Mamba-Based Tumor Segmentation

Clay Crews†
Department of Computer Science
University of South Carolina
Columbia, SC, United States
jccrews@email.sc.edu

Lexington Whalen†
Department of Computer Science
University of South Carolina
Columbia, SC, United States
LAWHALEN@email.sc.edu

***Abstract*—U-net and transformer-based architectures have dominated medical image segmentation, particularly for tumor segmentation from brain MRI scans. However, recent advances in Structured State Space Sequence Models (SSMs) present a promising alternative with comparable modeling power to transformers but linear scalability in sequence length, making them well-suited for large medical images.**

We propose exploring the application of the Mamba block, a variant of SSMs, for tumor segmentation in brain MRI scans. By leveraging selective state spaces, Mamba blocks can potentially enhance the efficiency and effectiveness of tumor segmentation while maintaining linear scalability.

Our methodology involves implementing and fine-tuning a Mamba-based architecture on a dataset of annotated brain MRI scans. We will evaluate the model's performance using standard segmentation metrics and compare the results with U-net and transformer-based approaches. We open-source the code here: https://github.com/lxaw/mamba-tumor-seg.

***Index Terms*—Mamba, Segmentation, Selective State Spaces**

I. Introduction

MRI imaging to identify brain tumors has naturally been a target for accurate image segmentation through the use of machine learning (ML) architecture. The use of computer-aided diagnosis of tumors can be crucial to quickly identify tumors and determine a course of action for the patient. Gliomas are the most common primary brain malignancy with varying degrees of aggressiveness in the brain. The protocol for MRI image annotation of these tumors consists of the following labels: the whole tumor extent, the tumor core, the non-enhancing/necrotic tumor region, and regions of low grade gliomas [1]. These annotations of an MRI are used to determine the size and severity of a tumor.

To accurately segment these images, picking out the small features of the image becomes the focus. The use of the U-Net architecture, presented by Ronneberger, Fischer, and Brox [2], has had success in MRI image segmentation. U-Net builds on the approach of fully convolutional networks by improving the limitation on the large amount of data needed to accurately train the model. Very few training images are needed due to the contracting and expansive paths in this architecture. In the contracting path, images are downsampled and pooled along with increasing the number of feature channels. Each downsampled resolution produces a multi-channel feature map. The expansive path up samples and combines with each segmentation feature map from the contracting layer to localize features in the image. A 1x1 convolution layer is applied to the final image to map the large feature vector to a selected number of classes. All of this essentially creates a feature map where each pixel's relevance is taken into account and evaluated in the end result. An approach using the U-Net architecture would provide the detailed analysis needed for tumor annotation.

An arising issue in the complex recurrent or convolutional neural networks used for image segmentation is the length of the sequences given as input. Maintaining relevance of the current area of an image in relation to the rest of the image is key for more accurate segmentation. An approach with the Transformer architecture has a large emphasis on learning specific features in an image, outlined by Vaswani et al. 2017 [3]. This architecture follows an encoder-decoder pattern, connected through an attention mechanism, a transformer block. Transformer blocks show capability of learning long-distance dependency throughout an input image. In the self-attention mechanism for this block, a single element in a given sequence is compared to all of the elements in the sequence. However, this mechanism becomes computationally very expensive when it comes to long sequences.

Selective State Spaces (SSMs) pose an improvement in computational efficiency over Transformers for this long dependency range. SSMs represent the relevance and context of different parts of the input sequence giving selective attention. This model is widely used in control theory for time variant systems and in fields such as computational neuroscience. Selective attention is very efficient in handling long sequences over time and presents a more localized context of the data. Sections of sequences that deserve attention are represented by the model and will dynamically update these values to reflect the contextual relevance.

Making use of Structured State Spaces, presented by Gu, Goel, and Ré [4], for their superiority in long range dependency tasks, a specific type of SSM, the Mamba architecture, by Gu and Dao [5], proposed a foundational model to operate on arbitrary sequences from a variety of inputs in the domain of sequence modeling. This model achieves the power of Transformers while scaling linearly in sequencing length to be computationally efficient. Additionally, the selection mechanism of Mamba improves on an SSMs ability to focus on or ignore sections in an input sequence by parameterizing the

† Equal contributions

parameters of an SSM to reflect the current context of the input. The Mamba model and its application to brain tumor MRI imaging segmentation will be evaluated in this project.

II. Proposed Methodology

In this paper, we strive to maintain high accuracy in our image segmentation while also keeping low parameter counts. We primarily seek to use pyramidal pooling modules [6] to improve accuracy, and use a variant of the Mamba [5] architecture inspired by [7] to maintain small parameter sizes.

III. Input Data

We shall be using a Brain MRI segmentation dataset found in [8]. This dataset has been used in several papers regarding classification of the shape and severity of tumors, and provides an adequate dataset for research purposes in the field of medical image analysis, particularly in the area of brain MRI segmentation. Researchers have utilized this dataset in various studies focusing on the classification of tumor shapes and the assessment of tumor severity. With its comprehensive collection of brain MRI scans, along with corresponding segmentation masks, the dataset offers valuable resources for developing and evaluating algorithms aimed at automating tumor detection and analysis.

The dataset contains brain MRI images along with their segmentation masks for 110 patients. There were a total of 3143 train images, 393 validation images, and 393 test images. The distribution of the tumor and non-tumor images is shown in Figure 1.

IV. Output Data

The output of this model shall be segmentation masks for new, never before seen images.

V. Mamba

Mamba [5] is a recently proposed architecture that combines the modeling power of Transformers with the linear scaling efficiency of state space models (SSMs). The key idea is to augment SSMs with a selection mechanism that allows the model parameters to vary based on the input, enabling it to perform content-aware reasoning. Standard SSMs map an input sequence $x(t)$ to an output $y(t)$ via a latent state $h(t)$ using a linear time-invariant system:

$$h'(t) = \mathbf{A}h(t) + \mathbf{B}x(t)$$
$$y(t) = \mathbf{C}h(t) + \mathbf{D}x(t)$$

where the parameter matrices $\mathbf{A}$, $\mathbf{B}$, $\mathbf{C}$, $\mathbf{D}$ are fixed. In the original paper, the $\mathbf{D}$ matrix is considered only as a skip connection, and thus as it does not directly play a role in the differential equation, was ignored. This allows SSMs to be computed efficiently as a convolution. However, the time-invariance means the model cannot change its behavior for specific inputs, making it difficult to solve tasks requiring selective focus, such as selective copying or induction heads [5]. Mamba introduces a selection mechanism where the SSM parameters $\mathbf{B}$, $\mathbf{C}$, Δ (the state step size) vary for each input token x_t. This allows the model to selectively propagate or forget information based on the current token. The selective SSM is computed using a parallel scan operation to maintain linear complexity. The Mamba block interleaves the selective SSM with linear projections, convolutions, and activations:

1) Linear projection to expand the embedding dimension
2) Convolution to mix information between dimensions
3) Selective SSM via parallel scan to efficiently propagate information
4) Linear projection to get back to the target embedding size

These blocks are stacked to form the Mamba architecture. Compared to Transformers, Mamba can achieve comparable modeling performance while scaling linearly in sequence length rather than quadratically [5]. Overall, the selective SSM

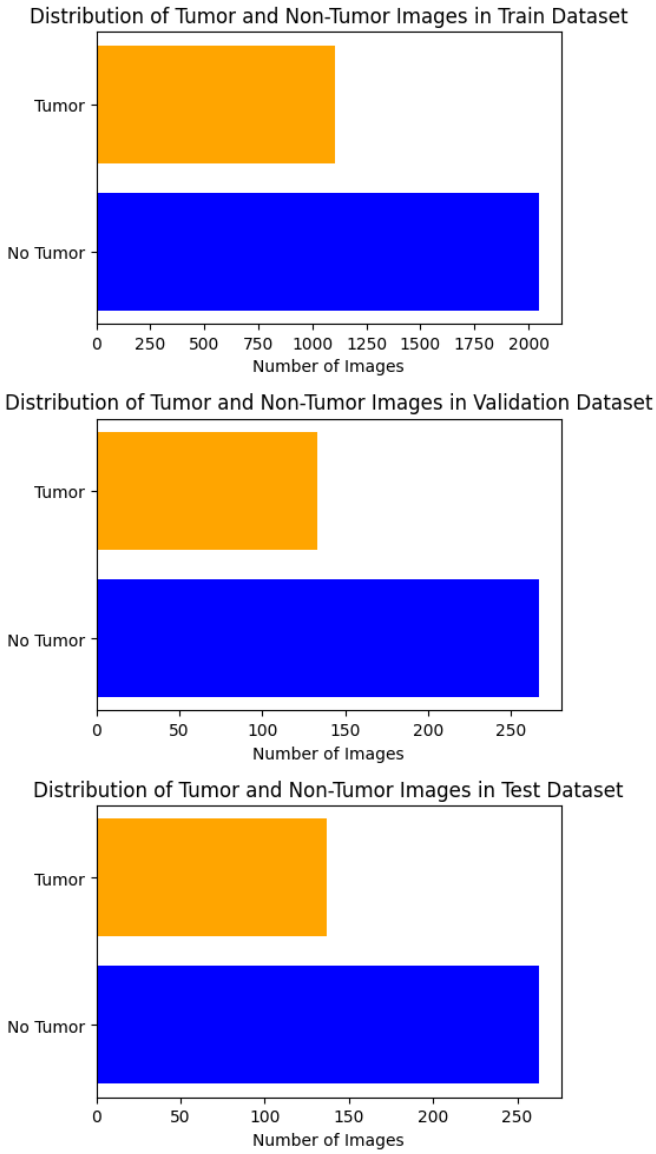

Fig. 1: Distributions of tumor and non-tumor images in the train, validation, and test datasets.

leverages the strengths of RNNs, CNNs and Transformers - it maintains an unbounded context window like RNNs, can be parallelized like CNNs, and achieves content-aware reasoning like Transformers, while being more efficient than all three. This makes Mamba a promising foundation model for processing long sequences across various domains.

VI. Compared Models

Prior to an analysis of how the Pyramidal U-Mamba compares, we shall explain what models we chose for comparison and why.

For this our analysis, we compare four models against our own developed ones. The models are U-Net [2], ResNet18 [9], ResNet50, U-Mamba [10], and SegViT [11]. Below we explain our choices.

A. U-Net

The core idea behind U-Net is to complement a traditional contracting network with successive layers that replace pooling operations with upsampling operators. These upsampling layers aim to increase the resolution of the output. Subsequent convolutional layers can then learn to assemble a precise output based on this high-resolution information [2]. A key modification in the U-Net architecture is the inclusion of a large number of feature channels in the upsampling part. This allows the network to effectively propagate contextual information to higher resolution layers. As a result, the expansive path of the network becomes more or less symmetric to the contracting part, leading to a distinctive U-shaped architecture. To predict pixels at the border regions of the image, the missing context is extrapolated by mirroring the input image. This tiling strategy is crucial for applying the network to large images, as it circumvents resolution limitations imposed by GPU memory constraints [2]. Due to its encoder-decoder architecture, skip connections, multi-scale feature extraction, and its efficiency, U-Net has been used in many denoising and diffusion models. For instance, DDPMs (Denoising Diffusion Probabilistic Models) use an architecture similar to U-Net for denoising and sample generation [12] while the popular Stable Diffusion architecture uses a U-Net based architecture for the diffusion process [**?**]. We have chosen to compare our novel segmentation architecture against the U-Net architecture due to its well-established reputation and widespread adoption in the field of medical image segmentation.

B. ResNet

We have also selected ResNet18 and ResNet50 as additional benchmarks. These architectures, introduced by He et al. in their seminal work "Deep Residual Learning for Image Recognition" [9], have revolutionized the field of deep learning by addressing the problem of vanishing gradients in deep neural networks. The key innovation in ResNets is the introduction of residual connections, which allow the network to learn residual functions with reference to the input layer, thereby facilitating the training of much deeper networks. ResNet18 and ResNet50, with 18 and 50 layers respectively, have been widely adopted in various computer vision tasks, including image classification, object detection, and segmentation. These models have demonstrated exceptional performance and generalization ability across diverse datasets. By comparing our proposed architecture against ResNet18 and ResNet50, we aim to assess its effectiveness in relation to these well-established and highly influential architectures. This comparison will provide valuable insights into the capabilities of our model and its potential to advance the state-of-the-art in image segmentation tasks.

C. U-Mamba

As our model takes much inspiration from the recently developed U-Mamba design [10], we also choose to incorporate it in our comparison. U-Mamba addresses these limitations by introducing a novel hybrid CNN-SSM block that leverages the strengths of both architectures. The convolutional layers in the block are responsible for local feature extraction, while the State Space Sequence Models (SSMs) [4], a new family of deep sequence models, are known for their strong capability in handling long sequences and capturing long-range dependencies. By integrating these two components, U-Mamba achieves a balance between local and global information processing, enabling it to effectively handle long-range dependencies in biomedical image segmentation tasks. Moreover, U-Mamba incorporates a self-configuring mechanism that allows it to automatically adapt to various datasets without manual intervention, enhancing its versatility and usability.

D. SegFormer

SegFormer [13] is a transformer-based semantic segmentation model that employs a hierarchical structure and a novel attention mechanism called Efficient Self-Attention (ESA). The SegFormer architecture consists of a transformer encoder for capturing long-range dependencies and a lightweight All-MLP decoder for generating high-resolution segmentation masks. The ESA module reduces the computational complexity of self-attention by performing attention operations in a local window and aggregating global information through a depth-wise convolution. This allows SegFormer to efficiently process high-resolution images while capturing both local and global context. Moreover, SegFormer introduces a position-sensitive embedding scheme that encodes both spatial and channel-wise information, enhancing the model's ability to capture fine-grained details. By comparing U-Mamba against SegFormer, we aim to evaluate the effectiveness of our hybrid CNN-SSM approach in relation to the transformer-based segmentation mechanism employed by SegFormer. This comparison will provide insights into the strengths and weaknesses of both architectures and their ability to handle complex biomedical image segmentation tasks. Furthermore, it will help us understand the potential of transformer-based approaches and hybrid approaches in advancing the state-of-the-art in biomedical image segmentation.

E. UltraLight VM-UNet

The UltraLight VM-UNet [7] is a lightweight neural network architecture designed for skin lesion segmentation tasks. It is built upon the Vision Mamba module, which is a state-space model (SSM) that can efficiently handle long-range dependencies in sequences, making it well-suited for image segmentation tasks.

The key innovation of the UltraLight VM-UNet is the proposed Parallel Vision Mamba Layer (PVM Layer), which processes deep features in parallel using multiple Vision Mamba blocks. Specifically, the input feature map is split into multiple sub-feature maps, each processed by a separate Vision Mamba block with a reduced channel count. This parallel processing approach allows the UltraLight VM-UNet to maintain high segmentation performance while significantly reducing the number of parameters and computational complexity.

The UltraLight VM-UNet is reported to have only 0.049 million parameters and a computational cost of 0.060 GFLOPs, which is significantly lower than traditional convolutional neural networks and transformers used for image segmentation tasks. Despite its lightweight nature, the authors demonstrate that the UltraLight VM-UNet achieves competitive performance on three publicly available skin lesion segmentation datasets, outperforming several state-of-the-art lightweight models.

The success of the UltraLight VM-UNet can be attributed to the authors' in-depth analysis of the key factors influencing the parameters of the Vision Mamba module. By identifying the number of input channels as a critical factor affecting the parameter count, they were able to design the PVM Layer to process features in parallel while keeping the overall channel count constant, leading to a significant reduction in parameters without compromising performance.

VII. Our Models

Inspired by [10] and [7], we sought to implement both models on a different problem: tumor segmentation.

Our goal was to 1) maintain dice scores comparable to state-of-the-art (SOTA) methods such as those listed above, while 2) being smaller than those above.

We now go into what we did for each of our models.

A. UMambaBot-PP

This model incorporates pyramidal pooling, which is a strategy used in convolutional neural networks (CNNs) to capture context and incorporate multi-scale information [6]. Pyramidal pooling involves parallel pooling operations at different scales, followed by concatenation of the resulting feature maps. This approach has been shown to improve the performance of CNNs in various computer vision tasks, including segmentation, by enabling the model to capture both local and global context.

In our UMambaBot-PP model, we integrated pyramidal pooling into the U-Mamba architecture, aiming to leverage the strengths of both the Mamba module and multi-scale feature extraction.

B. UL-VM-UNet-v1

For UL-VM-UNet-v1, the initial channel list was modified from [8, 16, 24, 32, 48, 64] to [16, 32, 64, 128, 256], increasing the number of channels at each step of the encoder and decoder. Additionally, the depth of the U-Net was reduced by one layer to keep the parameter count low while increasing the number of parameters in each step. The intention was to improve performance by increasing the capacity of the model while maintaining a reasonable parameter count.

C. UL-VM-UNet-v2

In UL-VM-UNet-v2, the number of parallel branches in the Parallel Vision Mamba (PVM) block was increased from 4 to 8. This modification directly decreases the parameter count, further proving the idea proposed in the UltraLight VM-UNet paper [7] that processing features in parallel with reduced channel counts can significantly reduce parameters while maintaining performance.

D. UL-VM-UNet-v3

UL-VM-UNet-v3 combines the modifications from UL-VM-UNet-v1 and UL-VM-UNet-v2. It increases the number of parallel branches in the PVM block to 8 and modifies the channel list to [16, 32, 64, 128, 256]. Additionally, the depth of the U-Net was reduced to 5 layers. This approach aims to balance the trade-off between model capacity and parameter efficiency.

E. UL-VM-UNet-v4

This model incorporates pyramidal pooling, similar to UMambaBot-PP, but within every PVM block in both the encoder and decoder in the UltraLight VM-UNet architecture. The goal was to leverage the benefits of multi-scale feature extraction while maintaining the parameter efficiency of the UltraLight VM-UNet.

F. UL-VM-UNet-v5

UL-VM-UNet-v5 is meant to be a comparison to the -v4 and -v6 UltraLight variations. This model trained and performed well with 1.1M parameters. Incorporating pyramidal pooling on the UL network scored slightly worse than without pyramidal pooling but required approximately 400,000 more parameters.

G. UL-VM-UNet-v6

Another variation to further test the effects of incorporating pyramidal pooling in the UltraLight VM-UNet architecture was made by only using pyramidal pooling in the bottleneck layer, similarly done in UMambaBot-PP. The main goal here, again, was to integrate pyramidal pooling for the benefit of multi-scale feature extraction while keeping a low parameter count.

VIII. Results

We now compare our models against ResNet18, ResNet50, standard U-Mamba bottleneck, UNet, and UltraLight VM-UNet. We train for 100 epochs on all models, and use dice-loss with Adam optimizer. We train on roughly 3000 images, and validate on roughly 400. we then test on roughly 400. We show the distributions of our data in Figure 1. Model results are shown in Table I and Figures 2,3, with additional segmentation results in Appendix B.

Model Name	#Params (M) ↓	Dice Score ↑
UMambaBot-PP(Ours)	10.0	0.8867
UMambaBot	**9.8**	0.8799
ResNet18	13.9	0.8648
ResNet50	42.9	0.8672
SegFormer	17.8	0.8454
UNet	7.8	**0.8929**
UL-VM-UNet	0.049	0.79955
UL-VM-UNet-v1(Ours)	0.20	0.8409
UL-VM-UNet-v2(Ours)	**0.042**	0.8243
UL-VM-UNet-v3(Ours)	0.176	0.8109
UL-VM-UNet-v4(Ours)	1.1	0.8548
UL-VM-UNet-v5(Ours)	0.7	**0.8633**
UL-VM-UNet-v6(Ours)	0.77	0.8628

TABLE I: Comparison of different segmentation models.

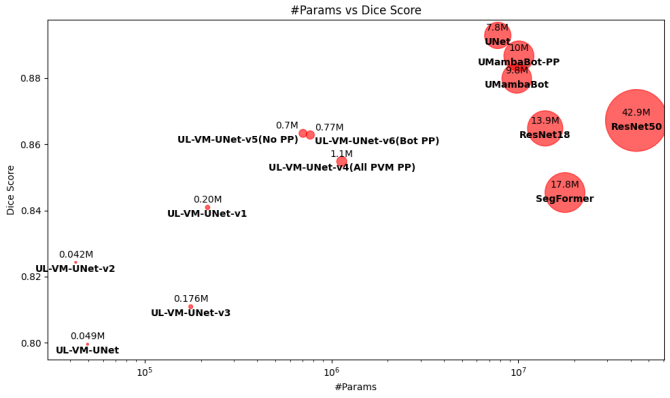

Fig. 2: A scatter plot of Dice Score vs model parameter count, where "M" means "millions. The points are scaled to help represent size.

The integrated pyramidal pooling module to UMambaBot resulted in a slight increase in performance over the base model. However, UMambaBot-PP had an increase in parameter count by approximately 200,000 parameters. Additionally, our UL-VM-UNet-v2 achieved the lowest parameter count of 0.042 M with an increase over the original UltraLight VM-UNet architecture. Integrating a pyramidal pooling module to the UltraLight model, as done in UL-VM-UNet-v4 and -v6, resulted in an increase in parameter count with no change or a decrease in performance.

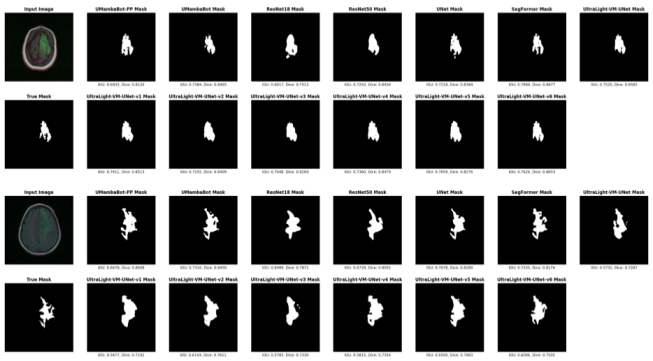

Fig. 3: Sample of segmentation results from each model and their dice score for the segmentation. (Additional samples in Appendix B)

IX. Conclusion

In this work, we explored the application of Structured State Space Sequence Models (SSMs), particularly the Mamba architecture, for tumor segmentation from brain MRI scans. The Mamba architecture leverages selective state spaces, enabling efficient and effective segmentation while maintaining linear scalability.

Our experiments and comparisons with state-of-the-art (SOTA) models demonstrate the potential of the Mamba-based approach in both model size and segmentation accuracy. The smallest model, ULVM-Net-v2, achieved a compelling dice score of 0.8243 with only 0.042M parameters, achieving comparable performance to the larger ResNet50 model (42.9M params) with a dice score of 0.8672, while also outperforming the original UltraLight VM-UNet in terms of both parameter count and dice score.

While the traditional U-Net model achieved the highest dice score of 0.8929, our UMambaBot-PP model closely followed with a dice score of 0.8867. Notably, our Mamba-based UMambaBot-PP and UMambaBot models outperformed the Segformer transformer model, indicating the potential of Mamba-based approaches to surpass transformer-based models for this task.

We were able to make the UltraLight model smaller and more accurate on our task, further proving the potential of Mamba-based systems that prioritize compactness and efficiency. The selective state spaces and parallel processing strategies employed in our models enabled us to strike a balance between model size and segmentation accuracy.

Overall, this work contributes to the exploration of Structured State Space Sequence Models for medical image segmentation tasks. The Mamba architecture's ability to capture long-range dependencies while maintaining linear scalability positions it as a promising alternative to traditional convolutional and transformer-based approaches, especially in resource-constrained environments.

Future work could involve further refinements and optimizations of the Mamba-based models, as well as their application

to other medical imaging modalities and segmentation tasks. Additionally, investigating the interpretability and robustness of these models could provide valuable insights for their deployment in real-world clinical settings.

REFERENCES

[1] S. Bakas, M. Reyes, A. Jakab, S. Bauer, M. Rempfler, A. Crimi, R. T. Shinohara, C. Berger, S. M. Ha, M. Rozycki, M. Prastawa, E. Alberts, J. Lipkova, J. Freymann, J. Kirby, M. Bilello, H. Fathallah-Shaykh, R. Wiest, J. Kirschke, B. Wiestler, R. Colen, A. Kotrotsou, P. Lamontagne, D. Marcus, M. Milchenko, A. Nazeri, M.-A. Weber, A. Mahajan, U. Baid, E. Gerstner, D. Kwon, G. Acharya, M. Agarwal, M. Alam, A. Albiol, A. Albiol, F. J. Albiol, V. Alex, N. Allinson, P. H. A. Amorim, A. Amrutkar, G. Anand, S. Andermatt, T. Arbel, P. Arbelaez, A. Avery, M. Azmat, P. B., W. Bai, S. Banerjee, B. Barth, T. Batchelder, K. Batmanghelich, E. Battistella, A. Beers, M. Belyaev, M. Bendszus, E. Benson, J. Bernal, H. N. Bharath, G. Biros, S. Bisdas, J. Brown, M. Cabezas, S. Cao, J. M. Cardoso, E. N. Carver, A. Casamitjana, L. S. Castillo, M. Catà, P. Cattin, A. Cerigues, V. S. Chagas, S. Chandra, Y.-J. Chang, S. Chang, K. Chang, J. Chazalon, S. Chen, W. Chen, J. W. Chen, Z. Chen, K. Cheng, A. R. Choudhury, R. Chylla, A. Clérigues, S. Colleman, R. G. R. Colmeiro, M. Combalia, A. Costa, X. Cui, Z. Dai, L. Dai, L. A. Daza, E. Deutsch, C. Ding, C. Dong, S. Dong, W. Dudzik, Z. Eaton-Rosen, G. Egan, G. Escudero, T. Estienne, R. Everson, J. Fabrizio, Y. Fan, L. Fang, X. Feng, E. Ferrante, L. Fidon, M. Fischer, A. P. French, N. Fridman, H. Fu, D. Fuentes, Y. Gao, E. Gates, D. Gering, A. Gholami, W. Gierke, B. Glocker, M. Gong, S. González-Villá, T. Grosges, Y. Guan, S. Guo, S. Gupta, W.-S. Han, I. S. Han, K. Harmuth, H. He, A. Hernández-Sabaté, E. Herrmann, N. Himthani, W. Hsu, C. Hsu, X. Hu, X. Hu, Y. Hu, Y. Hu, R. Hua, T.-Y. Huang, W. Huang, S. V. Huffel, Q. Huo, V. HV, K. M. Iftekharuddin, F. Isensee, M. Islam, A. S. Jackson, S. R. Jambawalikar, A. Jesson, W. Jian, P. Jin, V. J. M. Jose, A. Jungo, B. Kainz, K. Kamnitsas, P.-Y. Kao, A. Karnawat, T. Kellermeier, A. Kermi, K. Keutzer, M. T. Khadir, M. Khened, P. Kickingereder, G. Kim, N. King, H. Knapp, U. Knecht, L. Kohli, D. Kong, X. Kong, S. Koppers, A. Kori, G. Krishnamurthi, E. Krivov, P. Kumar, K. Kushibar, D. Lachinov, T. Lambrou, J. Lee, C. Lee, Y. Lee, M. Lee, S. Lefkovits, L. Lefkovits, J. Levitt, T. Li, H. Li, W. Li, H. Li, X. Li, Y. Li, H. Li, Z. Li, X. Li, Z. Li, X. Li, W. Li, Z.-S. Lin, F. Lin, P. Lio, C. Liu, B. Liu, X. Liu, M. Liu, J. Liu, L. Liu, X. Llado, M. M. Lopez, P. R. Lorenzo, Z. Lu, L. Luo, Z. Luo, J. Ma, K. Ma, T. Mackie, A. Madabushi, I. Mahmoudi, K. H. Maier-Hein, P. Maji, C. Mammen, A. Mang, B. S. Manjunath, M. Marcinkiewicz, S. McDonagh, S. McKenna, R. McKinley, M. Mehl, S. Mehta, R. Mehta, R. Meier, C. Meinel, D. Merhof, C. Meyer, R. Miller, S. Mitra, A. Moiyadi, D. Molina-Garcia, M. A. B. Monteiro, G. Mrukwa, A. Myronenko, J. Nalepa, T. Ngo, D. Nie, H. Ning, C. Niu, N. K. Nuechterlein, E. Oermann, A. Oliveira, D. D. C. Oliveira, A. Oliver, A. F. I. Osman, Y.-N. Ou, S. Ourselin, N. Paragios, M. S. Park, B. Paschke, J. G. Pauloski, K. Pawar, N. Pawlowski, L. Pei, S. Peng, S. M. Pereira, J. Perez-Beteta, V. M. Perez-Garcia, S. Pezold, B. Pham, A. Phophalia, G. Piella, G. N. Pillai, M. Piraud, M. Pisov, A. Popli, M. P. Pound, R. Pourreza, P. Prasanna, V. Prkovska, T. P. Pridmore, S. Puch, Élodie Puybareau, B. Qian, X. Qiao, M. Rajchl, S. Rane, M. Rebsamen, H. Ren, X. Ren, K. Revanuru, M. Rezaei, O. Rippel, L. C. Rivera, C. Robert, B. Rosen, D. Rueckert, M. Safwan, M. Salem, J. Salvi, I. Sanchez, I. Sánchez, H. M. Santos, E. Sartor, D. Schellingerhout, K. Scheufele, M. R. Scott, A. A. Scussel, S. Sedlar, J. P. Serrano-Rubio, N. J. Shah, N. Shah, M. Shaikh, B. U. Shankar, Z. Shboul, H. Shen, D. Shen, L. Shen, H. Shen, V. Shenoy, F. Shi, H. E. Shin, H. Shu, D. Sima, M. Sinclair, O. Smedby, J. M. Snyder, M. Soltaninejad, G. Song, M. Soni, J. Stawiaski, S. Subramanian, L. Sun, R. Sun, J. Sun, K. Sun, Y. Sun, G. Sun, S. Sun, Y. R. Suter, L. Szilagyi, S. Talbar, D. Tao, D. Tao, Z. Teng, S. Thakur, M. H. Thakur, S. Tharakan, P. Tiwari, G. Tochon, T. Tran, Y. M. Tsai, K.-L. Tseng, T. A. Tuan, V. Turlapov, N. Tustison, M. Vakalopoulou, S. Valverde, R. Vanguri, E. Vasiliev, J. Ventura, L. Vera, T. Vercauteren, C. A. Verrastro, L. Vidyaratne, V. Vilaplana, A. Vivekanandan, G. Wang, Q. Wang, C. J. Wang, W. Wang, D. Wang, R. Wang, Y. Wang, C. Wang, G. Wang, N. Wen, X. Wen, L. Weninger, W. Wick, S. Wu, Q. Wu, Y. Wu, Y. Xia, Y. Xu, X. Xu, P. Xu, T.-L. Yang, X. Yang, H.-Y. Yang, J. Yang, H. Yang, G. Yang, H. Yao, X. Ye, C. Yin, B. Young-Moxon, J. Yu, X. Yue, S. Zhang, A. Zhang, K. Zhang, X. Zhang, L. Zhang, X. Zhang, Y. Zhang, L. Zhang, J. Zhang, X. Zhang, T. Zhang, S. Zhao, Y. Zhao, X. Zhao, L. Zhao, Y. Zheng, L. Zhong, C. Zhou, X. Zhou, F. Zhou, H. Zhu, J. Zhu, Y. Zhuge, W. Zong, J. Kalpathy-Cramer, K. Farahani, C. Davatzikos, K. van Leemput, and B. Menze, "Identifying the best machine learning algorithms for brain tumor segmentation, progression assessment, and overall survival prediction in the brats challenge," 2019.

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

[3] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, L. Kaiser, and I. Polosukhin, "Attention is all you need," 2023.

[4] A. Gu, K. Goel, and C. Ré, "Efficiently modeling long sequences with structured state spaces," 2022.

[5] A. Gu and T. Dao, "Mamba: Linear-time sequence modeling with selective state spaces," 2023.

[6] H. Zhao, J. Shi, X. Qi, X. Wang, and J. Jia, "Pyramid scene parsing network," in *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 6230–6239, 2017.

[7] R. Wu, Y. Liu, P. Liang, and Q. Chang, "Ultralight vm-unet: Parallel vision mamba significantly reduces parameters for skin lesion segmentation," 2024.

[8] M. Buda, "Brain mri segmentation," May 2019.

[9] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," 2015.

[10] J. Ma, F. Li, and B. Wang, "U-mamba: Enhancing long-range dependency for biomedical image segmentation," 2024.

[11] A. Dosovitskiy, L. Beyer, A. Kolesnikov, D. Weissenborn, X. Zhai, T. Unterthiner, M. Dehghani, M. Minderer, G. Heigold, S. Gelly, J. Uszkoreit, and N. Houlsby, "An image is worth 16x16 words: Transformers for image recognition at scale," 2021.

[12] J. Ho, A. Jain, and P. Abbeel, "Denoising diffusion probabilistic models," 2020.

[13] E. Xie, W. Wang, Z. Yu, A. Anandkumar, J. M. Alvarez, and P. Luo, "Segformer: Simple and efficient design for semantic segmentation with transformers," 2021.

APPENDIX

A. Training Pipeline

The training pipeline for our model consists of the following steps:

1. Data preprocessing: The dataset is split into training, validation, and test sets. The images are resized to a uniform size and normalized.

2. Model architecture: We employ a convolutional neural network (CNN) with the following layers: [Describe the layers and their configurations].

3. Training: The model is trained using the training set for 100 epochs with a batch size of 32. We use the Adam optimizer with a learning rate of 0.0003.

4. Evaluation: The trained model is evaluated on the validation set to monitor its performance and prevent overfitting. We use the dice loss metric.

5. Testing: Once the model achieves satisfactory performance on the validation set, it is tested on the held-out test set to assess its generalization ability.

B. Supplementary Figures

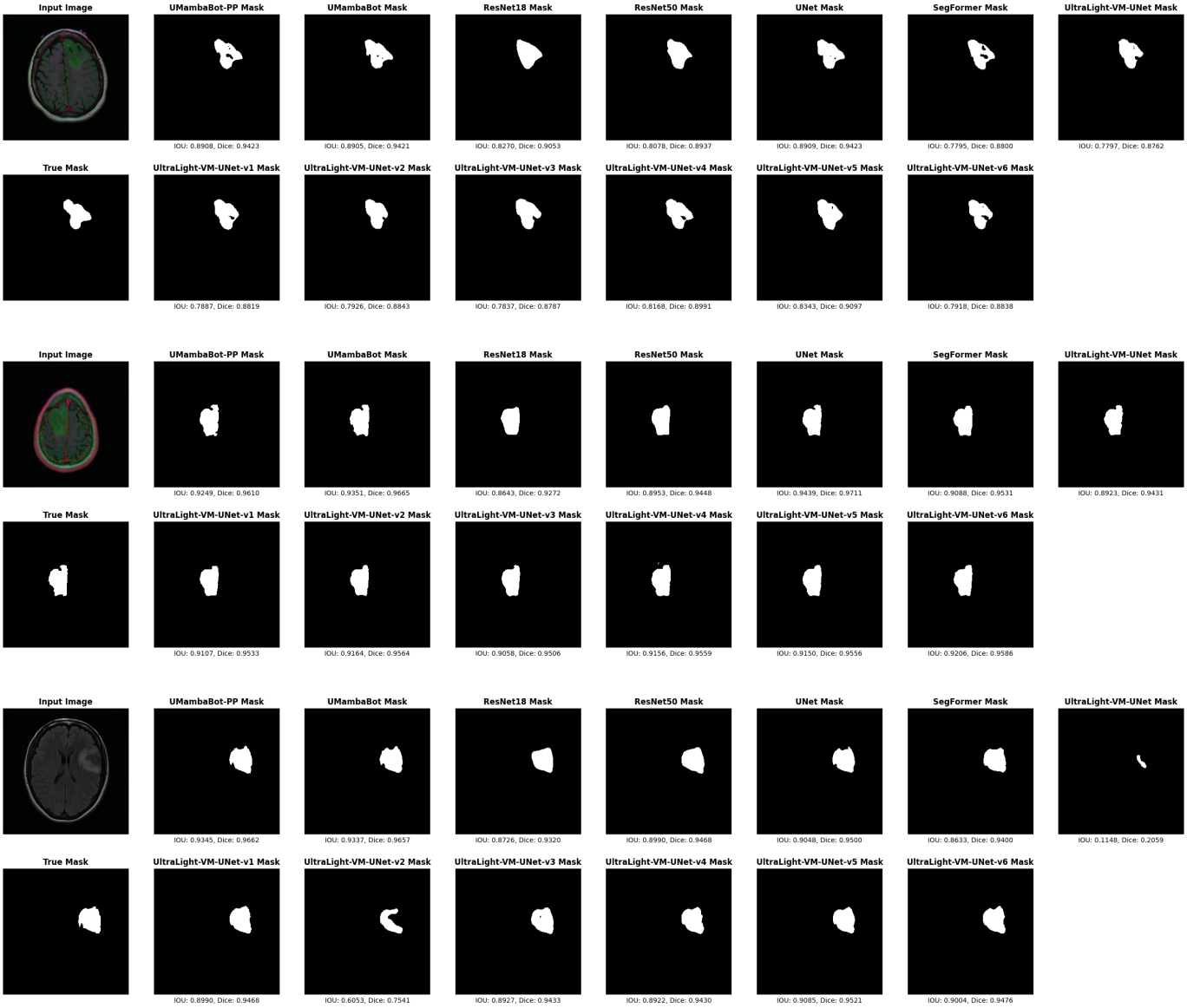

Fig. 4: In some cases, such as this one, it was observed that models with too few parameters (UltraLight-VM-UNet, UltraLight-VM-UNet-v2) don't detect sections of the tumor during segmentation.

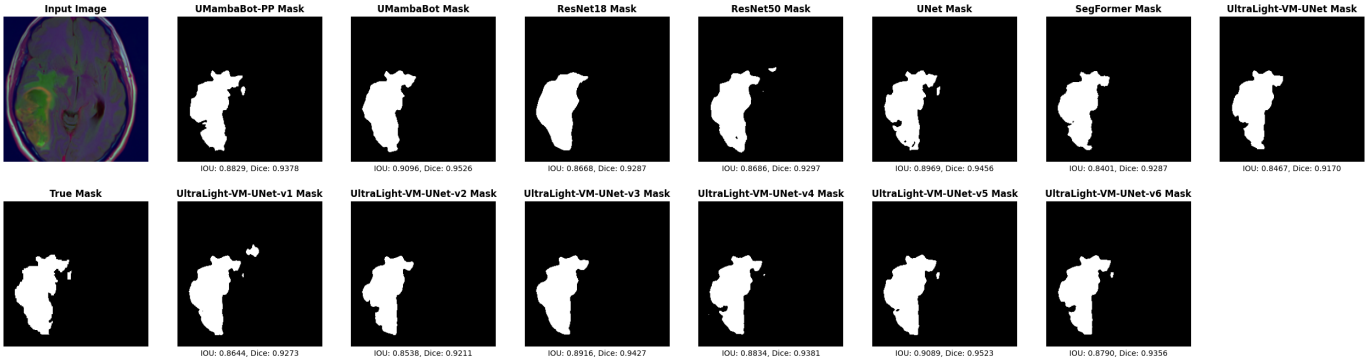

Fig. 5: UltraLight-VM-UNet-v5 achieved segmentation results equivalent to UMambaBot with approximately 9.1 M less parameters.

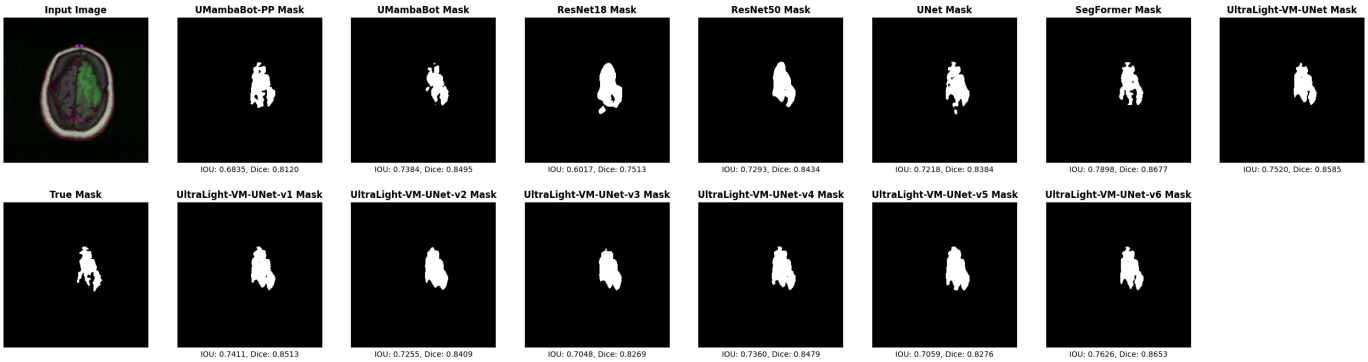

Fig. 6: An example of models with pyramidal pooling in the PVM blocks (UltraLight-VM-UNet-v4, UltraLight-VM-UNet-v6) achieved an improvement over a model without (UltraLight-VM-UNet-v5).