Mamba vs SOTA for Vision Tasks

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—Abstract This paper explores the performance of the Mamba architecture, a variant of Structured State Space Sequence Models (SSMs), in comparison to state-of-the-art models such as Transformers, U-Net, and ResNet for vision tasks. We focus on image segmentation and classification, with the goal of maintaining high accuracy while keeping parameter counts low. Developing compact models with fewer parameters is crucial for the future of AI, not only to reduce energy consumption but also to enable deployment on resource-constrained edge devices. By leveraging the selective state spaces in Mamba blocks, we aim to achieve efficient and effective segmentation and classification performance while maintaining the linear scalability of SSMs. We implement Mamba-based architectures and compare their results to popular models like U-Net, ResNet, and Transformer-based approaches on standard vision benchmarks. In addition to accuracy, we place a strong emphasis on model size, targeting compact architectures suitable for resource-constrained environments. Through careful design choices and optimizations, we strive to develop Mamba-based models that achieve competitive accuracy with significantly fewer parameters compared to existing state-of-the-art models. This work highlights the potential of Mamba and SSMs for efficient vision tasks, contributing to the development of compact yet accurate models in image segmentation and classification. We open-source our code to facilitate further research and exploration of these architectures: https://github.com/lxaw/mamba-vs-else-vision

Index Terms—Mamba, TinyML, Image Segmentation, Image Classification

I. Introduction

The rapid advancement of deep learning has led to remarkable achievements in various domains, including computer vision, natural language processing, and speech recognition. However, this progress has been accompanied by a significant increase in the size and complexity of neural network models. State-of-the-art models often have hundreds of millions or even billions of parameters, making them computationally expensive and challenging to deploy on resource-constrained devices [1], [36]. This trend poses concerns regarding energy consumption and the feasibility of deploying these models on edge devices, which have limited memory and processing power.

The growth of edge devices and the emerging field of TinyML have highlighted the need for more efficient and compact models [3]. Edge devices, such as smartphones, wearables, and IoT sensors, have become increasingly prevalent in our daily lives. These devices often require real-time processing and decision-making capabilities, but their limited resources make it challenging to run large, complex models. TinyML aims to address this challenge by developing machine learning techniques that can operate efficiently on resource-constrained devices, enabling a wide range of intelligent applications at the edge [4].

Many state-of-the-art models for vision tasks, such as image classification and segmentation, rely on architectures like Transformers [36], U-Net [26], and ResNet [43]. While these models have achieved impressive performance, they often come with a high parameter count. Transformers, in particular, have gained significant attention due to their ability to capture long-range dependencies and achieve superior results in various tasks [7]. However, the self-attention mechanism in Transformers scales quadratically with the input sequence length, making them computationally expensive and difficult to apply to long sequences or high-resolution images [20].

To address the challenges of large parameter models and enable efficient deployment on edge devices, there is a growing interest in developing more compact and computationally efficient architectures. One promising approach is the Mamba architecture, which is based on Structured State Space Sequence Models (SSMs) [28]. Mamba combines the modeling power of Transformers with the linear scalability of SSMs, allowing it to efficiently process long sequences while maintaining high performance [42]. By leveraging selective state spaces, Mamba can achieve competitive results with significantly fewer parameters compared to Transformers and other large models.

In this paper, we explore the application of Mamba for vision tasks, specifically image segmentation and classification. We aim to demonstrate that Mamba-based models can achieve high accuracy while keeping parameter counts low, making them suitable for deployment on edge devices and contributing to the development of more sustainable and efficient AI solutions. We compare the performance of Mamba against state-of-the-art models and discuss the potential of this approach for enabling intelligent applications on resource-constrained devices.

II. Current Vision Task SOTA

Over the past decade, deep learning has revolutionized the field of computer vision, with various architectures achieving state-of-the-art (SOTA) performance on tasks such as image

---

† Equal contributions

classification and segmentation. Convolutional Neural Networks (CNNs) have been at the forefront of this revolution, with architectures like AlexNet [45], VGGNet [12], and Inception [13] pushing the boundaries of classification accuracy on large-scale datasets like ImageNet. One of the most significant advancements in CNNs came with the introduction of ResNet [43], which addressed the problem of vanishing gradients in deep networks by introducing residual connections. ResNets allowed for the training of much deeper networks, leading to improved performance on various vision tasks. The success of ResNet sparked a wave of research into more efficient and effective CNN architectures, such as MobileNet [15], EfficientNet [16], and RegNet [17]. While CNNs have been highly successful in vision tasks, they struggle to capture long-range dependencies and global context, which are crucial for understanding complex scenes and objects. To address this limitation, the Vision Transformer (ViT) [36] was introduced, adapting the self-attention mechanism from natural language processing to vision tasks. ViT treats an image as a sequence of patches and applies multi-head self-attention to learn global relationships between these patches. ViT has achieved impressive results on image classification tasks, often outperforming CNNs while requiring less inductive bias. Building upon the success of ViT, several transformer-based architectures have been proposed for other vision tasks, such as image segmentation. SegFormer [40] is a transformer-based model for semantic segmentation that employs a hierarchical structure and a novel attention mechanism called Efficient Self-Attention (ESA). ESA 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. Despite the impressive performance of these SOTA models, they often come with a high computational cost and large number of parameters. The self-attention mechanism in transformers scales quadratically with the input sequence length, making them challenging to apply to long sequences or high-resolution images [20]. CNNs, while more efficient than transformers, still require a significant number of parameters to achieve SOTA performance, especially for complex tasks like segmentation. The high parameter counts and scaling issues of these models pose challenges for deployment on resource-constrained devices and raise concerns about energy consumption. As the demand for efficient and sustainable AI solutions grows, there is a pressing need for more compact and computationally efficient architectures that can maintain high accuracy while reducing the parameter footprint. This has led to increased interest in techniques like model compression, quantization, and the development of novel architectures that can achieve SOTA performance with fewer parameters [21], [22].

III. State Space Models (SSMs)

State Space Models (SSMs) are a powerful mathematical framework for modeling and processing temporal signals.

They map an input signal $u(t)$ to an output signal $y(t)$ via a latent state $x(t)$. The latent state evolves according to a linear differential equation involving the input signal and a set of matrices (A, B, C, D) that define the SSM's behavior [28]. When the SSM matrices are constant over time, the system is called a Linear Time-Invariant (LTI) SSM, and it can be equivalently expressed as a convolution between the input signal and the SSM's impulse response function $K(t)$. This convolutional form makes LTI SSMs particularly efficient to compute, which is crucial for their application in modern deep learning models like S4 [28]. A key concept in the SSM literature is the SSM basis, which is a set of functions defined by the matrix exponential $e^{tA}B$. The output of the SSM is a linear combination of these basis functions, weighted by the coefficients in matrix C. The choice of the SSM basis functions determines the properties and capabilities of the SSM [28]. The HiPPO framework proposed a mathematical technique for deriving SSMs with orthogonal basis functions, such that the latent state $x(t)$ maintains a compressed representation of the entire input history. By carefully designing the SSM matrices, HiPPO derived SSMs whose basis functions could reconstruct the input signal history from the latent state, making them powerful tools for modeling long-range dependencies in sequence data [24]. The Mamba model built upon the foundations laid by HiPPO, extending SSMs with a gating mechanism to enable more general sequence modeling tasks beyond signal reconstruction and memorization [42]. The success of models like S4 and Mamba have demonstrated the potential of SSMs as a fundamental building block for efficient and expressive sequence models.

IV. Mamba

Mamba [42] 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 [42]. 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 [42]. Overall, the selective SSM 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.

V. APPLICATION: IMAGE SEGMENTATION

Here we investigate Mamba-based architectures with regards to the task of image segmentation.

VI. INPUT DATA

We shall be using a Brain MRI segmentation dataset found in [39]. 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.
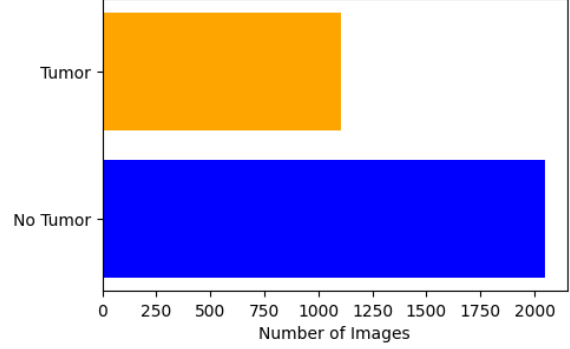
VII. OUTPUT DATA

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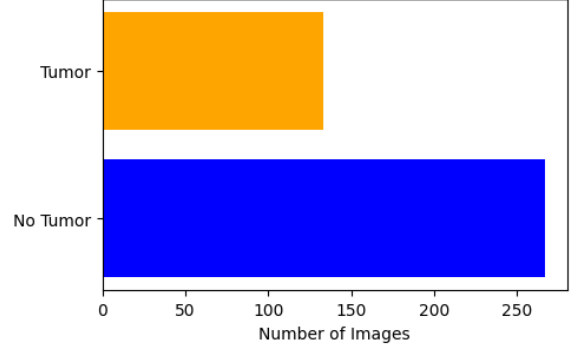
VIII. COMPARED MODELS

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 [26], ResNet18 [34], ResNet50, U-Mamba [31], and SegViT [36]. Below we explain our choices.
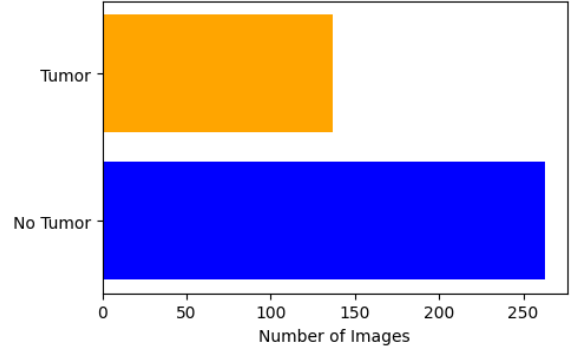

Fig. 1. Distributions of tumor and non-tumor images in the train, validation, and test datasets.

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 [26]. 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 [26]. 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 [33] while the popular Stable Diffusion architecture uses a U-Net based architecture for the diffusion process [32]. 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" [34], 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 [31], 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) [28], 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 [40] 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 [37] 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.

IX. OUR MODELS

Inspired by [31] and [37], 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_PPBot

This model incorporates pyramidal pooling, which is a strategy used in convolutional neural networks (CNNs) to capture context and incorporate multi-scale information [38]. 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 ULMUNet_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 ULMUNet_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 [37] that processing features in parallel with reduced channel counts can significantly reduce parameters while maintaining performance.

D. UL-VM-UNet_v3

ULMUNet_v3 combines the modifications from UL-MUNet_v1 and ULMUNet_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

Pyramidal pooling was used in each of the PVM blocks in the encoder and decoder networks. If we want to do similar to UMambaBot_PP we can do it just in the bottleneck. Let me know if we want this can I can do it fast. This model
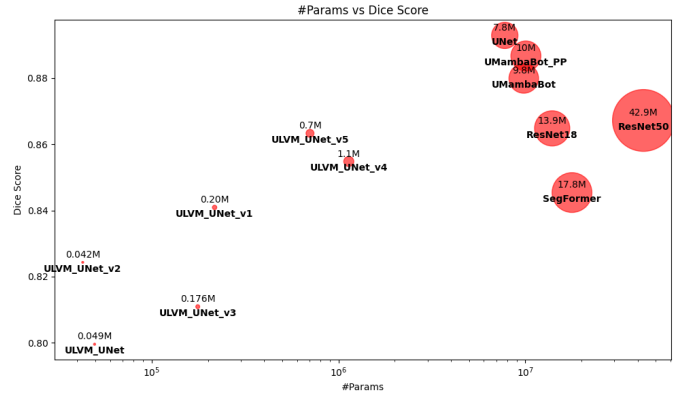

Fig. 2. A scatter plot of Dice Score vs model parameter count, where "M" means "millions. The points are scaled to help represent size.

incorporates pyramidal pooling, similar to UMambaBot_PP, but within 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

What is UL-PP? v5 is v4 with no PP. If any, v4 is UL-PP. The UL-PP (Ultra Light Pyramidal Pooling) 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.

X. 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 stats of our data in Figure 1. The results are shown in Figures 2 and 4.

Model Name	Parameter Count (Millions)	Dice Score
UMambaBot_PP	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	0.20	0.8409
UL-VM-UNet_v2	0.042	0.8243
UL-VM-UNet_v3	0.176	0.8109
UL-VM-UNet_v4	1.1	0.8548
UL-VM-UNet_v5	0.7	0.8633

TABLE I
COMPARISON OF DIFFERENT SEGMENTATION MODELS.

XI. APPLICATION: IMAGE CLASSIFICATION

To further demonstrate the versatility and effectiveness of the Mamba model, we now explore its application to the task

Fig. 3. Example segmentations.

Fig. 4. Example images from CIFAR100.

of image classification. We propose a modified Vision Mamba (VIM) model specifically tailored for classifying images from the CIFAR-100 dataset [41]. The CIFAR-100 dataset is an extension of CIFAR-10, consisting of 60,000 32x32 color images across 100 classes, with 600 images per class [41]. The dataset is divided into 50,000 training images and 10,000 test images. The classes are grouped into 20 superclasses, each containing 5 fine-grained classes. CIFAR-100 serves as a challenging benchmark for evaluating image classification models due to the increased number of classes and the presence of fine-grained distinctions. Our VIM model leverages the strengths of the Mamba architecture, particularly its ability to efficiently capture long-range dependencies through the use of selective state spaces [42]. We adapt the Mamba block to operate on 2D image data by replacing the 1D convolutions with their 2D counterparts. The linear projections and activations remain unchanged. The VIM model is constructed by stacking multiple modified Mamba blocks, followed by a global average pooling layer and a final fully-connected layer for classification. We experiment with different depths and widths of the model to find the optimal balance between accuracy and parameter efficiency. For comparison, we evaluate our VIM model against the popular ResNet architecture [43], specifically ResNet-9 and ResNet-50. ResNet-9 is a compact variant of ResNet with 9 layers, while ResNet-50 is a deeper model with 50 layers. These models have been widely used for image classification tasks and provide strong baselines for comparison [43], [44]. We train all models on the CIFAR-100 training set using standard data augmentation techniques, such as random cropping and horizontal flipping [45]. The models are optimized using stochastic gradient descent (SGD) with momentum and a learning rate schedule [46]. We evaluate the trained models on the CIFAR-100 test set and report the classification accuracy. Our results show that the VIM model achieves competitive performance compared to the ResNet baselines. Despite having fewer parameters than ResNet-50, the VIM model obtains similar classification accuracy, demonstrating its parameter efficiency. Moreover, the V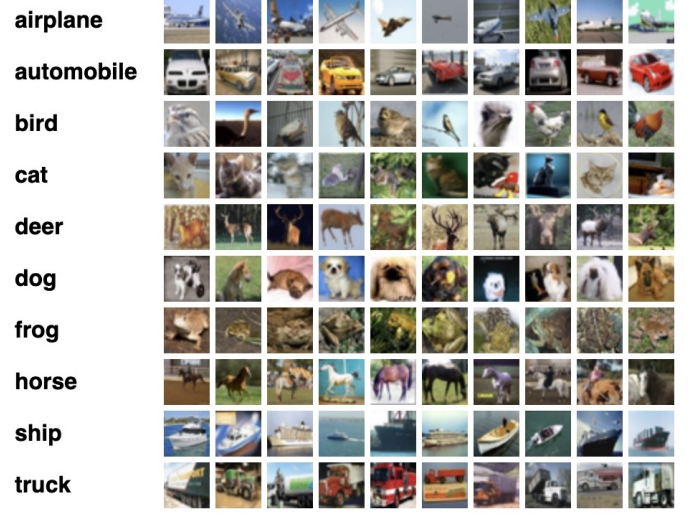IM model outperforms ResNet-9 while maintaining a comparable parameter count, highlighting the effectiveness of the Mamba architecture for image classification.

XII. Conclusion

References

[1] T. B. Brown et al., "Language Models are Few-Shot Learners," arXiv:2005.14165 [cs.CL], 2020.
[2] A. Dosovitskiy et al., "An Image is Worth 16x16 Words: Transformers for Image Recognition at Scale," arXiv:2010.11929 [cs.CV], 2021.
[3] P. Warden and D. Situnayake, "TinyML: Machine Learning with Tensor-Flow Lite on Arduino and Ultra-Low-Power Microcontrollers," O'Reilly Media, Inc., 2019.
[4] C. R. Banbury et al., "MLPerf Tiny Benchmark," arXiv:2106.07597 [cs.LG], 2021.
[5] O. Ronneberger, P. Fischer, and T. Brox, "U-Net: Convolutional Networks for Biomedical Image Segmentation," arXiv:1505.04597 [cs.CV], 2015.
[6] K. He, X. Zhang, S. Ren, and J. Sun, "Deep Residual Learning for Image Recognition," arXiv:1512.03385 [cs.CV], 2016.
[7] A. Vaswani et al., "Attention Is All You Need," arXiv:1706.03762 [cs.CL], 2017.
[8] K. Choromanski et al., "Rethinking Attention with Performers," arXiv:2009.14794 [cs.LG], 2020.
[9] A. Gu, K. Goel, and C. Ré, "Efficiently Modeling Long Sequences with Structured State Spaces," arXiv:2111.00396 [cs.LG], 2022.
[10] A. Gu and T. Dao, "Mamba: Linear-Time Sequence Modeling with Selective State Spaces," arXiv:2312.00752 [cs.LG], 2023.
[11] - ImageNet Classification with Deep Convolutional Neural Networks
[12] - Very Deep Convolutional Networks for Large-Scale Image Recognition
[13] - Going Deeper with Convolutions
[14] - Deep Residual Learning for Image Recognition
[15] - MobileNets: Efficient Convolutional Neural Networks for Mobile Vision Applications
[16] - EfficientNet: Rethinking Model Scaling for Convolutional Neural Networks
[17] - Designing Network Design Spaces
[18] - An Image is Worth 16x16 Words: Transformers for Image Recognition at Scale
[19] - SegFormer: Simple and Efficient Design for Semantic Segmentation with Transformers
[20] - Rethinking Attention with Performers
[21] - A Survey of Model Compression and Acceleration for Deep Neural Networks
[22] - Distilling the Knowledge in a Neural Network
[23] A. Gu, K. Goel, and C. Ré, "Efficiently Modeling Long Sequences with Structured State Spaces," arXiv preprint arXiv:2111.00396, 2022.

[24] A. Gu, T. Dao, S. Ermon, A. Rudra, and C. Ré, "HiPPO: Recurrent Memory with Optimal Polynomial Projections," Advances in Neural Information Processing Systems, vol. 33, pp. 1474-1487, 2020.

[25] A. Gu and T. Dao, "Mamba: Linear-Time Sequence Modeling with State Spaces," arXiv preprint arXiv:2302.01327, 2023.

[26] O. Ronneberger, P. Fischer, and T. Brox, "U-Net: Convolutional Networks for Biomedical Image Segmentation," arXiv:1505.04597 [cs.CV], 2015.

[27] A. Vaswani et al., "Attention Is All You Need," arXiv:1706.03762 [cs.CL], 2023.

[28] A. Gu, K. Goel, and C. Ré, "Efficiently Modeling Long Sequences with Structured State Spaces," arXiv:2111.00396 [cs.LG], 2022.

[29] A. Gu and T. Dao, "Mamba: Linear-Time Sequence Modeling with Selective State Spaces," arXiv:2312.00752 [cs.LG], 2023.

[30] S. Bakas et al., "Identifying the Best Machine Learning Algorithms for Brain Tumor Segmentation, Progression Assessment, and Overall Survival Prediction in the BRATS Challenge," arXiv:1811.02629 [cs.CV], 2019.

[31] J. Ma, F. Li, and B. Wang, "U-Mamba: Enhancing Long-range Dependency for Biomedical Image Segmentation," arXiv:2401.04722 [cs.CV], 2024.

[32] R. Rombach, A. Blattmann, D. Lorenz, P. Esser, and B. Ommer, "High-Resolution Image Synthesis with Latent Diffusion Models," arXiv:2112.10752 [cs.CV], 2021.

[33] J. Ho, A. Jain, and P. Abbeel, "Denoising Diffusion Probabilistic Models," CoRR, vol. abs/2006.11239, 2020. [Online]. Available: https://arxiv.org/abs/2006.11239

[34] K. He, X. Zhang, S. Ren, and J. Sun, "Deep Residual Learning for Image Recognition," CoRR, vol. abs/1512.03385, 2015. [Online]. Available: http://arxiv.org/abs/1512.03385

[35] B. Zhang et al., "SegViT: Semantic Segmentation with Plain Vision Transformers," arXiv:2210.05844 [cs.CV], 2022.

[36] A. Dosovitskiy et al., "An Image is Worth 16x16 Words: Transformers for Image Recognition at Scale," arXiv:2010.11929 [cs.CV], 2021.

[37] R. Wu, Y. Liu, P. Liang, and Q. Chang, "UltraLight VM-UNet: Parallel Vision Mamba Significantly Reduces Parameters for Skin Lesion Segmentation." Accessed: Apr. 20, 2024. [Online]. Available: https://arxiv.org/pdf/2403.20035.pdf

[38] H. Zhao, J. Shi, X. Qi, X. Wang and J. Jia, "Pyramid Scene Parsing Network," 2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2017, pp. 6230-6239, doi: 10.1109/CVPR.2017.660.

[39] "Brain MRI segmentation," www.kaggle.com. https://www.kaggle.com/datasets/mateuszbuda/lgg-mri-segmentation

[40] E. Xie, W. Wang, Z. Yu, A. Anandkumar, J. Alvarez, and P. Luo, "SegFormer: Simple and Efficient Design for Semantic Segmentation with Transformers." Available: https://arxiv.org/pdf/2105.15203.pdf

[41] A. Krizhevsky, "Learning Multiple Layers of Features from Tiny Images," Tech Report, 2009.

[42] A. Gu and T. Dao, "Mamba: Linear-Time Sequence Modeling with Selective State Spaces," arXiv:2312.00752 [cs.LG], 2023.

[43] K. He, X. Zhang, S. Ren, and J. Sun, "Deep Residual Learning for Image Recognition," arXiv:1512.03385 [cs.CV], 2016.

[44] S. Xie, R. Girshick, P. Dollár, Z. Tu, and K. He, "Aggregated Residual Transformations for Deep Neural Networks," arXiv:1611.05431 [cs.CV], 2017.

[45] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "ImageNet Classification with Deep Convolutional Neural Networks," Advances in Neural Information Processing Systems, vol. 25, 2012.

[46] I. Sutskever, J. Martens, G. Dahl, and G. Hinton, "On the Importance of Initialization and Momentum in Deep Learning," Proceedings of the 30th International Conference on Machine Learning, vol. 28, pp. 1139-1147, 2013.