Improving State Space Models for Computer Vision

Jing Ni
New York University
New York, USA
jn2901@nyu.edu

Lucas Martinez
New York University
New York, USA
lom2017@nyu.edu

Marie Lou Panthagani
New York University
New York, USA
mp6828@nyu.edu

Riley Daggs
New York University
New York, USA
rpd4362@nyu.edu

Siddharth Agrawal
New York University
New York, USA
sa3117@nyu.edu

***Abstract*—This work explores enhancements to state-space models (SSMs) for computer vision tasks, specifically through experiments with Vision Mamba (Vim) and MambaVision. For Vim, we explored several enhancements to improve accuracy and efficiency. We integrated Squeeze-and-Excitation blocks to recalibrate channel-wise feature responses, replaced standard convolutions with MobileNetV2-inspired depthwise convolutions to reduce computational overhead, and applied pruning techniques including: dynamic pruning Iterative Magnitude Pruning, the Early-Bird Lottery Ticket Hypothesis, and gradient-based pruning. For MambaVision, our contributions are twofold: (1) integrating a sliding window attention (SWA) mechanism to expand the model's receptive field for high-resolution images and (2) replacing the default S6 kernel with S4 and S5 kernels, which have shown superior performance in continuous signal processing tasks. These approaches aim to enhance the capacity of SSMs in vision tasks that demand large receptive fields and continuous-like signal processing.**

Note: all code and resources used for this work are publicly available in our GitHub repository [1], ensuring transparency and reproducibility.

I. Introduction

State space models (SSMs) have recently gained traction in computer vision for their ability to model long-range dependencies in sequential data [1], [2]. Vision Mamba (Vim) [3] serves as an initial exploration into integrating SSMs for visual data processing. Vim employs bidirectional state-space models to process input patches efficiently while maintaining a lightweight architecture. Specifically, Vim consists of Conv1D layers, state-space models (SSMs), and a gating mechanism to extract features in both forward and backward directions, combined with a residual connection for stable learning. The Vision Mamba block just described is illustrated in Figure 1, and forms the core of Vim's architecture.

To evaluate Vision Mamba, we performed the following experiments:

- **Squeeze-and-Excitation (SE) [4] Integration:** The SE block was integrated into Vim to recalibrate channel-wise feature responses. While accuracy decreased slightly due to overfitting, it demonstrated the SE block's capacity to enhance feature representation.
- **MobileNetV2-Inspired Depthwise Convolutions [5]:** Replacing standard convolutions with depthwise convolutions reduced parameter count while maintaining performance, showcasing Vim's efficiency.
- **Sparse Subnetwork Exploration:** Techniques such as Dynamic Pruning with Cosine Scheduler, Iterative Magnitude Pruning (IMP) [6], Gradient Pruning, and Early-Bird Lottery Ticket Hypothesis [7] were applied to identify sparse subnetworks within Vim.

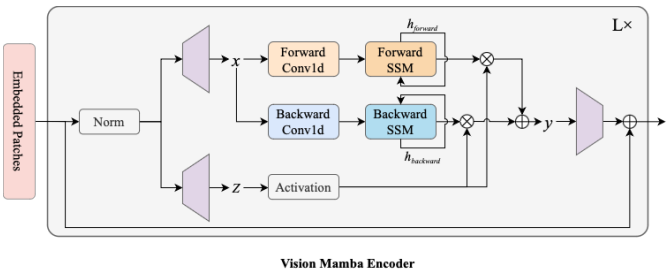

Fig. 1. The Vision Mamba Encoder Block (VisionEncoderMambaBlock)

While Vision Mamba provided a strong foundation for efficient vision modeling, its performance highlighted areas for improvement, particularly in handling larger datasets and high-resolution inputs. Building on this foundation, Ali Hatamizadeh and Jan Kautz introduced MambaVision [8], which employs a hybrid Vision Transformer [9] and Mamba [1] based architecture, and has achieved state-of-the-art performance in large-scale image classification tasks, particularly on datasets like ImageNet. However, further exploration is necessary to improve the model's handling of high-resolution inputs and continuous signal-like images.

We propose two modifications to MambaVision to address these gaps:

1) **Sliding Window Attention (SWA)**: Inspired by Samba, a model that improves context length and perplexity in language modeling through adding sliding window attention (SWA) to Mamba, we hypothesize that SWA will increase MambaVision's receptive field. This enhancement could be particularly effective in tasks that require processing large, high-resolution images, such as medical microscopy and pathology.
2) **Registers**: Registers performed well in [10] so we want to experiment and see if they also perform well with MambaVision.
3) **Relative Atention**: Relative Attention as proposed by [11] merges the strengths of convolutions (translational

[1]GitHub repository for this project: https://github.com/rdaggs/vim_ssm_cv24

equivariance) with strengths of self-attention (global receptive field and input independent weighting) to improve generalisation across all dataset sizes. We aim to use this in MambaVision to improve its generalisation capability on smaller datasets like CIFAR100.

By building upon Vim and extending MambaVision with these modifications, we aim to explore how these models can be further optimized for vision tasks. Our experiments focus on enhancing accuracy, efficiency, and generalization. The following sections provide a detailed description of our methods, experimental results, and insights gained from these explorations.

II. Related Work

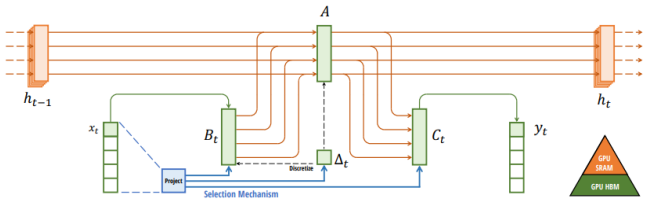

Fig. 2. **Mamba Overview**: Structured SSMs independently map each channel (e.g. $D = 5$) of an input x to output y through a higher dimensional latent state h (e.g. $N = 4$). Prior SSMs avoid materializing this large effective state (DN, times batch size B and sequence length L) through clever alternate computation paths requiring time-invariance: the (Δ, A, B, C) parameters are constant across time. Our selection mechanism adds back input-dependent dynamics, which also requires a careful hardware-aware algorithm to only materialize the expanded states in more efficient levels of the GPU memory hierarchy.

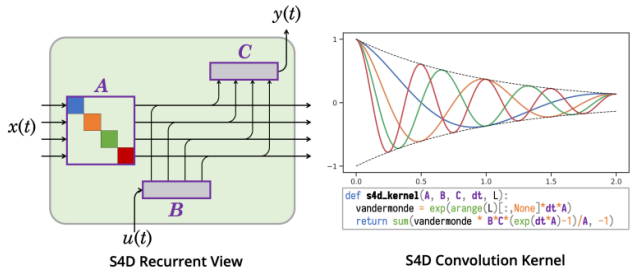

Fig. 3. **The S4 kernel** combines recurrent state-space modeling with convolution. In the recurrent view, the hidden state evolves as $u(t+1) = Au(t) + Bx(t)$, and the output is $y(t) = Cu(t)$. The convolution kernel computes the state evolution using the matrix exponential $e^{A\Delta t}$ and is given by $\text{sum}(V \cdot BC^T(e^{\Delta t A})/A)$, capturing long-range dependencies efficiently. Legendre polynomials are used as the basis functions for diagonalizing A

A. History of State Space Models:

State Space Models (SSMs) have long been explored for sequence modeling due to their mathematical ability to capture long-range dependencies, but early models faced scalability issues in terms of computation and memory for very long sequences. The Structured State Space model (S4) [2]. introduced a breakthrough by parameterizing the state matrix A with a low-rank correction, making it computationally efficient through the reduction to a Cauchy kernel. They achieve this efficiency by restricting the state matrix A to a specific basis function, such as Legendre polynomials or sinusoidal functions, which allows the model to approximate the sequence history in a compact and computationally tractable manner. This enabled S4 to excel on tasks requiring long-range dependencies, setting new benchmarks across a variety of domains, most notable of which is the Long Range Arena benchmark [12], which were a set of benchmarks modeling long-range sequences varying from $1K$ to $16K$ tokens, dominated mostly by transformers up till that point.

Building on this, [13] developed the S5: Simplified State Space Layers for Sequence Modeling. They enhanced the SSM approach in S4 by using multi-input, multi-output models, while retaining S4's efficiency through parallelization, achieving strong results in long-sequence tasks. The S5 also formulated a simpler formulation that could be computed without the use of FFT and Legendre polynomials as was required in S4.

Most recently, [1] prposed Mamba with S6, addressing SSM limitations in content-based reasoning via a novel "selection mechanism". This innovation combined with efficient hardware-aware algorithms led to state-of-the-art performance on tasks across various modalities, rivaling Transformers in both speed and accuracy on long sequences. This model boasts linear time complexity and either surpasses or matches the performance of Transformers in various language modeling tasks. Mamba's key innovation lies in a unique selection mechanism that allows for efficient processing of long sequences, while also taking into account the specific characteristics of the GPU hardware for efficient inference and training. Together, these advances mark a significant evolution in the practical use of SSMs for sequence modeling.

S4ND [14] pioneered the first successful application of State Space Models (SSMs) in Computer Vision. This was achieved by extending the one-dimensional S4 kernel to N dimensions through an outer product operation between N 1-Dimensional S4 kernels. Furthermore, by constraining the matrix A to coefficients of sinusoidal basis functions, the authors effectively implemented a running Fourier Transform on the image. This innovative approach enabled S4ND to match the accuracy of ConvNext by simply substituting its convolutional blocks with S4ND kernels. Notably, when applied to video analysis, S4ND surpassed the performance of an inflated 3D ConvNeXt model by 4% on the HMDB-51 activity classification benchmark.

[8] introduced a novel hybrid vision backbone called MambaVision, that integrates the Mamba architecture with Vision Transformers (ViT) to enhance visual feature modeling. By incorporating self-attention blocks in the final layers, MambaVision effectively captures long-range spatial dependencies, while a novel mixer block with a symmetric path further improves global context modeling. Their hierarchical MambaVision models achieve State-of-the-Art (SOTA) performance in image classification on the ImageNet-1K dataset and outperform comparable backbones in object detection, instance segmentation, and semantic segmentation on the MS COCO and ADE20K datasets.

Samba [15] demonstrated that hybrid architectures combining Mamba with Sliding Window Attention (SWA) can improve memory recall and context length, particularly in language models. Given the structural similarities between text and high-resolution image data in terms of spatial relationships, there is a strong motivation to apply SWA in vision models to handle large image resolutions.

The Mamba model has introduced a novel approach to SSMs by utilizing the S6, which is equipped with a selection mechanism that excels in handling discrete-time data, particularly text-based tasks. However, the authors noted a limitation of S6, specifically that S4 and S5 were shown to have superior performance in continuous-time signal tasks. They demonstrated this with a benchmark on audio processing task where Mamba-S4 outpeformed the Mamba-S6. The paper attributes this to the "No Free Lunch" theorem and that the inductive biases of S4/S5 allow it to better adapt to continuous-time signals like audio whereas S6 adapts better to discrete-time signals like text with its "selection mechanism". This shows the trade-offs between continuous and discrete time modeling in SSMs. Continuous-time data such as audio benefits from SSMs like S4 and S5, while discrete-time data such as text sees improved performance with S6's "selection mechanism". This raises the question of whether images, often considered continuous-time signals in 2-Dimensions, could similarly benefit from S4/S5. We would like to explore this nuance in our work, which remains underexplored in the context of computer vision.

B. History of Mamba based Architectures for Vision

Mamba-based architectures have been explored for vision tasks with various modifications to enhance global context understanding and computational efficiency. Vim or VisionMamba [3] proposed a bidirectional SSM-based framework to process tokens in forward and backward directions, enhancing spatial understanding by leveraging more global context.

EfficientVMamba [16] employs an atrous-based selective scan with skip sampling, combining hierarchical SSM and CNN blocks. SSMs process higher-resolution inputs for global context, while CNNs handle lower resolutions.

VMamba [17] introduces a Cross-Scan Module (CSM) utilizing a four-way selective scan approach to expand the global receptive field and capture surrounding token information. Additional architectural adaptations, such as depth-wise convolutions and a hierarchical structure, enhance vision task suitability.

Additionally, register tokens [10], originally introduced in [18], have been found effective in Mamba-based architectures.

C. Self-Attention Variants for Computer Vision

Transformer-based architectures have become a cornerstone in computer vision research, with significant developments in hybrid and hierarchical designs. The Vision Transformer (ViT) [19] pioneered the use of pure attention mechanisms in vision tasks, demonstrating remarkable performance on large datasets but requiring extensive pretraining due to its lack of inductive biases. Swin Transformer [20] addressed these limitations by introducing a hierarchical design with shifted windows for localized attention, enabling scalability to dense prediction tasks while maintaining computational efficiency. CoAtNet [11] further reduced the requirement of extensive pertaining and bridged the generalisation gap by combining convolutional and attention mechanisms by vertically stacking depthwise convolutions and attention layers, as well as utilising a variant of relative attention, improved generalization and efficiency.

III. METHODS

In this section we outline the methodologies employed to enhance the performance and efficiency of computer vision models within the Mamba framework. Our approach leverages advanced pruning techniques, lightweight architectures, and state-of-the-art feature extraction mechanisms to address the challenges of computational complexity and resource constraints in vision tasks. We categorize our methods into two primary experimental pipelines: Vision Mamba and Mamba Vision, each focusing on distinct aspects of architectural optimization and integration.

A. Vision Mamba

Vision Mamba (Vim) [3] redefines visual representation learning by utilizing state space models (SSMs) to process images as sequential data. Unlike traditional Transformers, Vim replaces self-attention with bidirectional SSMs, achieving efficient global context modeling while significantly reducing computational and memory costs.

To handle spatial information, Vim incorporates position embeddings and processes images by splitting them into patches, projecting these patches into token sequences, and using bidirectional SSMs to extract features. This design enables Vim to excel in dense prediction tasks such as segmentation and object detection while maintaining scalability for high-resolution images.

The following subsections detail the key techniques we experimented with to improve Vision Mamba's performance during our exploration.

1) ***Squeeze and Excitation****:* The Squeeze-and-Excitation (SE) block introduces an innovative mechanism to enhance the representational power of convolutional neural networks (CNNs) by explicitly modeling the interdependencies between feature channels [4]. Unlike traditional approaches that focus on spatial correlations, the SE block focuses on channel-wise relationships. The mechanism operates in two main stages: squeeze, which uses global average pooling to encode the global spatial information into a channel descriptor, and excitation, which applies a gating mechanism to adaptively recalibrate the channel responses. This recalibration allows the network to selectively emphasize informative channels while suppressing less relevant ones.

The SE block can be seamlessly integrated into existing architectures with minimal computational overhead. In our case, for Vim, we introduced this block at the end of the

forward pass; more specifically, at the end of the *VisionEncoderMambaBlock* — a core building block in Vim that applies Mamba-based operations for visual feature extraction — just before the residual connection, to recalibrate channel-wise feature responses, as seen in Figure 4

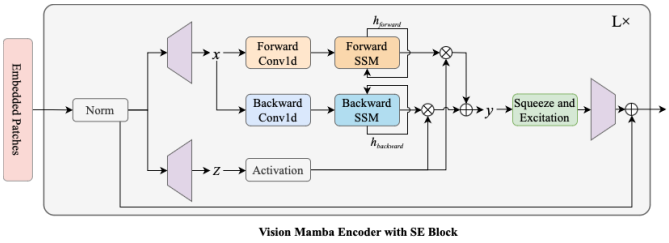

Fig. 4. VisionEncoderMambaBlock with Squeeze and Excitation integrated

The SE architecture, demonstrated using Inception and Residual modules as examples, is illustrated in Figure 5. These examples are provided to help the reader understand the inner workings and design principles of the SE block. Note that the Inception and Residual modules themselves were not directly used in this work.

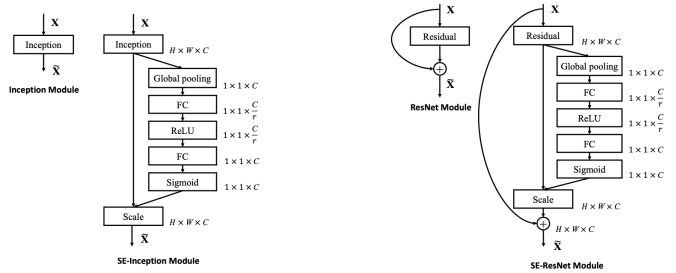

Fig. 5. Demonstration of SE block integration using the Inception and Residual modules as examples. These serve to illustrate the SE block's architecture and functionality.

Its application has demonstrated consistent performance improvements across various vision tasks. Their effectiveness was validated in the ILSVRC 2017 competition [4], where SE networks achieved state-of-the-art results, outperforming previous architectures such as ResNet.

*2) **MobileNetV2**:* MobileNetV2 [5] builds upon its predecessor, MobileNetV1 [21], with a novel architectural innovation: the inverted residual structure with linear bottlenecks, designed to balance computational efficiency with high accuracy. This architecture is particularly optimized for mobile and resource-constrained environments, addressing the need for lightweight yet effective deep learning models. A key component of MobileNetV2 is its use of depthwise separable convolutions, which process high-dimensional feature maps with minimal computational overhead. Additionally, the introduction of linear bottlenecks eliminates non-linear transformations in the narrow layers, preserving critical low-dimensional representations and preventing information loss. By combining these techniques, MobileNetV2 achieves an impressive trade-off between accuracy and computational cost, making it well-suited for tasks such as image classification, object detection, and semantic segmentation on mobile devices and embedded systems.

*3) **Dynamic Pruning with Cosine Scheduler**:* Dynamic pruning is a technique that iteratively reduces the number of active parameters in a neural network during training, focusing computational resources on the most impactful connections. This approach dynamically adjusts the sparsity of the model, allowing it to maintain competitive performance while significantly reducing memory and computational requirements. By progressively pruning less influential weights, dynamic pruning encourages the network to learn robust representations in a more efficient manner.

The integration of a cosine scheduler further enhances this process by controlling the pruning rate throughout the training cycle. Initially, fewer weights are pruned to allow the network to stabilize, and the rate of pruning gradually increases following a cosine decay pattern. This smooth transition ensures that the model retains important features during the early stages of training, while effectively reducing redundancy in later stages.

To preserve critical network components, pruning was weakened in key layers, such as the input *first_layer* and the *output_head*. Specifically, these layers were pruned at 50% of the calculated sparsity level to ensure their integrity while the remaining layers followed the full sparsity schedule. This selective pruning strategy minimizes potential disruptions to the network's core functionality.

*4) **Iterative Magnitude Pruning**:* Identifying the winning ticket is particularly important for heavyweight models such as the examples explored in our experiment while being ideal for resource-constrained environments. IMP [6] is a structured pruning method that works by repeatedly removing a percentage of the network's smallest-magnitude weights—those that contribute least to the model's output, to then retraini the remaining weights to recover performance. This process exploits the idea that some weights in overparameterized networks are redundant and can be safely removed without significant accuracy loss. By iteratively pruning and fine-tuning, IMP identifies a smaller and far more computationally efficient subnetwork that retains the original model's performance, otherwise known as the winning lottery ticket. The iterative structure is designed to dynamically account for sparsity, but we chose to assert the sparsity value on a more aggressive schedule at

$$\text{prune\_intensity} = \text{prune\_intensity} + p \cdot 0.04 \qquad (1)$$

where prune_intensity represents the current sparsity level of the model, p is a scaling factor (e.g., pruning iteration index), and 0.04 is a fixed increment controlling the rate of sparsity increase. This dynamic rule allows for a gradual increase in pruning intensity. This approach is particularly valuable for applications like image classification or object detection on mobile devices, where computational and memory efficiency are critical. IMP and its ability to target a given sparsity for deploying lightweight deep learning models is particularly useful in combatting the intense compute that is required with

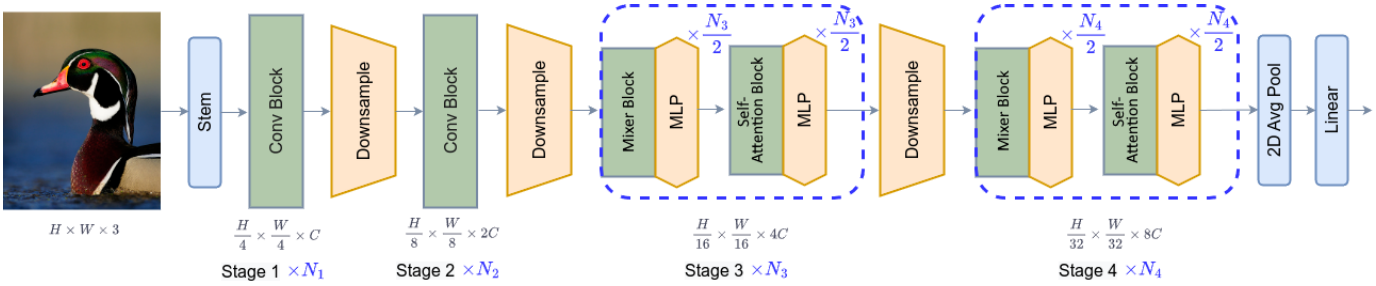

Fig. 6. GMambaVision: Generalized MambaVision incorporating convolutional layers, mixer blocks, and self-attention variants.

even the smaller versions of vision mamba.

*5) **Early-Bird Lottery Ticket Hypothesis**:* The objective of the Early-Bird Lottery Ticket (EBLT) hypothesis is adjacent to Iterative Magnitude Pruning and also a structured pruning method. EBLT [7] leverages a similar principle of the lottery ticket hypothesis but addresses it in a strikingly different way where we aim to uncover the winning ticket at a very early training stage. The hypothesis asserts that the emergent key patterns within the network can be uncovered in one of the first 6 epochs. The process involves training a network and inviting this search through mask distance

$$\textit{Mask Distance} = \|M_t - M_{t+1}\|$$

which quantifies the change in pruning masks across epochs and their role in the importance with respect to the loss landscape. This method aims to stabilize the the mask distance to determine when the winning lottery ticket has been uncovered. In our reconaissance, this method was praised within the bidirectionality of multi-head-attention and thusly was seen as a key method for creating lightweight vision mamba.

*6) **Gradient Pruning**:* Gradient pruning is a pruning method that evaluates the strength of gradients during training to decide which weights or structures (e.g., neurons, filters) should be pruned. The intuition lies within the idea that weights with less intense gradients contribute less to the loss improvement and can be removed without significantly affecting performance. While we did not have access to sparse tensor cores for our implementation (where real training improvements are realized in this niche), the goal was to be exhaustive in the comparison of sparsity driven pruning methods. The method we implemented was

$$\textit{importance} = s * \|g_w\| \quad \textit{(individual weight)}$$

and which then was used to calculate the smallest *k* parameters enabling finer control over the pruning process. This is useful when the raw magnitudes of gradients are small or need to be emphasized while ranking importance .

B. Mamba Vision

We propose and experiment with several modifications to MambaVision to address its current limitations:

1) **Shifted Window Attention**: Drawing inspiration from the Samba model, which enhances context length and perplexity in language modeling through Sliding Window Attention (SWA), we hypothesize that integrating the Computer Vision task analogous Shifted Window Attention into MambaVision will significantly expand its receptive field. This modification is expected to be particularly effective for tasks involving large, high-resolution images, such as medical microscopy and pathological image analysis like the USCF-Cancer Dataset [22].
2) **Incorporation of Relative Attention**: To address generalization challenges associated with smaller datasets such as CIFAR100, we introduce the relative attention mechanism inspired by CoaT [11]. Relative attention combines convolutional inductive biases such as locality and translational equivariance, with the global receptive field and input-independent weighting of self-attention, as formulated in Equation (2). This was also inspired and motivated by the fact that EfficientVMamba [16] and MambaVision [8] have a very similar architecture to CoaTNet, utilising convolutions in the first two stages for fast and efficient feature extraction while maintaining convolutional inductive biases in earlier stages.

$$y_i^{\text{pre}} = \sum_{j \in \mathcal{G}} \frac{\exp\left(x_i^\top x_j + w_{i-j}\right)}{\sum_{k \in \mathcal{G}} \exp\left(x_i^\top x_k + w_{i-k}\right)} x_j \qquad (2)$$

where x_i and x_j are input features, w_{i-j} is an input-independent parameter of static value (used to represent the translationally equivariant depthwise convolution), $\mathcal{G}$ is the set of neighbouring positions, and the softmax weighting captures input-adaptive similarity $x_i^\top x_j$ adjusted by the translational equivariant w_{i-j}

3) **Integration of Registers**: Inspired by [18], we experiment with incorporating register tokens into the Vision Transformer. These additional tokens serve as placeholders for internal computations, addressing artifacts in feature maps commonly observed in supervised and self-supervised Vision Transformers (ViTs). Such artifacts, are found in predominantly high-norm tokens in low-informative background regions, and can be mitigated by introducing these extra tokens, which are discarded after encoding. This was achieved by appending a single column of additional tokens to the end of the feature map derived from the first two convolutional stages.

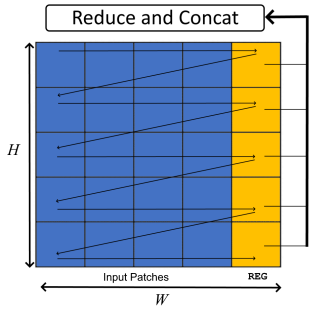

Fig. 7. Registers added to MambaVision's Mixer and Self-Attention Blocks.

Specifically, the feature map in $\mathbb{R}^{C\times H\times W}$ is transformed into a feature map of dimensionality $\mathbb{R}^{C\times H\times W+1}$, using approach as shown in Fig. 7. This modification ensures that valid convolutions can be computed for the downsampling layers, while also adhering to the principle introduced in [10], which demonstrated that VisionMamba prefers evenly inserted registers throughout the input token sequence. Following the approach in [10], given n d-dimensional register vectors, we first apply a linear layer to reduce their dimensionality by a factor of r (here, 2), and then concatenate them into a single vector of dimension $n \times \frac{d}{r}$, which is used as the global representation.

To implement these modifications, we propose **GMambaVision** (Figure 6), a generalized version of MambaVision. GMambaVision combines convolutional layers for efficient feature extraction in the first two stages, followed by a hybrid approach in subsequent stages. Specifically, it employs $\lceil N/2 \rceil$ blocks of LSTM/RNN/SSM-based mixer blocks followed by $\lfloor N/2 \rfloor$ blocks utilizing self-attention variants tailored to computer vision, such as shifted window attention and relative attention.

Furthermore, we also experimented with attention-based pooling and integrating ConvNeXt principles in Mambavision:

*1) **Mamba Vision: Attention-Based Pooling**:* Attention pooling enhances standard pooling by dynamically assigning importance to spatial features [23]. Unlike max pooling or average pooling, attention pooling uses softmax-normalized importance scores to aggregate features, retaining the most critical information. This mechanism is particularly beneficial for tasks requiring high spatial resolution, as it reduces the risk of losing key spatial details during downsampling.

Our implementation of attention pooling involves computing importance scores via a lightweight fully connected network, followed by a weighted sum over spatial regions. This mechanism is integrated within MambaVision's hierarchical structure to emphasize critical features at each stage of the downsampling process.

*2) **Integrating ConvNeXt Principles into MambaVision**:* ConvNeXt [24] principles were incorporated to enhance MambaVision's performance on CIFAR-100. The following modifications were made:

- **Kernel Size Reduction**: Kernel sizes were reduced from 7×7 to 3×3 to better accommodate smaller image resolutions in CIFAR-100.
- **Mixup Augmentation**: Mixup [25], a data augmentation technique, was implemented to improve generalization by creating linear combinations of image-label pairs.
- **Layer Normalization**: Batch normalization was replaced with layer normalization [26] to ensure smoother gradient flow and stability during training.

These adjustments collectively aimed to align MambaVision's architecture with the smaller scale and unique challenges of the CIFAR-100 dataset, improving its ability to capture both local and global features effectively.

IV. Experiments and Results

This section outlines the experimental setup, methodologies, and findings from evaluating Vision Mamba and its extensions. We implemented various techniques to enhance the efficiency and accuracy of Vision Mamba and tested them on standard image datasets. Our focus remained on improving accuracy, reducing computational overhead, and uncovering sparse subnetwork structures.

The experiments were conducted on both NYU's Greene HPC cluster and Google Colab environment using V100 GPUs. CIFAR-10 dataset [27] was used for initial evaluations. This section directly presents the experimental steps, implementation details, and results.

A. Vision Mamba

When experimenting with Vision Mamba, we first focused on finding a concrete set of parameters we could use across our various experiments. This was done with the idea of finding good overall results, and for us to be able to load models saved from our previous experiments without any issues. Thus, after a couple of trials and error, we configured the Vision Mamba model with the following parameters for all of our experiments:

- **dim:** 256 (feature dimension)
- **dt_rank:** 32 (rank of the state-space parameter)
- **dim_inner:** 256 (inner dimension for MLP layers)
- **d_state:** 256 (state size for the SSMs)
- **num_classes:** 10 (output classes for CIFAR-10)
- **image_size:** 32 (input image resolution)
- **patch_size:** 16 (patch size for embedding)
- **channels:** 3 (number of input channels)
- **dropout:** 0.1 (dropout rate for regularization)
- **depth:** 10 (number of VisionEncoderMambaBlocks)

This configuration ensures a balance between model capacity and computational efficiency, making Vision Mamba suitable for experiments on both our small-scale datasets and our limited hardware capabilities. However, it is important to mention that some experiments yielded slightly better results when setting *dim, dim_inner, and d_state* to 96. More on details in our GitHub reposirtory.

Additionally, the training process uses the CIFAR-10 dataset, where images are preprocessed with normalization

to center pixel values around zero and converted to tensors. The training and testing datasets are loaded with mini-batches using DataLoader, with shuffling enabled for training to ensure randomness. Both loaders utilize parallel data loading (num_workers=4) for efficiency, providing preprocessed batches for model training and evaluation.

For experimental comparison, we first trained and evaluated the unmodified Vision Mamba model on the CIFAR-10 dataset. This resulted in an accuracy of **66%**, which serves as our baseline metric.

The following experiments summarize our experiments and findings:

- **Squeeze-and-Excitation (SE):** Integrating SE blocks at the end of the VisionEncoderMambaBlock resulted in accuracy ranging from 61% to 68%, depending on hyperparameter tuning. This might suggest that the performance of SE modules within Vision Mamba is highly sensitive to hyperparameter choices and requires careful tuning for optimal results. Note that for our experiment, the SE reduction factor was set to 4.
- **MobileNetV2 Depthwise Convolutions:** Replacing standard convolutions with depthwise convolutions slightly improved accuracy to 68.3% . This suggests that replacing standard convolutions with MobileNetV2-inspired depthwise convolutions might effectively reduce computational overhead, according to Sandler et al. [5], while slightly improving accuracy.
- **Dynamic Pruning with Cosine Scheduler:** Pruning weights iteratively with a final sparsity of 60% improved accuracy up to 70%. Critical layers, like the input and output layers, were pruned less aggressively (50% less) to maintain stability.
- **Iterative Magnitude Pruning:** Iterative magnitude pruning realized a less robust loss than anticipated in [6] with as low as a 3% loss in accuracy at the benchmark 60% sparsity for CIFAR100. Due to the fact that the loss landscape of our baseline model accuracy was far sharper than in the paper, the higher sparsity levels likely created this accuracy disparagment.
- **Early-Bird Lottery Ticket Hypothesis:** The lottery ticket search was conducted from epochs 2-6 where at each of the epochs, a structured pruning of 50% as applied in the paper was then trained to baseline model realizing a nominal and general uniform accuracy loss of 2% to 2.5%
- **Gradient Pruning:** No constructive results were found in this implementation.

B. Mamba Vision

The following experiments were conducted using the same hyperparameters and training setup as in [28] to ensure consistency and facilitate generalization on smaller datasets like CIFAR100. The primary modifications to the baseline MambaVision include:

1) **Sliding Window Attention (SWA) and Relative Attention Blocks**: In MambaVision, the last two stages contain $\lfloor N/2 \rfloor$ self-attention blocks, operating on 1D embeddings $\in \mathbb{R}^{C \times W}$. These blocks were adapted to work with 2D embeddings $\in \mathbb{R}^{C \times H \times W}$, enabling the use of SWA or relative attention. For relative attention, cached relative positional encodings were utilized to compute multi-headed relative attention using equation 2 for each block. For SWA, the window size was halved (using floor division) for each stage, with window shifts of $\text{window\_width}/2, \text{window\_height}/2$ for every odd block $(2n+1)$. Convolutional downsampling using a 3×3 convolution with a stride of 2 was utilised at the end of each stage, similar to the baseline architecture.
2) **Incorporating Registers into Mamba and Attention Blocks**: The reduction factor r to reduce the dimensionality of register tokens was kept as 1 (no reduction) for MambaVision-T to adhere to the paper [10].

Furthermore, additional experiments were conducted with modified hyperparameters as follows:

- **Mamba Vision: Attention-Based Pooling** The introduction of attention pooling to MambaVision marked a significant shift in the model's ability to retain critical spatial information during downsampling. Starting with the baseline MambaVision model trained on CIFAR-100, which achieved a Top-1 accuracy of 71.71% and a final training loss of 3.20, we replaced all standard pooling layers with attention pooling mechanisms.
  Initially, the model experienced a drop in performance. During the first 10 epochs, training loss increased from 3.15 to 3.75, and Top-1 accuracy showed a slight decline. This behavior was expected as the model adjusted to the new pooling mechanism, which emphasized a different weighting scheme for spatial features. However, by epoch 50, the loss began stabilizing, and accuracy showed consistent improvement, the model achieved a Top-1 accuracy of 74.55% and a final loss of 2.96. This progression highlights the model's ability to leverage the attention pooling mechanism effectively, ultimately surpassing the baseline performance. Attention pooling proved particularly beneficial in retaining critical spatial features, as evidenced by the consistent improvement in Top-5 accuracy, which reached 93.52%.
- **Integrating ConvNeXt Principles into MambaVision:** For ConvNeXt integration, we trained a modified version of MambaVision incorporating reduced kernel sizes, mixup augmentation, and layer normalization. The training configuration included a batch size of 16, a learning rate of $1e{-}3$, and weight decay of $1e{-}2$. This experiment aimed to evaluate the effectiveness of ConvNeXt-inspired adjustments in improving accuracy while maintaining efficiency.

C. Summary of Results

Table I provides a comprehensive summary of the experimental results for the Vision Mamba model. Each experiment is detailed with its corresponding accuracy and key observations. This summary highlights the impact of various

TABLE I
SUMMARY OF EXPERIMENTAL RESULTS ON VISION MAMBA

Experiment	Accuracy (%)	Remarks
Original (Baseline)	**66%**	Unmodified Vision Mamba architecture.
Iterative Magnitude Pruning	**63% (typical)** 68% (few cases)	Accuracy generally deteriorates to 63%, but occasionally reaches 67%. Sensitive to pruning schedules.
Early-Bird Lottery	**63.5% to 64%**	Accuracy loss between 2% and 2.5% observed consistently at 60% sparsity.
Dynamic Pruning	**70%**	Best performance observed at 60% sparsity using cosine decay scheduling.
Squeeze & Excitation	**61% to 68%**	Results varied depending on hyperparameter tuning, showing sensitivity to overfitting.
MobileNetV2	**67.5% to 68.3%**	Slight accuracy improvement while reducing computational costs.

architectural modifications and pruning techniques on model performance, offering a clear comparison between approaches.

TABLE II
TOP-1 AND TOP-5 ACCURACIES FOR MAMBAVISION MODIFICATIONS

Model	Top-1 Accuracy (%)	Top-5 Accuracy (%)
Baseline MambaVision	71.71	92.37
MambaVision + Attention Pooling	**74.55**	**93.52**
MambaVision + ConvNeXt Principles	72.50	93.10

Table II summarizes the experimental results obtained with the baseline MambaVision model and the introduced modifications. Attention pooling demonstrated the most significant improvement in Top-1 accuracy, increasing from 71.71% to 74.55%, while applying ConvNeXt principles generated accuracy that did not deviate much from the base MambaVision model.

TABLE III
INITIAL AND FINAL LOSSES FOR MAMBAVISION MODIFICATIONS

Model	Initial Loss	Final Loss
Baseline MambaVision	3.15	3.20
MambaVision + Attention Pooling	3.75	2.96
MambaVision + ConvNeXt Principles	3.10	3.05

Table III further discusses the behavior of the attention pooling mechanism. The initial decline in performance, as indicated by a higher training loss during the first 10 epochs, can be attributed to the model adjusting to the new pooling strategy, which diverges from the uniform feature weighting of standard pooling. Over subsequent epochs, the loss stabilized, and accuracy improved consistently, showcasing the model's ability to adapt to attention-based mechanisms. This suggests that attention pooling requires a longer training horizon to fully leverage its benefits.

Additionally, the ConvNeXt-inspired modifications provided improved regularization and training stability. While the performance gains were not as pronounced as with attention pooling, the reduced kernel sizes and mixup augmentation effectively mitigated overfitting, particularly important for smaller datasets like CIFAR-100.

Comparing these results with prior attempts at integrating self-attention mechanisms, as explored in [8] and [16], it becomes evident that MambaVision's hierarchical design already incorporates efficient feature downsampling to a degree that complements attention pooling. This aligns with findings from CoAtNet [11], where relative attention provided improved generalization on smaller datasets, benefiting from inductive biases. Future work could explore the impact of attention pooling and ConvNeXt principles on larger datasets like ImageNet or high-resolution medical imagery, as the gains on CIFAR-100 indicate potential scalability.

TABLE IV
CIFAR100 ACCURACY USING MAMBAVISION-T BASELINE

Self-Attention Block	Mixer Block	Registers	Top-1 Accuracy
Vanilla	Mamba	-	61.26
Swin [29]	Mamba	-	61.25
CoaT [11]	**Mamba**	-	**64.77**
Vanilla	Mamba	✓	61.92

TABLE V
CIFAR100 ACCURACY QUANTISATION (TRANSFER LEARNING)

Quantisation Method	Top-1 Accuracy
FP16	89.93
FP32	91.21
AMP	91.20

Table IV summarizes the results of experiments conducted on CIFAR100. The integration of CoaT's relative attention outperformed both the Swin block and the vanilla self-attention

mechanism. This improvement is likely due to the added inductive bias of relative attention, enhancing generalization on smaller datasets. Notably, shifting to Shifted Window Attention did not yield significant improvements, possibly because the convolutional feature extraction stages effectively downsample the spatial dimensions, making $O(n^2)$ complexity self-attention manageable within the model. Perhaps if the dataset had been of higher resolution, such as certain medical datasets like USCF-Cancer Dataset [22] or COCO [30], it might have been more appropriate to utilize the Swin Transformer's shifted window attention block. However, the CoaT relative attention block performed well, likely due to the inductive bias introduced by relative attention, which enhances generalization capabilities [11]. This characteristic was particularly beneficial when trained on a relatively small dataset like CIFAR100. Table V demonstrates that reducing precision to FP16 results in a drop in model performance when quantized. Nevertheless, Automatic Mixed Precision (AMP) proved to be an effective quantization strategy for transfer learning on the CIFAR100 downstream task. This suggests that significant training speedups can be achieved simply by employing AMP for transfer learning with MambaVision.

V. Conclusion

This study explored enhancements to state-space models for computer vision tasks, with a focus on Vision Mamba and MambaVision. Our experiments demonstrated that Vim exhibits robustness in its ability to adapt to pruning techniques, with an average accuracy variation of approximately 3% across experiments. Notably, the integration of dynamic pruning yielded the highest accuracy improvement, achieving up to 70%, while maintaining computational efficiency. Other methods, such as MobileNetV2-inspired depthwise convolutions and Squeeze-and-Excitation blocks, highlighted trade-offs between computational cost and accuracy gains (i.e., resulted in lower accuracy), indicating potential for further fine-tuning.

These findings underscore the versatility of Vim as a foundation for visual representation learning, particularly in resource-constrained environments.

Building on these results, our work with MambaVision demonstrates the potential for extending SSM-based models to high-resolution image tasks, leveraging sliding window attention, registers, and Relative Attention. We found Relative attention to work notably well for generalisation when trained on smaller datasets like CIFAR100, and thus, it could also address the generalisation gap in medical datasets where data scale is low.

Future research will aim to refine these methods further and evaluate their applicability on larger datasets and more complex vision tasks, particularly in medical imaging and other domains requiring high-resolution processing.

A. Limitations

While our experiments provided valuable insights into Vim and MambaVision, several limitations were encountered during the study. Firstly, Vim was trained with varying numbers of epochs and batch sizes across different experiments, which may have introduced inconsistencies in the training setups and affected direct comparisons of results. However, for the final run, we standardized the settings as much as possible to ensure consistency. In addition, we used different CPUs and computational resources, which introduces variability in training times and efficiency. These factors underscore the need for more standardized experimental setups to ensure consistent and reliable benchmarking.

Another limitation lies in the scale of the datasets used. While CIFAR-10 [27] and CIFAR-100 [31] provided a sufficient baseline for initial experiments, it is relatively small compared to the large-scale datasets, that are commonly used for training state-space models. This smaller dataset size may limit the generalizability of our findings to more complex vision tasks. Future efforts could focus on evaluating our methods on additional small-scale datasets, such as ImageNette [32], and/or larger and more complex datasets, including ImageNet-1K [33], and high-resolution medical pathology datasets, such as the UCSF cancer datasets [22]. These extensions will help validate the scalability and effectiveness of our designs across diverse and high-resolution domains.

While we experimented with advanced quantization schemes such as SmoothQuant [34] and QuaRot [35], their implementation posed significant challenges within the limited timeframe. These methods either failed to train effectively with MambaVision or did not perform well. Future research will explore these and other robust quantization techniques to enhance the speed and efficiency of structured state-space models (SSM) for vision tasks.

Future work should address these limitations by incorporating standardized training configurations, leveraging more powerful computational resources, and evaluating the models on larger datasets to validate their scalability and generalizability.

References

[1] J. T. H. Smith, A. Warrington, and S. W. Linderman, "Simplified state space layers for sequence modeling," 2023. [Online]. Available: https://arxiv.org/abs/2208.04933

[2] A. Gu, K. Goel, and C. Ré, "Efficiently modeling long sequences with structured state spaces," 2022. [Online]. Available: https://arxiv.org/abs/2111.00396

[3] L. Zhu, B. Liao, Q. Zhang, X. Wang, W. Liu, and X. Wang, "Vision mamba: Efficient visual representation learning with bidirectional state space model," 2024. [Online]. Available: https://arxiv.org/abs/2401.09417

[4] J. Hu, L. Shen, S. Albanie, G. Sun, and E. Wu, "Squeeze-and-excitation networks," 2019. [Online]. Available: https://arxiv.org/abs/1709.01507

[5] M. Sandler, A. Howard, M. Zhu, A. Zhmoginov, and L.-C. Chen, "Mobilenetv2: Inverted residuals and linear bottlenecks," 2019. [Online]. Available: https://arxiv.org/abs/1801.04381

[6] M. Paul, F. Chen, B. W. Larsen, J. Frankle, S. Ganguli, and G. K. Dziugaite, "Unmasking the lottery ticket hypothesis: What's encoded in a winning ticket's mask?" 2022. [Online]. Available: https://arxiv.org/abs/2210.03044

[7] H. You, C. Li, P. Xu, Y. Fu, Y. Wang, X. Chen, R. G. Baraniuk, Z. Wang, and Y. Lin, "Drawing early-bird tickets: Towards more efficient training of deep networks," 2022. [Online]. Available: https://arxiv.org/abs/1909.11957

[8] A. Hatamizadeh and J. Kautz, "Mambavision: A hybrid mamba-transformer vision backbone," 2024. [Online]. Available: https://arxiv.org/abs/2407.08083

[9] 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," Jun. 2021, arXiv:2010.11929 [cs]. [Online]. Available: http://arxiv.org/abs/2010.11929

[10] F. Wang, J. Wang, S. Ren, G. Wei, J. Mei, W. Shao, Y. Zhou, A. Yuille, and C. Xie, "Mamba-r: Vision mamba also needs registers," 2024. [Online]. Available: https://arxiv.org/abs/2405.14858

[11] Z. Dai, H. Liu, Q. V. Le, and M. Tan, "Coatnet: Marrying convolution and attention for all data sizes," 2021. [Online]. Available: https://arxiv.org/abs/2106.04803

[12] Y. Tay, M. Dehghani, S. Abnar, Y. Shen, D. Bahri, P. Pham, J. Rao, L. Yang, S. Ruder, and D. Metzler, "Long range arena: A benchmark for efficient transformers," 2020. [Online]. Available: https://arxiv.org/abs/2011.04006

[13] J. T. H. Smith, A. Warrington, and S. W. Linderman, "Simplified state space layers for sequence modeling," 2023. [Online]. Available: https://arxiv.org/abs/2208.04933

[14] E. Nguyen, K. Goel, A. Gu, G. W. Downs, P. Shah, T. Dao, S. A. Baccus, and C. Ré, "S4nd: Modeling images and videos as multidimensional signals using state spaces," 2022. [Online]. Available: https://arxiv.org/abs/2210.06583

[15] L. Ren, Y. Liu, Y. Lu, Y. Shen, C. Liang, and W. Chen, "Samba: Simple hybrid state space models for efficient unlimited context language modeling," 2024. [Online]. Available: https://arxiv.org/abs/2406.07522

[16] X. Pei, T. Huang, and C. Xu, "Efficientvmamba: Atrous selective scan for light weight visual mamba," 2024. [Online]. Available: https://arxiv.org/abs/2403.09977

[17] Y. Liu, Y. Tian, Y. Zhao, H. Yu, L. Xie, Y. Wang, Q. Ye, and Y. Liu, "Vmamba: Visual state space model," 2024. [Online]. Available: https://arxiv.org/abs/2401.10166

[18] T. Darcet, M. Oquab, J. Mairal, and P. Bojanowski, "Vision transformers need registers," 2024. [Online]. Available: https://arxiv.org/abs/2309.16588

[19] A. Dosovitskiy, L. Beyer, A. Kolesnikov, D. Weissenborn, X. Zhai, T. Unterthiner, M. Dehghani, M. Minderer, G. Heigold, S. Gelly *et al.*, "An image is worth 16x16 words: Transformers for image recognition at scale," *arXiv preprint arXiv:2010.11929*, 2020.

[20] Z. Liu, Y. Lin, Y. Cao, H. Hu, Y. Wei, Z. Zhang, S. Lin, and B. Guo, "Swin transformer: Hierarchical vision transformer using shifted windows," in *Proceedings of the IEEE/CVF international conference on computer vision*, 2021, pp. 10 012–10 022.

[21] A. G. Howard, M. Zhu, B. Chen, D. Kalenichenko, W. Wang, T. Weyand, M. Andreetto, and H. Adam, "Mobilenets: Efficient convolutional neural networks for mobile vision applications," 2017. [Online]. Available: https://arxiv.org/abs/1704.04861

[22] E. Calabrese, J. Villanueva-Meyer, J. Rudie, A. Rauschecker, U. Baid, S. Bakas, S. Cha, J. Mongan, and C. Hess, "The university of california san francisco preoperative diffuse glioma mri (ucsf-pdgm) (version 4) [dataset]," The Cancer Imaging Archive, 2022. [Online]. Available: https://doi.org/10.7937/tcia.bdgf-8v37

[23] J. Hu, L. Shen, and G. Sun, "Squeeze-and-excitation networks," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 42, no. 8, pp. 2011–2023, 2020.

[24] Z. Liu, H. Mao, C.-Y. Wu, C. Feichtenhofer, T. Darrell, and S. Xie, "Convnext: Revisiting convolutional networks for image recognition at scale," *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 4817–4827, 2022.

[25] H. Zhang, M. Cisse, Y. N. Dauphin, and D. Lopez-Paz, "mixup: Beyond empirical risk minimization," in *Proceedings of the International Conference on Learning Representations (ICLR)*, 2018.

[26] J. L. Ba, J. R. Kiros, and G. E. Hinton, "Layer normalization," 2016. [Online]. Available: https://arxiv.org/abs/1607.06450

[27] A. Krizhevsky, V. Nair, and G. Hinton, "Cifar-10 (canadian institute for advanced research)." [Online]. Available: http://www.cs.toronto.edu/~kriz/cifar.html

[28] J. H. Tan, "Pre-training of lightweight vision transformers on small datasets with minimally scaled images," 2024. [Online]. Available: https://arxiv.org/abs/2402.03752

[29] Z. Liu, Y. Lin, Y. Cao, H. Hu, Y. Wei, Z. Zhang, S. Lin, and B. Guo, "Swin transformer: Hierarchical vision transformer using shifted windows," 2021. [Online]. Available: https://arxiv.org/abs/2103.14030

[30] T.-Y. Lin, M. Maire, S. Belongie, L. Bourdev, R. Girshick, J. Hays, P. Perona, D. Ramanan, C. L. Zitnick, and P. Dollár, "Microsoft coco: Common objects in context," 2015. [Online]. Available: https://arxiv.org/abs/1405.0312

[31] A. Krizhevsky, V. Nair, and G. Hinton, "Cifar-100 (canadian institute for advanced research)." [Online]. Available: http://www.cs.toronto.edu/~kriz/cifar.html

[32] ""fastai/imagenette: a smaller subset of 10 easily classified classes from imagenet," 2024. [Online]. Available: https://github.com/fastai/imagenette

[33] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei, "Imagenet: A large-scale hierarchical image database," in *2009 IEEE Conference on Computer Vision and Pattern Recognition*, 2009, pp. 248–255.

[34] G. Xiao, J. Lin, M. Seznec, H. Wu, J. Demouth, and S. Han, "Smoothquant: Accurate and efficient post-training quantization for large language models," 2024. [Online]. Available: https://arxiv.org/abs/2211.10438

[35] S. Ashkboos, A. Mohtashami, M. L. Croci, B. Li, P. Cameron, M. Jaggi, D. Alistarh, T. Hoefler, and J. Hensman, "Quarot: Outlier-free 4-bit inference in rotated llms," 2024. [Online]. Available: https://arxiv.org/abs/2404.00456