- Our experiments with *HUS* also indicate that high values of the sliding step (*SS*) are not necessarily critical for achieving optimal performance. The results suggest a possible correlation between hierarchical window sizes. Specifically, for a window size of 2^n, typically associated with the first convolutional layer of *U-Net*, the following configuration yields strong results: $RWSS = 2^n$, $DWSS = 2^{n-1}$, and $FOWSS = 2^{n-2}$.

The *HUS* algorithm [3] effectively addresses limitations associated with traditional *U-Net* segmentation:

- The requirement for training and test datasets to have similar resolutions.
- The substantial memory consumption of *U-Net* when processing high-resolution images.
- The sensitivity of classical *U-Net* to variations in the spatial location of the target object.

VI. DECLARATION OF NO COMPETING INTEREST

The author declares that he has no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

REFERENCES

[1] Krizhevsky, Alex and Sutskever, Ilya and Hinton, Geoffrey E.,*ImageNet Classification with Deep Convolutional Neural Networks*,Advances in Neural Information Processing Systems (NIPS), volume 25, pages 1097–1105, 2012,doi:10.1145/3065386,https://proceedings.neurips.cc/paper/2012/file/c399862d3b9d6b76c8436e924a68c45b-Paper.pdf

[2] Ronneberger, Olaf and Fischer, Philipp and Brox, Thomas,*U-Net: Convolutional Networks for Biomedical Image Segmentation*,Medical Image Computing and Computer-Assisted Intervention (MICCAI),pages 234–241, 2015, Springer,doi:10.1007/978-3-319-24574-4_28,https://arxiv.org/abs/1505.04597

[3] Lesław M. Pawlaczyk, *U-Net segmentation implementation*,2025, https://github.com/palles77/UnetSegmentation,Accessed: January 29, 2025

[4] Ren, Kan and Chang, Longdan and Wan, Minjie and Gu, Guohua and Chen, Qian,*An improved U-net based retinal vessel image segmentation method*,Heliyon, volume 8, number 10, pages e11187, 2022, Elsevier,doi:10.1016/j.heliyon.2022.e11187,https://www.sciencedirect.com/science/article/pii/S2405844022024756

[5] Liu, Rui and Pu, Wen and Nan, Hong and Zou, Yao,*Retina image segmentation using the three-path Unet model*,Scientific Reports, volume 13, number 1, pages 22579, 2023, Nature Publishing Group,doi:10.1038/s41598-023-50141-0,https://www.nature.com/articles/s41598-023-50141-0

[6] Das, Smita and Chakraborty, Suvadip and Mishra, Madhusudhan and Majumder, Swanirbhar,*Assessment of retinal blood vessel segmentation using U-Net model: A deep learning approach*,Franklin Open, volume 8, pages 100143, 2024, Elsevier,doi:10.1016/j.fraope.2024.100143,https://www.sciencedirect.com/science/article/pii/S2773186324000732

[7] Yun, Jiang and Wang, Falin and Gao, Jing and Cao, Simin,*Multi-Path Recurrent U-Net Segmentation of Retinal Fundus Image*,Applied Sciences, volume 10, number 11, pages 3777, 2020, MDPI,doi:10.3390/app10113777,https://www.researchgate.net/publication/341810610_Multi-Path_Recurrent_U-Net_Segmentation_of_Retinal_Fundus_Image

[8] Huang, Ko-Wei and Yang, Yao-Ren and Huang, Zih-Hao and Liu, Yi-Yang and Lee, Shih-Hsiung,*Retinal Vascular Image Segmentation Using Improved UNet Based on Residual Module*,Bioengineering, volume 10, number 6, pages 722, 2023, MDPI,doi:10.3390/bioengineering10060722,https://www.mdpi.com/2306-5354/10/6/722

[9] Westwańska, W. and Respondek, J., *Counting instances of objects in color images using U-Net network on example of honey bees*, in Proceedings of the 2019 Federated Conference on Computer Science and Information Systems (FedCSIS), Annals of Computer Science and Information Systems, vol. 18, pp. 87–90, Leipzig, Germany, September 2019, doi:10.15439/2019F94, https://ieeexplore.ieee.org/document/8859742

[10] Kai Jin, Xingru Huang, Jingxing Zhou, Yunxiang Li, Yan Yan, Yibao Sun, Qianni Zhang, Yaqi Wang, Juan Ye, *FIVES: A Fundus Image Dataset for Artificial Intelligence based Vessel Segmentation*, 2022 Data Descriptor in Scientific Data, volume 9, number 475, pages 475, 2022, Nature Portfolio,doi:10.1038/s41597-022-01590-2,https://www.nature.com/articles/s41597-022-01564-3,https://figshare.com/ndownloader/files/34969398

[11] Holm, Sven, Russell, Greg, Nourrit, Vincent, McLoughlin, Niall, *DR HAGIS - a fundus image database for the automatic extraction of retinal surface vessels from diabetic patients*, 2017 *Journal of Medical Imaging (Bellingham)*, volume 4, number 1, pages 014503, 2017, SPIE,doi:10.1117/1.JMI.4.1.014503,https://pmc.ncbi.nlm.nih.gov/articles/PMC5299858/pdf/JMI-004-014503.pdf,http://personalpages.manchester.ac.uk/staff/niall.p.mcloughlin/DRHAGIS.zip

[12] Budai, Attila, Bock, Rüdiger, Maier, Andreas, Hornegger, Joachim, Michelson, Georg,*Robust Vessel Segmentation in Fundus Images*,2013 *International Journal of Biomedical Imaging*,volume 2013, article 154860, 2013, Hindawi,doi:10.1155/2013/154860,http://www5.informatik.uni-erlangen.de/Forschung/Publikationen/2013/Budai13-RVS.pdf,https://www5.cs.fau.de/fileadmin/research/datasets/fundus-images/all.zip

[13] Hoover, A. D., Kouznetsova, V., Goldbaum, M.,*Locating blood vessels in retinal images by piecewise threshold probing of a matched filter response*,2000 *IEEE Transactions on Medical Imaging*,volume 19, number 3, pages 203–210, 2000, IEEE,doi:10.1109/42.845178,https://cecas.clemson.edu/~ahoover/stare/probing/index.html

[14] Jaccard, P., *Étude comparative de la distribution florale dans une portion des Alpes et des Jura*, 1901, *Bulletin de la Société Vaudoise des Sciences Naturelles*, volume 37, pages 547–579, Société Vaudoise des Sciences Naturelles.