MambaTree: Tree Topology is All You Need in State Space Model

Yicheng Xiao[1†*], **Lin Song**[2,3✉*],
Shaoli Huang[3], **Jiangshan Wang**[1], **Siyu Song**[4], **Yixiao Ge**[2,3], **Xiu Li**[1✉], **Ying Shan**[2,3]

[1]Tsinghua Shenzhen International Graduate School, Tsinghua University
[2]ARC Lab, Tencent PCG [3]Tencent AI Lab [4]South China Normal University
xiaoyc23@mails.tsinghua.edu.cn ronnysong@tencent.com

Abstract

The state space models, employing recursively propagated features, demonstrate strong representation capabilities comparable to Transformer models and superior efficiency. However, constrained by the inherent geometric constraints of sequences, it still falls short in modeling long-range dependencies. To address this issue, we propose the MambaTree network, which first dynamically generates a tree topology based on spatial relationships and input features. Then, feature propagation is performed based on this graph, thereby breaking the original sequence constraints to achieve stronger representation capabilities. Additionally, we introduce a linear complexity dynamic programming algorithm to enhance long-range interactions without increasing computational cost. MambaTree is a versatile multimodal framework that can be applied to both visual and textual tasks. Extensive experiments demonstrate that our method significantly outperforms existing structured state space models on image classification, object detection and segmentation. Besides, by fine-tuning large language models, our approach achieves consistent improvements in multiple textual tasks at minor training cost. Code is available at https://github.com/EasonXiao-888/GrootVL.

1 Introduction

Mainstream fundamental models are primarily based on CNN [30, 62, 44, 32, 13] and Transformer architectures [15, 43, 42, 59, 14], which dominate in visual and language tasks. However, the small receptive field of CNNs and the high complexity of Transformers make it challenging to strike a good balance between effectiveness and efficiency. The state space models (SSMs) [22, 24, 52] attempt to disrupt this impasse, which model sequences in a recurrent form. Different from the previous recurrent neural networks [31, 7], these approaches draw inspiration from control systems, leveraging structural parameter initialization to attain stable optimization and superior computing performance. Nevertheless, it remains susceptible to the intrinsic flaw shared by recurrent neural networks, $i.e.$, a deficiency in capturing long-range dependencies.

Recently, an improved selection mechanism known as Mamba [19] is proposed to mitigate the challenges of SSMs. This approach introduces weight modulation during the propagation process, which substantially enlarges the effective receptive field and achieves impressive performance in NLP tasks. Besides, numerous studies aim to extend Mamba into computer vision, by employing various pre-defined strategies to map 2D image features into 1D sequences. ViM [77] and VMamba [41] utilize a multi-directional raster-scanning strategy, while LocalMamba [34] further confines its

*Equal contribution. † Work done during an internship at Tencent. ✉ Corresponding author.

38th Conference on Neural Information Processing Systems (NeurIPS 2024).

https://doi.org/10.52202/079017-2398

propagation range within a local window. They have successfully adapted Mamba to image inputs. Nevertheless, as shown in Fig. 1(a), both raster-scanning and local-scanning strategies introduce spatial discontinuities between adjacent pixels, and feature transformations in Mamba rely on the feature relationships, thereby impeding the effective information flow in a sequence. Additionally, PlainMamba [69] introduces a continuous scanning strategy, aiming to alleviate this issue by simply adjusting the propagation direction at discontinuous positions. However, all these methods rely on fixed propagation trajectories, which ignore the inherent spatial structure and cannot dynamically adjust the topology based on input. Therefore, this paper endeavors to explore a new perspective: *introducing an input-aware topological network for feature propagation in state space models*.

To achieve it, we develop a tree state space model and propose a new framework, termed MambaTree, which adaptively generates a tree topology based on the input feature and then performs feature propagation on it. Specifically, two sub-networks, MambaTreeV and MambaTreeL, are designed for visual and language tasks respectively, which are illustrated in Fig. 1(b) and Fig. 1(d). For visual tasks, motivated by [71, 54], we first utilize the dissimilarity between adjacent features to construct a minimum spanning tree on a four-connected planner graph. This process can adaptively encode the spatial and semantic information into a tree graph [71, 54]. Then, we iteratively traverse each pixel, considering it as the root vertex, and aggregate the features of other pixels using the state transition function of Mamba. Intuitively, this operation requires two levels of traversal across the entire pixel set, resulting in an unacceptable quadratic complexity relative to the number of pixels. However, given that the tree graph is acyclic, we propose a dynamic programming algorithm to achieve linear complexity propagation. With such an input-aware tree topology, our approach enables more effective long-range interactions while maintaining consistent linear complexity with Mamba. Furthermore, our method can also be applied to language tasks by constructing a tree topology based on the dissimilarity between token features, which overcomes the geometrical constraints of the text sequence. Using a similar aggregation process as MambaTreeV, MambaTreeL can significantly enhance the language representation of a pre-trained Large Language Model [19].

We conduct extensive experiments to validate the effectiveness of MambaTreeV on multiple visual benchmarks, $i.e.$ image classification on ImageNet [12], object detection and instance segmentation on MSCOCO [39] as well as semantic segmentation on ADE20K [75]. Results show that our method notably outperforms existing SSM-based methods for all benchmarks and achieves competitive performance with CNN and Transformer-based approaches. Moreover, with LoRA finetuning [33], MambaTreeL demonstrates consistent improvements for a pre-trained large language model at minor training cost.

2 Related Work

2.1 Conventional Vision Foundation Models

The evolution of deep neural networks has been a significant catalyst in machine vision perception. CNN-based models [30, 51, 35, 25, 61, 72, 38, 55, 73] firstly emerge as pivotal landmarks, with ResNet [30] notably standing out for its inventive residual connection module, garnering widespread adoption across diverse domains of visual recognition. Furthermore, more efficient convolution operations are formulated, such as depth-wise convolutions introduced by MobileNet [32], paving the way for lightweight models. Additionally, deformable convolution [10] has been proposed to enhance the receptive field. Subsequently, ViT [15] has significantly improved the vision recognition paradigm. It reformulates the architecture design and training mechanism by combining transformer architecture in natural language processing, aiming to improve computational efficiency and broaden the scope of applications. After research discourse is centred on hierarchical ViTs [43, 42, 11, 63, 14, 56, 5] which design networks by decreasing feature resolution across the backbone gradually. Furthermore, recent research built on CNN serves to re-emphasize the capabilities of convolutional networks. For example, InternImage [62] presents a large model based on deformable CNN, while UniRepLKNet [13] exhibits significant performance through large kernel convolution.

2.2 Explorations about State Space Models

State space models (SSMs) have emerged as a novel class of models within the deep learning paradigm, showing significant potential for sequence transforming [23, 22, 52]. These methods have attracted significant attention due to their linear scalability with sequence length. The early method,

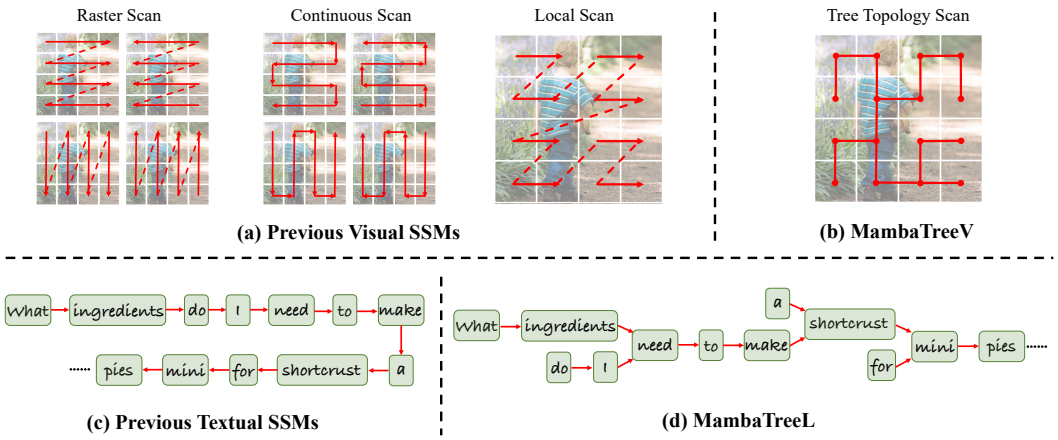

Figure 1: **Comparison of different propagation strategies for multi-modal tasks.** For visual tasks, the previous strategies (a) are based on fixed patterns, while our method can adaptively generate the propagation topology according to input features. For textual tasks, compared to previous methods (c), our approach (d) can break the inherent constraints of text sequences, facilitating the effective transmission of long-range information.

LSSL [23], draws inspiration from continuous state space models in control systems and attempts to address the long-range dependency problem through a combination with HIPPO [20] initialization. S4 [22] proposes to normalize the parameters into a diagonal matrix, prompting a subsequent series of research on structured SSMs [24, 21, 26, 19]. Recently, the Selective State Space Model [19], known as Mamba, strikes a balance between effectiveness and efficiency through the design of an input-dependent parameter initialization strategy, which has emerged as a formidable competitor to both transformer and CNN structures. In addition to showcasing superior outcomes in sequence modeling, Mamba has been seamlessly incorporated into the visual domain [77, 41, 34, 69, 68]. These studies often rely on handcrafted fixed scanning mechanisms to mitigate the execution bias of the selective state space model on 2D non-causal images. However, such simplistic approaches cannot effectively capture spatial relationships in an input-dependent paradigm. To address this limitation, we propose an effective framework MambaTree in this work to enhance long-range modeling for both vision and language tasks by introducing an input-aware tree-based topological structure.

3 Method

In this section, we first revisit the selective state space model [19] and then elaborate on our input-aware topology scanning algorithm for state space modeling. With this superior algorithm, we develop a tree SSM and propose a novel framework called MambaTree, which consists of two sub-networks: MambaTreeV for visual tasks and MambaTreeL for fine-tuning a pre-trained language model [19].

3.1 Revisiting Selective State Space Model

State Space Models (SSMs) are commonly regarded as continuous linear time-invariant systems [64] that map input stimulation $x(t) \in \mathbb{R}^{1\times D}$ to output signal $y(t) \in \mathbb{R}^{1\times D}$ through a state vector $h(t) \in \mathbb{R}^{1\times N}$, where t, D and N indicate the time step, channel number of the signal and state size, respectively. These models can be formulated as the following linear ordinary differential equations:

$$h'(t) = \mathbf{A}h(t) + \mathbf{B}x(t), \quad y(t) = \mathbf{C}h(t) + \mathbf{D}x(t), \tag{1}$$

where $\mathbf{A} \in \mathbb{R}^{N\times N}$, $\mathbf{B} \in \mathbb{R}^{N\times D}$, $\mathbf{C} \in \mathbb{R}^{N\times D}$ and feedthrough coefficient $\mathbf{D} \in \mathbb{R}^{D}$.

Discretization. Although SSM serves as a powerful tool in systems and control engineering, its time-continuous nature poses challenges for integration into deep learning architectures. To alleviate this issue, most methods utilize the zero-order hold rule [19] to discretize the continuous system described by Eq. (1) and convert continuous variables ($\mathbf{A}$, $\mathbf{B}$, $\mathbf{C}$, $\mathbf{D}$) into corresponding discrete

https://doi.org/10.52202/079017-2398

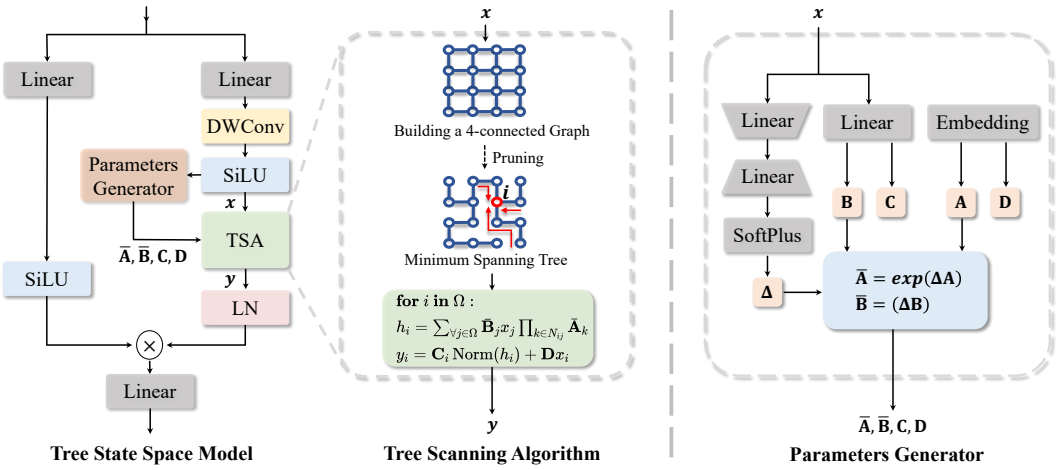

Figure 2: **Illustration of Tree State Space Model.** With an image feature map x, we perform Tree Scanning Algorithm (TSA) to construct a 4-connected graph with edge weights measured by dissimilarity between pixels. Then, we obtain an MST with vertices set Ω through a pruning algorithm and perform the state transition for each vertex in this topology (detailed in Sec. 3.2). Red arrows describe the propagation source of vertex i.

parameters ($\bar{\mathbf{A}}$, $\bar{\mathbf{B}}$, $\bar{\mathbf{C}}$, $\bar{\mathbf{D}}$) over the specified sampling time-scale $\Delta \in \mathbb{R}^D$:

$$\bar{\mathbf{A}} = e^{\Delta \mathbf{A}}, \quad \bar{\mathbf{B}} = \left(e^{\Delta \mathbf{A}} - I\right) \mathbf{A}^{-1} \mathbf{B}, \quad \bar{\mathbf{C}} = \mathbf{C}, \quad \bar{\mathbf{D}} = \mathbf{D} \tag{2}$$

In addition, many improved methods [41, 19] use an approximation of $\bar{\mathbf{B}}$ based on the first-order Taylor Series:

$$\bar{\mathbf{B}} = \left(e^{\Delta \mathbf{A}} - I\right) \mathbf{A}^{-1} \mathbf{B} \approx (\Delta \mathbf{A})(\Delta \mathbf{A})^{-1} \Delta \mathbf{B} = \Delta \mathbf{B} \tag{3}$$

Selective Mechanism . Previous SSMs store information through finite states and inherent time-invariance, which limits their effectiveness. Therefore, Mamba [19] introduces a dynamic mechanism to selectively filter out input into a sequential state. Specifically, it utilizes Linear Projection to calculate the parameters $\{\mathbf{B}_i\}_{i=1}^L$, $\{\mathbf{C}_i\}_{i=1}^L$ and $\{\mathbf{\Delta}_i\}_{i=1}^L$ from the input sequence $\{x_i\}_{i=1}^L$ with $x_i \in \mathbb{R}^{1 \times D}$ directly to improve the context-aware ability. Then the output sequence $\{y_i\}_{i=1}^L$ can be computed with those input-adaptive discretized parameters as follows:

$$h_i = \bar{\mathbf{A}}_i h_{i-1} + \bar{\mathbf{B}}_i x_i, \quad y_i = \mathbf{C}_i h_i + \mathbf{D} x_i \tag{4}$$

3.2 Tree State Space Model

Mamba [19] has showcased remarkable performance in modeling the dependencies of consecutive words in a sequence. However, its applicability in long-context tasks, especially visual modeling, still poses certain challenges. For visual tasks, many methods attempt to address this problem by employing fixed scanning strategies, such as multi-directional raster scan [41, 77], local scan [34], and continuous scan [69]. However, these handcrafted scanning methods fail to effectively preserve the 2D structural information of images.

Following the design in Mamba [19], we construct a transform block as a tree state space model, which is presented in Fig. 2. The only difference between our block and Mamba lies in the replacement of the structured state space block with the proposed tree scanning algorithm. In the tree scanning algorithm, we generate a tree topology and then propagate the state of each vertex along the topological path to obtain strong feature representations. In addition, our algorithm can effectively enhance language representations by incorporating such a tree topology during text processing, which overcomes the geometrical constraints of text sequences. In the following, we elaborate on the proposed tree scanning algorithm and its applications for multi-modal tasks.

Tree Scanning Algorithm. Given an input feature $X = \{x_i\}_{i=1}^L$ where L is the sequence length (or the number of input pixels), we construct an undirected m-connected graph $G = (V, E)$ for the

https://doi.org/10.52202/079017-2398

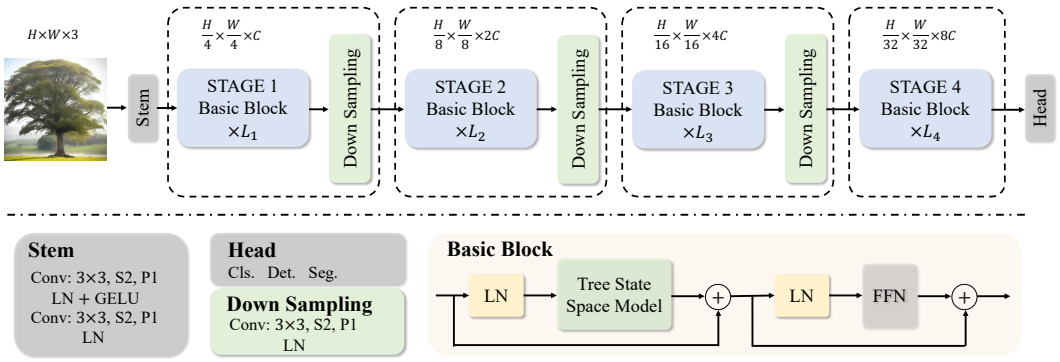

Figure 3: **Overview of MambaTreeV.** LN means LayerNorm and FFN is a feed-forward network in the basic block. S2 and P1 denote stride of 2 and padding size of 1 in convolution, respectively.

feature. m is a hyper-parameter that indicates the number of adjacent tokens. Following [71, 54], we set $m = 4$ for visual tasks, meaning each pixel is connected to its four neighboring pixels. For language tasks, we set $m = 3$ by default, meaning each token is connected to the previous three tokens. In addition, the vertices V represent the pixel (or token) embeddings, and the E indicates the edges of the graph. The edge weight is calculated by the feature dissimilarity between adjacent vertices. Besides, the metric of dissimilarity uses cosine distance by default, and the comparison with other metrics refers to Table 5.

We use the Contractive Boruvka algorithm [2] to prune the edges with significant dissimilarity, which generates a minimum spanning tree (MST) $\mathcal{G}_T$ whose sum of dissimilarity weights is minimum out of all spanning trees. In the propagation process, we iteratively traverse each vertex, treating it as the root, and aggregate the features of the remaining vertices. Intuitively, applying state propagation within such a geometric configuration makes its preferential interactions among vertices with small spatial and feature distances. Following the Mamba, we employ the data-dependent transition matrix for state propagation. For a vertex k, we denote the transition matrix with its parent as $\bar{\mathbf{A}}_k$. Furthermore, following the Eq. (4), the state aggregation process for the i-th vertex can be formulated as:

$$h_i = \sum_{\forall j \in \Omega} S(E_{ij}) \bar{\mathbf{B}}_j x_j, \quad S(E_{ij}) = \prod_{k \in N_{ij}} \bar{\mathbf{A}}_k, \tag{5}$$

where Ω denotes the index set of all vertices in the tree. $S(E_{ij})$ represents the path weight of hyperedge E_{ij} traced from j-th vertex to i-th vertex in the tree $\mathcal{G}_T$, and N_{ij} indicates the index set of all vertices on this hyperedge. For visual tasks, we iterate over each vertex, treating it as the root of the spanning tree $\mathcal{G}_T$, and aggregate the states from the other vertices, thereby obtaining the transformed states $\{h_i\}_{i=1}^{L}$. For textual tasks, because of the causal prediction manner in large language models, we only take the last token as root and aggregate from other tokens. To achieve end-to-end training, we derive the derivative of the output hidden state h_i to the input variables $\bar{\mathbf{A}}_k$, $\bar{\mathbf{B}}_j$ and x_j as follows:

$$\frac{\partial h_i}{\partial x_j} = S\left(E_{ij}\right) \bar{\mathbf{B}}_j, \quad \frac{\partial h_i}{\partial \bar{\mathbf{B}}_j} = S\left(E_{ij}\right) x_j \tag{6}$$

$$\frac{\partial h_i}{\partial \bar{\mathbf{A}}_k} = \sum_{\forall j \in C_k^i} \bar{\mathbf{B}}_j x_j S(E_{kj}) S(E_{in}), \tag{7}$$

where C_k^i indicates the children of vertex k in tree $\mathcal{G}_T$ whose root is the vertex i, and n denotes the parent of vertex k in Eq. (7). Finally, the output feature Y can be formulated as:

$$Y = \mathbf{C} \odot Norm(H) + \mathbf{D} \odot X, \tag{8}$$

where Y, H and X indicate the stack of $\{y_i\}_{i=1}^{L}$, $\{h_i\}_{i=1}^{L}$ and $\{x\}_{i=1}^{L}$ respectively. $\odot$ denotes the element-wise multiplication.

Efficient Implementation for Multi-Modality. For visual tasks, the tree scanning algorithm requires two levels of traversal across the entire pixel set, resulting in an unacceptable quadratic complexity relative to the number of pixels $\mathcal{O}(L^2)$. To alleviate this issue, we utilize a dynamic

https://doi.org/10.52202/079017-2398

Algorithm 1 Vision Tree Scanning

Input: Input feature $\{x_i\}_{i=1}^{L}$; Input matrix $\{\bar{\mathbf{B}}_i\}_{i=1}^{L}$; State matrix $\{\bar{\mathbf{A}}_i\}_{i=1}^{L}$; Gradient of loss to hidden states $\{\frac{\partial Loss}{\partial h_i}\}_{i=1}^{L}$; Minimum Spanning Tree $\mathcal{G}_T$.

Traverse Path: $Root, \ldots, Leaf \leftarrow BFS(\mathcal{G}_T)$ ▷ *Breadth-first topological order of* $\mathcal{G}_T$

Forward:

Initialization: $\{\xi_i\}_{i=1}^{L} \leftarrow \{x_i\}_{i=1}^{L}$

for $i \leftarrow Leaf$ to $Root$ **do**

$\xi_i = \bar{\mathbf{B}}_i x_i + \sum_{\forall j \in \{t|\text{Par}(t)=i\}} \xi_j \bar{\mathbf{A}}_j$

end for

for $i \leftarrow Root$ to $Leaf$ **do**

if i is $Root$ **then**

$h_i = \xi_i$

else

$h_i = \bar{\mathbf{A}}_i(h_{\text{Par}(i)} - \bar{\mathbf{A}}_i \xi_i) + \xi_i = (1 - \bar{\mathbf{A}}_i^2)\xi_i + \bar{\mathbf{A}}_i h_{\text{Par}(i)}$

end if

end for

Backward:

Initialization: $\{\eta_i\}_{i=1}^{L} \leftarrow \{\frac{\partial Loss}{\partial h_i}\}_{i=1}^{L}$

for $i \leftarrow Leaf$ to $Root$ **do**

$\eta_i = \bar{\mathbf{B}}_i \frac{\partial Loss}{\partial h_i} + \sum_{\forall j \in \{t|\text{Par}(t)=i\}} \eta_j \bar{\mathbf{A}}_j$

end for

for $i \leftarrow Root$ to $Leaf$ **do**

if i is $Root$ **then**

$\frac{\partial Loss}{\partial x_i} = \eta_i \bar{\mathbf{B}}_i\,, \quad \frac{\partial Loss}{\partial \bar{\mathbf{B}}_i} = \eta_i x_i\,, \quad \frac{\partial Loss}{\partial \bar{\mathbf{A}}_i} = 0$

else

$\frac{\partial Loss}{\partial x_i} = (1 - \bar{\mathbf{A}}_i^2)\eta_i \bar{\mathbf{B}}_i + \bar{\mathbf{A}}_i \frac{\partial Loss}{\partial x_{\text{Par}(i)}} \bar{\mathbf{B}}_i\,, \quad \frac{\partial Loss}{\partial \bar{\mathbf{B}}_i} = (1 - \bar{\mathbf{A}}_i^2)\eta_i x_i + \bar{\mathbf{A}}_i \frac{\partial Loss}{\partial \bar{\mathbf{B}}_{\text{Par}(i)}} x_i$

$\frac{\partial Loss}{\partial \bar{\mathbf{A}}_i} = \eta_i * (h_i - \bar{\mathbf{A}}_i \xi_i) + \xi_i * (\frac{\partial Loss}{\partial x_i} - \bar{\mathbf{A}}_i \eta_i) = \eta_i h_i + \xi_i \frac{\partial Loss}{\partial x_i} - 2\eta_i \xi_i \bar{\mathbf{A}}_i$

end if

end for

Output: Hidden states $\{h_i\}_{i=1}^{L}$; Grad. of loss to input feature $\{\frac{\partial Loss}{\partial x_i}\}_{i=1}^{L}$; Grad. of loss to input matrix $\{\frac{\partial Loss}{\partial \bar{\mathbf{B}}_i}\}_{i=1}^{L}$; Grad. of loss to state matrix $\{\frac{\partial Loss}{\partial \bar{\mathbf{A}}_i}\}_{i=1}^{L}$.

programming procedure to accelerate the inference and training processes as elaborated in Algorithm 1, which results in linear complexity $\mathcal{O}(L)$. For textual tasks, we perform a unidirectional aggregation approach (shown in Algorithm 2 of Appendix B) in adherence to the causal nature of language. Moreover, we provide the back-propagation process for both Vision Tree Scanning and Language Tree Scanning processes, whose detailed proofs refer to Appendix C.

3.3 Application for Vision and Language

MambaTreeV Given an image with a shape of $H \times W \times 3$, our goal is to obtain high-quality visual features for downstream tasks. To this end, we propose an effective vision architecture MambaTreeV which consists of a stem module, several basic blocks and downsampling layers to generate hierarchical representations illustrated in Fig. 3. Overall, our MambaTreeV comprises four stages similar to previous general vision backbones [44, 43, 62, 41]. We integrate the stem module before the first stage to decrease the resolution of the input image signal by a factor of 4, resulting in a feature map with a shape of $\frac{H}{4} \times \frac{W}{4} \times C$. It includes two convolutions, two Layer Normalization (LN) layers and one GELU activation function. The kernel size for both convolutions is 3 with a stride of 2 and padding of 1. Similarly, a downsampling layer consists of a 3×3 convolution with a stride of 2 and padding of 1 and an LN layer. Positioned between two stages, it serves to downsample the input feature map by a factor of 2. Motivated by [62, 41], we devise a residual block with skip connections to integrate our fundamental Tree State Space Model in Sec. 3.2. In detail, we first normalize the input features with LN layer. Spatial priors and long-range dependencies are then obtained through our tree scanning algorithm with residual connections established alongside the input features. Finally, a feedforward neural network is utilized to project the normalized features to output signals as shown

Method	Type	#Param.	#FLOPs	Top-1 Acc.
Deit-S [59]	T	22M	4.6G	79.9
Swin-T [43]	T	28M	4.6G	81.3
CoAtNet-0 [11]	T	25M	4.0G	81.6
SG-Former-S [50]	T	23M	4.8G	83.2
ConvNeXt-T [44]	C	29M	4.5G	82.1
SLaK-T [40]	C	30M	5.0G	82.5
UniRepLKNet-T [13]	C	31M	4.9G	83.2
InternImage-T [62]	C	30M	5.0G	83.5
ViM-S [77]	S	26M	5.1G	80.5
LocalViM-S [34]	S	28M	4.8G	81.2
PlainMamba-L2 [69]	S	25M	8.1G	81.6
Mamba-2D-S [37]	S	24M	-	81.7
S4ND-ConvNeXt-T [48]	S	30M	-	82.2
VMamba-T [41]	S	31M	4.9G	82.5
LocalVMamba-T [34]	S	26M	5.7G	82.7
MambaTreeV-T (Ours)	S	30M	4.8G	**83.4**
Swin-S [43]	T	50M	8.7G	83.0
CoAtNet-1 [11]	T	42M	8.0G	83.3
ConvNeXt-S [44]	C	50M	8.7G	83.1
SLaK-S [40]	C	55M	9.8G	83.8
UniRepLKNet-S [13]	C	56M	9.1G	83.9
InternImage-S [62]	C	50M	8.0G	84.2
HyenaViT-B [17]	S	88M	-	78.5
S4ND-ViT-B [48]	S	89M	-	80.4
PlainMamba-L3 [69]	S	50M	14.4G	82.3
VMamba-S [41]	S	50M	8.7G	83.6
LocalVMamba-S [34]	S	50M	11.4G	83.7
MambaTreeV-S (Ours)	S	51M	8.5G	**84.2**
Deit-B [59]	T	86M	55.4G	83.1
Swin-B [43]	T	88M	15.4G	83.5
CoAtNet-2 [11]	T	75M	16.0G	84.1
ConvNeXt-B [44]	C	89M	15.4G	83.8
SLaK-B [40]	C	95M	17.0G	84.0
Mamba-2D-B [37]	S	92M	-	83.0
VMamba-B [41]	S	89M	15.4G	83.9
MambaTreeV-B (Ours)	S	91M	15.1G	**84.8**

Table 1: **Image classification performance on the ImageNet-1K validation set.** T, C and S indicate the model type of Transformer, CNN and SSM, respectively. All models take a scale of 224^2 as input.

in Fig. 3. Based on the above origin components, we develop our MambaTreeV in three scales, $i.e.$, MambaTreeV-Tiny, MambaTreeV-Small and MambaTreeV-Base.

MambaTreeL Recurrent neural networks rely on fixed memory to preserve past information, which poses limitations when handling long contexts where relevant words are distant from the current moment. While Mamba [19] employs a selection mechanism to enhance context awareness, its fixed memory size cannot expand over time, resulting in restricted state space. Therefore, the ability to extrapolate decreases during scrolling as the prompt extends. To mitigate this issue, we propose an effective fine-tuning paradigm. Specifically, the tree-based topology branch is built upon one-way scrolling with a scaling factor, enabling state transitions within such a structure. This arrangement facilitates the preferential interaction of semantically related tokens. It is noteworthy that this paradigm does not introduce any additional training parameters. Instead, it utilizes pretrained state transformation parameters to conduct semantic aggregation by incorporating topological structures. Experimental results demonstrate the effectiveness of our approach.

4 Experiments

We conduct extensive experiments to evaluate the effectiveness of MambaTreeV and compare it with advanced CNN-based, Transformer-based, and SSM-based models covering various downstream tasks, including image classification, object detection and semantic segmentation. Furthermore, we validate the capability of MambaTreeL in the field of natural language understanding.

4.1 Image Classification

Settings. We assess the classification performance of MambaTreeV on the ImageNet-1k dataset [12]. Following previous practices [43, 44, 62, 41], all MambaTreeV models are trained for 300 epochs from scratch using AdamW optimizer with a warm-up strategy of 20 epochs. During training, we utilize a Cosine Scheduler with an initial learning rate of 1×10^{-3} and weight decay of 0.05. In addition, the exponential moving average (EMA) is also applied.

Results. The comparison results summarized in Table 1 show MambaTreeV leading all SSM-based models and competitive with advanced CNNs and Transformers across tiny, small, and base scales. Specifically, MambaTreeV-T achieves 83.4% Top-1 Acc. boosting ViM-S by 2.9%, LocalVim-S by 2.2%, PlainMamba-L2 by 1.8% and VMamba-T by 0.9% with similar FLOPs. Additionally, it surpasses ConvNeXt-T by 1.3% and Swin-T by 2.2%, demonstrating the effectiveness of our method.

https://doi.org/10.52202/079017-2398

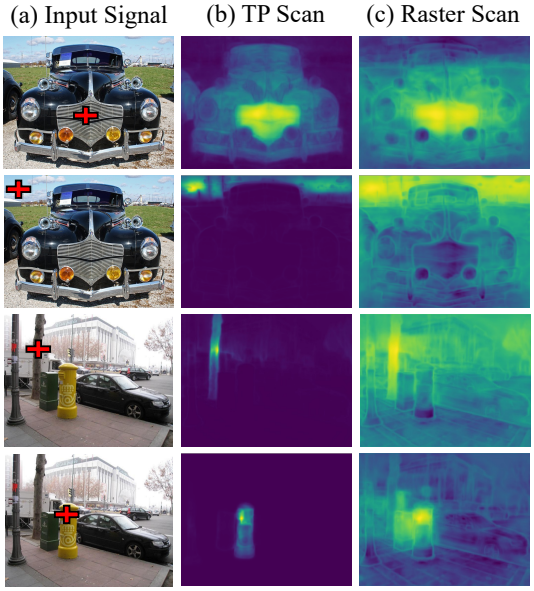

Figure 4: **Visualization of affinity maps in the specific position.** The Location is marked by the red cross in each input (a). TP is our tree topology scanning algorithm (b), which captures more detailed structural information and has a larger receptive field compared to raster scanning (c).

Method	Type	#FLOPs	mIoU SS	mIoU MS
Swin-T [43]	T	945G	44.5	45.8
ConvNeXt-T [44]	C	939G	46.0	46.7
SLaK-T [40]	C	936G	47.6	-
InternImage-T [62]	C	944G	47.9	48.1
UniRepLKNet-T [13]	C	946G	48.6	49.1
ViM-S [77]	S	-	44.9	-
LocalViM-S [34]	S	297G	46.4	47.5
PlainMamba-L2 [69]	S	285G	46.8	-
VMamba-T [41]	S	964G	47.3	48.3
LocalVMamba-T [41]	S	970G	47.9	49.1
MambaTreeV-T (Ours)	S	941G	**48.5**	**49.4**
Swin-S [43]	T	1038G	47.6	49.5
ConvNeXt-S [44]	C	1027G	48.7	49.6
SLaK-S [40]	C	1028G	49.4	-
InternImage-S [62]	C	1017G	50.1	50.9
UniRepLKNet-S [13]	C	1036G	50.5	51.0
PlainMamba-L3 [69]	S	419G	49.1	-
VMamba-S [41]	S	1081G	49.5	50.5
LocalVMamba-S [34]	S	1095G	50.0	51.0
MambaTreeV-S (Ours)	S	1019G	**50.7**	**51.7**

Table 2: **Semantic segmentation performance on ADE20K val set.** The crop size is all set to 512^2. SS and MS denote single-scale and multi-scale testing, respectively.

4.2 Object Detection

Settings. We verify the detection performance of MambaTreeV on the MSCOCO 2017 dataset [39] with MMDetection library [3]. We follow previous works [41, 62, 43, 34, 53, 55, 74, 70, 6] to validate object detection and instance segmentation tasks with Mask-RCNN [29]. Specifically, We adopt the AdamW optimizer with a learning rate of 1×10^{-4} and batch size of 16 to optimize the model built upon our pre-trained classification backbones on ImageNet-1K. The training schedules include $1\times$ (12 epochs) and $3\times$ (36 epochs) with multi-scale data augmentation.

Results. As depicted in Table 8 (in Appendix A.), our method outperforms existing methods on most evaluation metrics, especially for instance segmentation. Under $1\times$ schedule, MambaTreeV-T achieves 47.0 in box mAP (AP^b), which is 1.1 points higher than ViM-S and 0.5 points higher than VMamba-T. It is worth noting that MambaTreeV-T outperforms ViM-S by 1.7 points with $1\times$ schedule and LocalVMamba-T by 0.4 points with $3\times$ schedule in mask mAP (AP^m). Moreover, the best AP^b 50.1 and AP^m 44.6 are obtained by MambaTreeV-S in $3\times$ schedule with multi-scale training.

4.3 Semantic Segmentation

Settings. To evaluate the semantic segmentation performance of our MambaTreeV series, we train our models with UperNet [65] initialized by pre-trained classification weights on ADE20K[75] for 160k iterations, following common practices without additional augmentations for fair comparison.

Results. Our method performs exceptionally well on segmentation tasks shown in Table 2. MambaTreeV-T yields a clear improvement of $+3.6$ in single-scale mIoU compared to ViM-S and $+1.9$ in multi-scale mIoU compared to LocalViM-S. Furthermore, MambaTreeV-S boosts InternImage-S by 0.6 and 0.8 in single-scale and multi-scale respectively. We consider the preservation of intricate structural details through tree topology scanning to be particularly advantageous for segmentation tasks that require pixel-level perception.

Method	PIQA ↑	Arc-E ↑	SST ↑	WG ↑	L-ppl ↓	Race ↑	BQA ↑	Average Acc. ↑
Mamba [19]	64.5	48.0	65.6	51.8	16.1	27.4	16.8	45.7
+ LoRA [33]	64.7	48.3	65.1	**52.2**	17.7	28.6	17.8	46.1
+ MambaTreeL (Ours)	**65.0**	**49.8**	**69.5**	51.1	**15.9**	**28.9**	**19.2**	**47.2**

Table 3: **Evaluation on language model benchmarks.** Arc-E, WG, L-ppl and BQA indicate Arc-easy [8], WinoGrande, LAMBADA [49] and Openbookqa [47] benchmark, respectively.

Scanning Strategy	Acc
Raster Scan	82.6
Cross Scan	83.1
Tree Topology Scan	**83.4**

Table 4: **Effectiveness of our algorithm.**

Distance Metric	Acc.
$Manhattan$	82.9
$Euclidean$	83.2
$Cosine$	**83.4**

Table 5: **Impact of different distance Metrics.**

Root Setting	Acc.
First vertex	82.9
Last vertex	83.0
All vertices	**83.4**

Table 6: **Superiority of traversing all vertices.**

4.4 Language Understanding

We regard Mamba [19] with 130M parameters as the base model. To verify the effectiveness of our MambaTreeL in nature language understanding, we first fine-tune pre-trained Mamba via LoRA [33] and MambaTreeL under the same setting with the Alpaca data [58], which contains 52000 instruction tuning data for supervised fine-tuning. Then we utilize popular language benchmarks provided in the open-sourced lm-evaluation-harness project [18] for evaluation, including PIQA [1], AI2-ARC [8], SST [60], WinoGrande, LAMBADA [49], Race [36] and Openbookqa [47]. The results in Table 3 demonstrate that our MambaTreeL provides a benefit of $+1.1\%$ in average Acc. compared to LoRA. Since the short prompt length of WinoGrande dataset, the performance degrades with a marginal gap.

4.5 Ablation Study & Qualitative Results

In this section, we conduct analysis experiments on ImageNet-1K dataset and present some visual results to illustrate the effectiveness of our algorithm.

Scanning Strategy. We conduct a head-to-head comparison of different scanning strategies, as shown in Table 4. The tree topology scanning outperforms previous strategies by 0.8% and 0.3%, highlighting the superiority of our algorithm in vision recognition.

Distance Metric. Before generating a minimum spanning tree from a connected graph, it is important to measure the edge weights between vertices. Therefore, we validate several distance metrics as illustrated in Table 5. The results indicate that $Cosine$ distance most effectively represents the relationship between vertices, performing 0.5% better than $Manhattan$ and 0.2% better than $Euclidean$.

Root Setting. We traverse all vertices, treating each as a root, and perform state transitions along the topological path from the other vertices toward the root. This traversal ensures that each vertex captures long-range dependencies. To verify the effectiveness of this operation, we consider only the first and last vertices as the root in Table 6. The results show reductions of 0.5% and 0.4%, respectively.

Inference speed comparison. As shown in Table 7, we report the inference throughputs of our method on an Nvidia V100 GPU. MambaTreeV-T* refers to each stage sharing the same tree topology structure, which enhances efficiency without compromising accuracy. To achieve better practical inference speed, we also introduce a cuda implementation optimized for GPUs. Compared with other counterparts, our approach exhibits superior effectiveness and faster inference speed.

Qualitative Results. To better illustrate the superiority of our scanning strategy, we visualize the affinity maps of different positions marked by the red cross in each input image. For example, we

https://doi.org/10.52202/079017-2398

Method (224x224)	Throughput (img/s)	GPU Memory	FLOPs	#Params.	Acc. (Top1.)
PlainMamba-L2 [69]	363	4204M	8.1G	25M	81.6
VMamba-T [41]	374	8646M	4.9G	31M	82.5
LocalVMamba-T [34]	311	11298M	5.7G	26M	82.7
MambaTreeV-T(one root)	283	6012M	4.8G	30M	83.0
MambaTreeV-T	281	6471M	4.8G	30M	83.4
MambaTreeV-T*	**392**	**4800M**	**4.8G**	30M	**83.4**

Table 7: **Runtime comparison on an Nvidia V100 GPU during inference.**

set the anchor point in the upper left corner of the sky as shown in the second row of in Fig. 4(a). Our method can easily identify white houses, flagpoles, and the sky, which raster scanning fails to achieve. This demonstrates the capability of our algorithm to preserve detailed structural information. More comparisons can be seen in Fig. 6 (in Appendix D.)

5 Conclusion & Limitations

In this paper, we propose a tree state space model to perform feature propagation on an input-aware topology. Besides, we introduce a linear complexity dynamic programming algorithm to enhance long-range interactions without increasing computational cost. With the proposed techniques, we establish the general multi-modal networks to break the original sequence constraints and achieve stronger representation capabilities. Extensive experiments demonstrate the effectiveness of our method in both visual and language tasks. The limitation of our method is that the tree structure is not a common paradigm, and it needs to be specifically optimized according to the hardware device.

Acknowledgments and Disclosure of Funding

This work was supported by the STI 2030-Major Projects under Grant 2021ZD0201404.

References

[1] Bisk, Y., Zellers, R., Gao, J., Choi, Y., et al.: Piqa: Reasoning about physical commonsense in natural language. In: AAAI. pp. 7432–7439 (2020) 9

[2] Borůvka, O.: O jistém problému minimálním (1926) 5

[3] Chen, K., Wang, J., Pang, J., Cao, Y., Xiong, Y., Li, X., Sun, S., Feng, W., Liu, Z., Xu, J., et al.: Mmdetection: Open mmlab detection toolbox and benchmark. arXiv preprint arXiv:1906.07155 (2019) 8

[4] Chen, Z., Duan, Y., Wang, W., He, J., Lu, T., Dai, J., Qiao, Y.: Vision transformer adapter for dense predictions. arXiv preprint arXiv:2205.08534 (2022) 16

[5] Cheng, C., Song, L., Xue, R., Wang, H., Sun, H., Ge, Y., Shan, Y.: Meta-adapter: An online few-shot learner for vision-language model. arXiv preprint arXiv:2311.03774 (2023) 2

[6] Cheng, T., Song, L., Ge, Y., Liu, W., Wang, X., Shan, Y.: Yolo-world: Real-time open-vocabulary object detection. arXiv preprint arXiv:2401.17270 (2024) 8

[7] Chung, J., Gulcehre, C., Cho, K., Bengio, Y.: Empirical evaluation of gated recurrent neural networks on sequence modeling. arXiv preprint arXiv:1412.3555 (2014) 1

[8] Clark, P., Cowhey, I., Etzioni, O., Khot, T., Sabharwal, A., Schoenick, C., Tafjord, O.: Think you have solved question answering? try arc, the ai2 reasoning challenge. arXiv preprint arXiv:1803.05457 (2018) 9

[9] Cubuk, E.D., Zoph, B., Mane, D., Vasudevan, V., Le, Q.V.: Autoaugment: Learning augmentation strategies from data. In: CVPR. pp. 113–123 (2019) 15

[10] Dai, J., Qi, H., Xiong, Y., Li, Y., Zhang, G., Hu, H., Wei, Y.: Deformable convolutional networks. In: ICCV. pp. 764–773 (2017) 2

[11] Dai, Z., Liu, H., Le, Q.V., Tan, M.: Coatnet: Marrying convolution and attention for all data sizes. NeurIPS **34**, 3965–3977 (2021) 2, 7

[12] Deng, J., Dong, W., Socher, R., Li, L.J., Li, K., Fei-Fei, L.: Imagenet: A large-scale hierarchical image database. In: CVPR. pp. 248–255. Ieee (2009) 2, 7

[13] Ding, X., Zhang, Y., Ge, Y., Zhao, S., Song, L., Yue, X., Shan, Y.: Unireplknet: A universal perception large-kernel convnet for audio, video, point cloud, time-series and image recognition. CVPR (2023) 1, 2, 7, 8

[14] Dong, X., Bao, J., Chen, D., Zhang, W., Yu, N., Yuan, L., Chen, D., Guo, B.: Cswin transformer: A general vision transformer backbone with cross-shaped windows. In: CVPR. pp. 12124–12134 (2022) 1, 2, 16

[15] Dosovitskiy, A., Beyer, L., Kolesnikov, A., Weissenborn, D., Zhai, X., Unterthiner, T., Dehghani, M., Minderer, M., Heigold, G., Gelly, S., Uszkoreit, J., Houlsby, N.: An image is worth 16x16 words: Transformers for image recognition at scale. In: ICLR (2021) 1, 2

[16] Fang, C., He, C., Xiao, F., Zhang, Y., Tang, L., Zhang, Y., Li, K., Li, X.: Real-world image dehazing with coherence-based label generator and cooperative unfolding network. arXiv preprint arXiv:2406.07966 (2024) 15

[17] Fu, D., Arora, S., Grogan, J., Johnson, I., Eyuboglu, E.S., Thomas, A., Spector, B., Poli, M., Rudra, A., Ré, C.: Monarch mixer: A simple sub-quadratic gemm-based architecture. NeurIPS **36** (2023) 7

[18] Gao, L., Tow, J., Abbasi, B., Biderman, S., Black, S., DiPofi, A., Foster, C., Golding, L., Hsu, J., Le Noac'h, A., Li, H., McDonell, K., Muennighoff, N., Ociepa, C., Phang, J., Reynolds, L., Schoelkopf, H., Skowron, A., Sutawika, L., Tang, E., Thite, A., Wang, B., Wang, K., Zou, A.: A framework for few-shot language model evaluation (12 2023) 9

[19] Gu, A., Dao, T.: Mamba: Linear-time sequence modeling with selective state spaces. arXiv preprint arXiv:2312.00752 (2023) 1, 2, 3, 4, 7, 9

[20] Gu, A., Dao, T., Ermon, S., Rudra, A., Ré, C.: Hippo: Recurrent memory with optimal polynomial projections. NeurIPS **33**, 1474–1487 (2020) 3

[21] Gu, A., Goel, K., Gupta, A., Ré, C.: On the parameterization and initialization of diagonal state space models. NeurIPS **35**, 35971–35983 (2022) 3

[22] Gu, A., Goel, K., Ré, C.: Efficiently modeling long sequences with structured state spaces. In: ICLR (2022) 1, 2, 3

[23] Gu, A., Johnson, I., Goel, K., Saab, K., Dao, T., Rudra, A., Ré, C.: Combining recurrent, convolutional, and continuous-time models with linear state space layers. NeurIPS **34**, 572–585 (2021) 2, 3

[24] Gupta, A., Gu, A., Berant, J.: Diagonal state spaces are as effective as structured state spaces. NeurIPS **35**, 22982–22994 (2022) 1, 3

[25] Han, K., Wang, Y., Xu, C., Guo, J., Xu, C., Wu, E., Tian, Q.: Ghostnets on heterogeneous devices via cheap operations. IJCV **130**(4), 1050–1069 (2022) 2

[26] Hasani, R., Lechner, M., Wang, T.H., Chahine, M., Amini, A., Rus, D.: Liquid structural state-space models. arXiv preprint arXiv:2209.12951 (2022) 3

[27] He, C., Li, K., Zhang, Y., Xu, G., Tang, L., Zhang, Y., Guo, Z., Li, X.: Weakly-supervised concealed object segmentation with sam-based pseudo labeling and multi-scale feature grouping. Advances in Neural Information Processing Systems **36** (2024) 15

[28] He, C., Shen, Y., Fang, C., Xiao, F., Tang, L., Zhang, Y., Zuo, W., Guo, Z., Li, X.: Diffusion models in low-level vision: A survey. arXiv preprint arXiv:2406.11138 (2024) 15

 https://doi.org/10.52202/079017-2398

[29] He, K., Gkioxari, G., Dollár, P., Girshick, R.: Mask r-cnn. In: ICCV. pp. 2961–2969 (2017) 8

[30] He, K., Zhang, X., Ren, S., Sun, J.: Deep residual learning for image recognition. In: CVPR. pp. 770–778 (2016) 1, 2

[31] Hochreiter, S., Schmidhuber, J.: Long short-term memory. Neural computation **9**(8), 1735–1780 (1997) 1

[32] Howard, A.G., Zhu, M., Chen, B., Kalenichenko, D., Wang, W., Weyand, T., Andreetto, M., Adam, H.: Mobilenets: Efficient convolutional neural networks for mobile vision applications. arXiv preprint arXiv:1704.04861 (2017) 1, 2

[33] Hu, E.J., Shen, Y., Wallis, P., Allen-Zhu, Z., Li, Y., Wang, S., Wang, L., Chen, W.: Lora: Low-rank adaptation of large language models. In: ICLR (2022) 2, 9

[34] Huang, T., Pei, X., You, S., Wang, F., Qian, C., Xu, C.: Localmamba: Visual state space model with windowed selective scan. arXiv preprint arXiv:2403.09338 (2024) 1, 3, 4, 7, 8, 10, 15, 16

[35] Krizhevsky, A., Sutskever, I., Hinton, G.E.: Imagenet classification with deep convolutional neural networks. NeurIPS **25** (2012) 2

[36] Lai, G., Xie, Q., Liu, H., Yang, Y., Hovy, E.H.: RACE: large-scale reading comprehension dataset from examinations. In: EMNLP. pp. 785–794. Association for Computational Linguistics (2017) 9

[37] Li, S., Singh, H., Grover, A.: Mamba-nd: Selective state space modeling for multi-dimensional data. arXiv preprint arXiv:2402.05892 (2024) 7

[38] Li, Y., Song, L., Chen, Y., Li, Z., Zhang, X., Wang, X., Sun, J.: Learning dynamic routing for semantic segmentation. In: CVPR (2020) 2

[39] Lin, T.Y., Maire, M., Belongie, S., Hays, J., Perona, P., Ramanan, D., Dollár, P., Zitnick, C.L.: Microsoft coco: Common objects in context. In: ECCV. pp. 740–755. Springer (2014) 2, 8

[40] Liu, S., Chen, T., Chen, X., Chen, X., Xiao, Q., Wu, B., Kärkkäinen, T., Pechenizkiy, M., Mocanu, D., Wang, Z.: More convnets in the 2020s: Scaling up kernels beyond 51x51 using sparsity. arXiv preprint arXiv:2207.03620 (2022) 7, 8

[41] Liu, Y., Tian, Y., Zhao, Y., Yu, H., Xie, L., Wang, Y., Ye, Q., Liu, Y.: Vmamba: Visual state space model. arXiv preprint arXiv:2401.10166 (2024) 1, 3, 4, 6, 7, 8, 10, 15, 16

[42] Liu, Z., Hu, H., Lin, Y., Yao, Z., Xie, Z., Wei, Y., Ning, J., Cao, Y., Zhang, Z., Dong, L., et al.: Swin transformer v2: Scaling up capacity and resolution. In: CVPR. pp. 12009–12019 (2022) 1, 2

[43] Liu, Z., Lin, Y., Cao, Y., Hu, H., Wei, Y., Zhang, Z., Lin, S., Guo, B.: Swin transformer: Hierarchical vision transformer using shifted windows. In: ICCV. pp. 10012–10022 (2021) 1, 2, 6, 7, 8, 15, 16

[44] Liu, Z., Mao, H., Wu, C.Y., Feichtenhofer, C., Darrell, T., Xie, S.: A convnet for the 2020s. In: CVPR. pp. 11976–11986 (2022) 1, 6, 7, 8, 16

[45] Loshchilov, I., Hutter, F.: Decoupled weight decay regularization. arXiv preprint arXiv:1711.05101 (2017) 15

[46] Luo, Z., Xiao, Y., Liu, Y., Li, S., Wang, Y., Tang, Y., Li, X., Yang, Y.: Soc: Semantic-assisted object cluster for referring video object segmentation. Advances in Neural Information Processing Systems **36** (2024) 15

[47] Mihaylov, T., Clark, P., Khot, T., Sabharwal, A.: Can a suit of armor conduct electricity? A new dataset for open book question answering. In: EMNLP. pp. 2381–2391. Association for Computational Linguistics (2018) 9

[48] Nguyen, E., Goel, K., Gu, A., Downs, G.W., Shah, P., Dao, T., Baccus, S.A., Ré, C.: S4nd: Modeling images and videos as multidimensional signals using state spaces. arXiv preprint arXiv:2210.06583 (2022) 7

[49] Radford, A., Wu, J., Child, R., Luan, D., Amodei, D., Sutskever, I., et al.: Language models are unsupervised multitask learners. OpenAI blog **1**(8), 9 (2019) 9, 19

[50] Ren, S., Yang, X., Liu, S., Wang, X.: Sg-former: Self-guided transformer with evolving token reallocation. In: ICCV. pp. 6003–6014 (2023) 7

[51] Simonyan, K., Zisserman, A.: Very deep convolutional networks for large-scale image recognition. In: Bengio, Y., LeCun, Y. (eds.) ICLR (2015) 2

[52] Smith, J.T., Warrington, A., Linderman, S.W.: Simplified state space layers for sequence modeling. arXiv preprint arXiv:2208.04933 (2022) 1, 2

[53] Song, L., Li, Y., Jiang, Z., Li, Z., Sun, H., Sun, J., Zheng, N.: Fine-grained dynamic head for object detection. NIPS (2020) 8

[54] Song, L., Li, Y., Li, Z., Yu, G., Sun, H., Sun, J., Zheng, N.: Learnable tree filter for structure-preserving feature transform. NeurIPS **32** (2019) 2, 5

[55] Song, L., Zhang, S., Yu, G., Sun, H.: Tacnet: Transition-aware context network for spatio-temporal action detection. In: CVPR (2019) 2, 8

[56] Song, L., Zhang, S., Liu, S., Li, Z., He, X., Sun, H., Sun, J., Zheng, N.: Dynamic grained encoder for vision transformers. NIPS (2021) 2

[57] Tang, L., Li, K., He, C., Zhang, Y., Li, X.: Source-free domain adaptive fundus image segmentation with class-balanced mean teacher. In: International Conference on Medical Image Computing and Computer-Assisted Intervention. pp. 684–694. Springer (2023) 15

[58] Taori, R., Gulrajani, I., Zhang, T., Dubois, Y., Li, X., Guestrin, C., Liang, P., Hashimoto, T.B.: Stanford alpaca: An instruction-following llama model (2023) 9

[59] Touvron, H., Cord, M., Douze, M., Massa, F., Sablayrolles, A., Jégou, H.: Training data-efficient image transformers & distillation through attention. In: ICML. pp. 10347–10357. PMLR (2021) 1, 7

[60] Wang, A., Singh, A., Michael, J., Hill, F., Levy, O., Bowman, S.R.: GLUE: A multi-task benchmark and analysis platform for natural language understanding. In: ICLR (2019) 9

[61] Wang, J., Song, L., Li, Z., Sun, H., Sun, J., Zheng, N.: End-to-end object detection with fully convolutional network. In: CVPR (2021) 2

[62] Wang, W., Dai, J., Chen, Z., Huang, Z., Li, Z., Zhu, X., Hu, X., Lu, T., Lu, L., Li, H., et al.: Internimage: Exploring large-scale vision foundation models with deformable convolutions. In: CVPR. pp. 14408–14419 (2023) 1, 2, 6, 7, 8, 15, 16

[63] Wang, W., Xie, E., Li, X., Fan, D.P., Song, K., Liang, D., Lu, T., Luo, P., Shao, L.: Pyramid vision transformer: A versatile backbone for dense prediction without convolutions. In: ICCV. pp. 568–578 (2021) 2

[64] Williams, R.L., Lawrence, D.A., et al.: Linear state-space control systems. John Wiley & Sons (2007) 3

[65] Xiao, T., Liu, Y., Zhou, B., Jiang, Y., Sun, J.: Unified perceptual parsing for scene understanding. In: ECCV. pp. 418–434 (2018) 8

[66] Xiao, Y., Luo, Z., Liu, Y., Ma, Y., Bian, H., Ji, Y., Yang, Y., Li, X.: Bridging the gap: A unified video comprehension framework for moment retrieval and highlight detection. CVPR (2024) 15

[67] Xiao, Y., Ma, Y., Li, S., Zhou, H., Liao, R., Li, X.: Semanticac: Semantics-assisted framework for audio classification. In: ICASSP 2023-2023 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP). pp. 1–5. IEEE (2023) 15

 https://doi.org/10.52202/079017-2398

[68] Xu, Z., Lin, Y., Han, H., Yang, S., Li, R., Zhang, Y., Li, X.: Mambatalk: Efficient holistic gesture synthesis with selective state space models. arXiv preprint arXiv:2403.09471 (2024) 3

[69] Yang, C., Chen, Z., Espinosa, M., Ericsson, L., Wang, Z., Liu, J., Crowley, E.J.: Plainmamba: Improving non-hierarchical mamba in visual recognition. arXiv preprint arXiv:2403.17695 (2024) 2, 3, 4, 7, 8, 10

[70] Yang, J., Song, L., Liu, S., Li, Z., Li, X., Sun, H., Sun, J., Zheng, N.: Dbq-ssd: Dynamic ball query for efficient 3d object detection. arXiv preprint arXiv:2207.10909 (2022) 8

[71] Yang, Q.: Stereo matching using tree filtering. IEEE TPAMI **37**(4), 834–846 (2014) 2, 5

[72] Yang, R., Song, L., Ge, Y., Li, X.: Boxsnake: Polygonal instance segmentation with box supervision. In: Proceedings of the IEEE/CVF International Conference on Computer Vision (2023) 2

[73] Zhang, S., Song, L., Gao, C., Sang, N.: Glnet: Global local network for weakly supervised action localization. IEEE Transactions on Multimedia **22**(10), 2610–2622 (2019) 2

[74] Zhang, S., Song, L., Liu, S., Ge, Z., Li, Z., He, X., Sun, J.: Workshop on autonomous driving at cvpr 2021: Technical report for streaming perception challenge. arXiv preprint arXiv:2108.04230 (2021) 8

[75] Zhou, B., Zhao, H., Puig, X., Fidler, S., Barriuso, A., Torralba, A.: Scene parsing through ade20k dataset. In: CVPR. pp. 633–641 (2017) 2, 8

[76] Zhou, H., Yang, R., Zhang, Y., Duan, H., Huang, Y., Hu, R., Li, X., Zheng, Y.: Unihead: unifying multi-perception for detection heads. TNNLS (2023) 15

[77] Zhu, L., Liao, B., Zhang, Q., Wang, X., Liu, W., Wang, X.: Vision mamba: Efficient visual representation learning with bidirectional state space model. arXiv preprint arXiv:2401.09417 (2024) 1, 3, 4, 7, 8, 16

Appendix

A Detailed Training Settings and Results

A.1 Image Classification.

We follow the previous works [62, 41, 43] to conduct the experiments. The models are trained with thirty-two 32GB V100 GPUs by default. We set betas and momentum of the AdamW [45, 76, 66] optimizer with $(0.9, 0.999)$ and 0.9, respectively. During training, we utilize a Cosine Scheduler with an initial learning rate of 1×10^{-3} and weight decay of 0.05. We adopt the common training data augmentation strategies following [34, 62], including AutoAugment [9] with $rand\text{-}m9\text{-}mstd0.5\text{-}inc1$. A MixUp strategy with a ratio of 0.8 is also adopted in each batch. Horizontal flip and Random resized crop strategy are both used in the process of training.

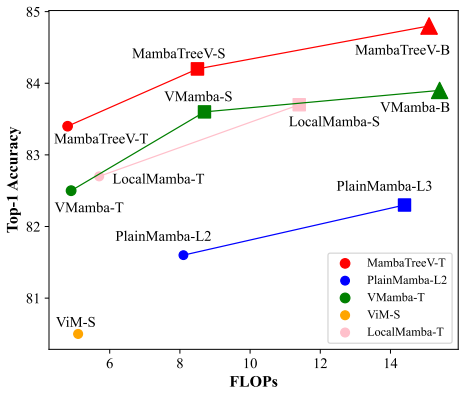

Figure 5: **Classification performance comparison among SSM-based vision foundation models.**

Performance Comparison. We compare various SSM-based visual foundation models as shown in Fig. 5, with different colors representing different models and different shapes indicating different model scales. The size of each shape indicates the number of model parameters. The horizontal axis denotes FLOPs and the vertical axis represents the Top-1 accuracy of the corresponding method on ImageNet-1K val dataset. Fig. 5 demonstrates that MambaTreeV is the best choice in terms of efficiency and effectiveness.

A.2 Object Detection.

For a fair comparison, we conduct the evaluation following common practice [62, 41, 43]. The models are trained with eight 32GB V100 GPUs by default. The input image is resized so that the shorter side is 800 pixels, while the longer side does not exceed 1333 pixels during the $1\times$ schedule. The number of warmup steps is set to 500 in the $1\times$ schedule. For $3\times$ schedule, the shorter side is resized to 480-800 pixels and the longer side does not exceed 1333 pixels. The number of warmup steps is set to 1000 in $3\times$ schedule. Results shown in Table 8 demonstrate the effectiveness of MambaTreeV in object detection and instance segmentation on COCO val2017.

A.3 Semantic Segmentation.

We optimize our MambaTreeV-T/S using AdamW optimizer with an initial learning rate of 6×10^{-5} which is decayed by a rate of 1.0 with the polynomial decay schedule following [62, 27, 57, 16, 28]. The number of warmup iters is set to 1600 with an initial learning rate of 1×10^{-6} [41, 34, 46, 67]. The default input resolution is 512×512 as well as FLOPs are calculated with an input size of 512×2048. The models are trained with eight 32GB V100 GPUs by default.

https://doi.org/10.52202/079017-2398

Method	#FLOPs.	Mask R-CNN 1× Schedule						Mask R-CNN 3× MS Schedule					
		AP^b	AP^b_{50}	AP^b_{75}	AP^m	AP^m_{50}	AP^m_{75}	AP^b	AP^b_{50}	AP^b_{75}	AP^m	AP^m_{50}	AP^m_{75}
Swin-T [43]	267G	42.7	65.2	46.8	39.3	62.2	42.2	46.0	68.1	50.3	41.6	65.1	44.9
ConvNeXt-T [44]	262G	44.2	66.6	48.3	40.1	63.3	42.8	46.2	67.9	50.8	41.7	65.0	44.9
CSWin-T [14]	279G	46.7	68.6	51.3	42.2	65.6	45.4	49.0	70.7	53.7	43.6	67.9	46.6
ViM-S [77]	218G	44.9	67.1	49.3	41.0	64.2	44.1	-	-	-	-	-	-
VMamba-T [41]	286G	46.5	68.5	50.7	42.1	65.5	45.3	48.5	69.9	52.9	43.2	66.8	46.3
L-Vmamba-T [34]	291G	46.7	68.7	50.8	42.2	65.7	45.5	48.7	70.1	53.0	43.4	67.0	46.4
MambaTreeV-T (Ours)	265G	**47.0**	**69.4**	**51.5**	**42.7**	**66.4**	**46.0**	**49.0**	**70.8**	**54.0**	**43.8**	**67.6**	**47.1**
Vit-Adapter-S [4]	403G	44.7	65.8	48.3	39.9	62.5	42.8	48.2	69.7	52.5	42.8	66.4	45.9
Swin-S [43]	354G	44.8	66.6	48.9	40.9	63.4	44.2	48.2	69.8	52.8	43.2	67.0	46.1
ConvNeXt-T [44]	348G	45.4	67.9	50.0	41.8	65.2	45.1	47.9	70.0	52.7	42.9	66.9	46.2
InternImage-S [62]	340G	47.8	69.8	52.8	43.3	67.1	46.7	49.7	71.1	54.5	44.5	68.5	47.8
VMamba-S [41]	400G	48.2	69.7	52.5	43.0	66.6	46.4	49.7	70.4	54.2	44.0	67.6	47.3
L-Vmamba-S [34]	414G	48.4	69.9	52.7	43.2	66.7	46.5	49.9	70.5	54.4	44.1	67.8	47.4
MambaTreeV-S (Ours)	341G	**48.6**	**70.3**	**53.5**	**43.6**	**67.5**	**47.1**	**50.1**	**71.2**	**54.9**	**44.6**	**68.7**	**47.8**

Table 8: **Object detection and instance segmentation performance on COCO val2017.** AP^b and AP^m indicate the mAP of detection and segmentation, respectively. MS indicates the multi-scale training strategy.

B Language Tree Topology Scanning Operator

Algorithm 2 Language Tree Scanning

Input: Input feature $\{x_i\}_{i=1}^L$; Input matrix $\{\bar{\mathbf{B}}_i\}_{i=1}^L$; State matrix $\{\bar{\mathbf{A}}_i\}_{i=1}^L$; Gradient of loss to hidden states $\{\frac{\partial Loss}{\partial h_i}\}_{i=1}^L$; Minimum Spanning Tree $\mathcal{G}_T$.

Traverse Path: $Root, \ldots, Leaf \leftarrow BFS(\mathcal{G}_T)$ ▷ *Breadth-first topological order of* $\mathcal{G}_T$

Forward:

Initialization: $\{\xi_i\}_{i=1}^L \leftarrow \{x_i\}_{i=1}^L$

for $i \leftarrow Leaf$ to $Root$ **do**

$\xi_i = \bar{\mathbf{B}}_i x_i + \sum_{\forall j \in \{t|\text{Par}(t)=i\}} \xi_j \bar{\mathbf{A}}_j$

end for

Backward:

for $i \leftarrow Root$ to $Leaf$ **do**

if i is $Root$ **then**

$\frac{\partial Loss}{\partial x_i} = \eta_i \bar{\mathbf{B}}_i$, $\frac{\partial Loss}{\partial \bar{\mathbf{B}}_i} = \eta_i x_i$, $\frac{\partial Loss}{\partial \bar{\mathbf{A}}_i} = 0$

else

$\frac{\partial Loss}{\partial x_i} = \frac{\partial Loss}{\partial h_i} \bar{\mathbf{B}}_i + \bar{\mathbf{A}}_i \frac{\partial Loss}{\partial x_{\text{Par}(i)}} \bar{\mathbf{B}}_i$, $\frac{\partial Loss}{\partial \bar{\mathbf{B}}_i} = \frac{\partial Loss}{\partial h_i} x_i + \bar{\mathbf{A}}_i \frac{\partial Loss}{\partial \bar{\mathbf{B}}_{\text{Par}(i)}} x_i$

$\frac{\partial Loss}{\partial \bar{\mathbf{A}}_i} = \frac{\partial Loss}{\partial x'_{Par(i)}} h_i$

end if

end for

Output: Hidden states $\{h_i\}_{i=1}^L$; Grad. of loss to input feature $\{\frac{\partial Loss}{\partial x_i}\}_{i=1}^L$; Grad. of loss to input matrix $\{\frac{\partial Loss}{\partial \bar{\mathbf{B}}_i}\}_{i=1}^L$; Grad. of loss to state matrix $\{\frac{\partial Loss}{\partial \bar{\mathbf{A}}_i}\}_{i=1}^L$.

C Algorithm Proof

In this section, we present detailed proofs for our tree scanning algorithm. The definitions of symbols are consistent with those in the main paper.

C.1 Proof for Algorithm 1.

We randomly take a vertex in the MST $\mathcal{G}_T$ as the $root$. According to the definition of the tree scanning algorithm introduced in Sec. 3.2, we can derive h_{root} as follows:

$$h_{root} = \sum_{\forall j \in C_{root}} S(E_{root,j}) \bar{\mathbf{B}}_j x_j, \quad S(E_{root,j}) = \prod_{k \in N_{root,j}} \bar{\mathbf{A}}_k, \tag{9}$$

which shows a process of aggregation from all leaf vertices to the $root$. Therefore, each vertex is only related to its child in this period. Taking vertex m as an example, the Aggr_m can be derived as:

$$\text{Aggr}_m(x) = \bar{\mathbf{B}}_m x_m + \sum_{\forall k \in \{t | \text{Par}(t) = i\}} \text{Aggr}_k(x) \bar{\mathbf{A}}_k. \tag{10}$$

We assume that one of the child of m is n and h_n can be derived as following:

$$h_n = \text{Aggr}_n(x) + \bar{\mathbf{A}}_n \widetilde{\text{Aggr}}_m(x), \tag{11}$$

where $\widetilde{\text{Aggr}}_m(x)$ indicates the aggregation value from the vertices $\in \Omega \backslash C_m^{root}$ to vertex m. Therefore, we can obtain the propagation relationship between the hidden state of parent m and child n:

$$\begin{aligned} h_n &= \text{Aggr}_n(x) + \bar{\mathbf{A}}_n \widetilde{\text{Aggr}}_m(x) \\ &= \text{Aggr}_n(x) + \bar{\mathbf{A}}_n (h_m - \bar{\mathbf{A}}_n \text{Aggr}_n(x)) \\ &= \bar{\mathbf{A}}_n h_m + (1 - \bar{\mathbf{A}}_n^2) \text{Aggr}_n(x) \end{aligned} \tag{12}$$

Through the above derivation, we can calculate $\{h_i\}_{i=1}^L$ with only two traversals ($i.e.$, the aggregation from $leaf$ to $root$ and the propagation from $root$ to $leaf$) in the forward process as shown in Algorithm 1, thereby reducing the computational complexity from $\mathcal{O}(L^2)$ to $\mathcal{O}(L)$.

Next, we analyze the backpropagation process in Algorithm 1. According to the chain rule, we can easily calculate the derivative of $loss$ with respect to x_i:

$$\begin{aligned} \frac{\partial loss}{\partial x_i} &= \sum_{j \in \Omega} \frac{\partial \text{ loss}}{\partial h_j} \frac{\partial h_j}{\partial x_i} \\ &= \bar{\mathbf{B}}_i \sum_{j \in \Omega} S\left(E_{ji}\right) \frac{\partial loss}{\partial h_j} \end{aligned} \tag{13}$$

Similarly, the derivative of $loss$ with respect to $\bar{\mathbf{B}}_i$ is:

$$\begin{aligned} \frac{\partial loss}{\partial \bar{\mathbf{B}}_i} &= \sum_{j \in \Omega} \frac{\partial \text{ loss}}{\partial h_j} \frac{\partial h_j}{\partial \bar{\mathbf{B}}_i} \\ &= x_i \sum_{j \in \Omega} S\left(E_{ji}\right) \frac{\partial loss}{\partial h_j} \end{aligned} \tag{14}$$

The above formulas are equivalent to replacing the input x with $\frac{\partial loss}{\partial h}$ during the forward process.

https://doi.org/10.52202/079017-2398

Subsequently, we assume that the vertex k is the child of vertex l and define C_l^k indicates the children of vertex l with the root of vertex k. $\frac{\partial loss}{\partial \bar{\mathbf{A}}_k}$ is formulated as follows:

$$
\begin{aligned}
\frac{\partial loss}{\partial \bar{\mathbf{A}}_k} &= \sum_{j\in\Omega} \frac{\partial loss}{\partial h_j}\frac{\partial h_j}{\partial \bar{\mathbf{A}}_k} \\
&= \sum_{j\in\Omega} \frac{\partial loss}{\partial h_j} \sum_{p\in\Omega} \frac{\partial S(E_{jp})\bar{\mathbf{B}}_p x'_p}{\partial \bar{\mathbf{A}}_k} \\
&= \sum_{j\in C_l^k} \frac{\partial loss}{\partial h_j} \sum_{p\in C_k^l} S(E_{kp})S(E_{jl})\bar{\mathbf{B}}_p x'_p + \sum_{j\in C_k^l} \frac{\partial loss}{\partial h_j} \sum_{p\in C_l^k} S(E_{kj})S(E_{pl})\bar{\mathbf{B}}_p x'_p \\
&= \sum_{j\in C_l^k} S(E_{jl})\frac{\partial loss}{\partial h_j} \sum_{p\in C_k^l} S(E_{kp})\bar{\mathbf{B}}_p x'_p + \sum_{j\in C_k^l} S(E_{kj})\frac{\partial loss}{\partial h_j} \sum_{p\in C_l^k} S(E_{pl})\bar{\mathbf{B}}_p x'_p \\
&= (\frac{\partial Loss}{\partial x_k} - \bar{\mathbf{A}}_k \mathrm{Aggr}_k(\frac{\partial loss}{\partial h})) * \mathrm{Aggr}_k(x) + \mathrm{Aggr}_k(\frac{\partial loss}{\partial h}) * (h_k - \bar{\mathbf{A}}_k \mathrm{Aggr}_k(x)) \\
&= \frac{\partial Loss}{\partial x_k}\mathrm{Aggr}_k(x) + \mathrm{Aggr}_k(\frac{\partial loss}{\partial h})h_k - 2\bar{\mathbf{A}}_k \mathrm{Aggr}_k(\frac{\partial loss}{\partial h})\mathrm{Aggr}_k(x) \\
&\triangleq \frac{\partial Loss}{\partial x_k}\xi_k + \eta_k h_k - 2\bar{\mathbf{A}}_k \eta_k \xi_k \quad (\textit{definition in Algorithm 1})
\end{aligned}
\tag{15}
$$

So far we have completed the proof of forward and back-propagation of Algorithm 1.

C.2 Proof for Algorithm 2.

We only take the last token as root and replace the transition source from Ω to C_i in sequence modeling tasks like nature language understanding to ensure causality. Therefore, only one traversal (from $leaf$ to $root$) is required for the forward process, and another traversal (from $root$ to $leaf$) is needed for the backpropagation process. The proof is similar to the Algorithm 1.

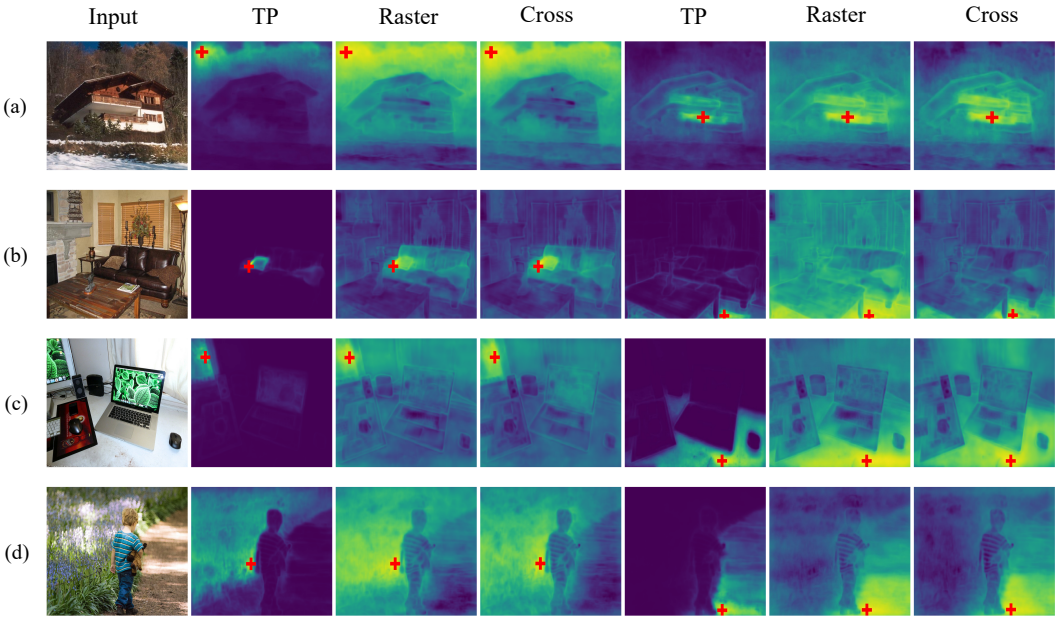

Figure 6: **Visualization of affinity maps in the specific position.** The Location is marked by the red cross in each affinity map. TP represents our Tree Scanning Algorithm.

https://doi.org/10.52202/079017-2398

D More Qualitative Results

Fig. 6 displays additional qualitative comparisons between our algorithm and previous scanning strategies ($e.g.$, cross-scanning and raster-scanning), which shows our advanced capability to perceive detailed structural information and capture long-range dependencies.

E Statistical Significance

Method	PIQA	Arc-Easy	SST	WinoGrande	LAM-ppl	Race	Openbookqa
MambaTreeL (Ours)	0.011	0.010	0.016	0.014	0.553	0.014	0.018

Table 9: **Standard error on language model benchmarks.** LAM-ppl indicates LAMBADA [49].

We calculate the standard deviation of our MambaTreeL on language model benchmarks in the open-sourced lm-evaluation-harness project as shown in Table 9.

https://doi.org/10.52202/079017-2398

NeurIPS Paper Checklist

1. **Claims**

   Question: Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope?

   Answer: [Yes]

   Justification: The claims in the abstract and introduction accurately reflect our motivation and contribution.

   Guidelines:

   - The answer NA means that the abstract and introduction do not include the claims made in the paper.
   - The abstract and/or introduction should clearly state the claims made, including the contributions made in the paper and important assumptions and limitations. A No or NA answer to this question will not be perceived well by the reviewers.
   - The claims made should match theoretical and experimental results, and reflect how much the results can be expected to generalize to other settings.
   - It is fine to include aspirational goals as motivation as long as it is clear that these goals are not attained by the paper.

2. **Limitations**

   Question: Does the paper discuss the limitations of the work performed by the authors?

   Answer: [Yes]

   Justification: We have discussed the limitations of our work.

   Guidelines:

   - The answer NA means that the paper has no limitation while the answer No means that the paper has limitations, but those are not discussed in the paper.
   - The authors are encouraged to create a separate "Limitations" section in their paper.
   - The paper should point out any strong assumptions and how robust the results are to violations of these assumptions (e.g., independence assumptions, noiseless settings, model well-specification, asymptotic approximations only holding locally). The authors should reflect on how these assumptions might be violated in practice and what the implications would be.
   - The authors should reflect on the scope of the claims made, e.g., if the approach was only tested on a few datasets or with a few runs. In general, empirical results often depend on implicit assumptions, which should be articulated.
   - The authors should reflect on the factors that influence the performance of the approach. For example, a facial recognition algorithm may perform poorly when image resolution is low or images are taken in low lighting. Or a speech-to-text system might not be used reliably to provide closed captions for online lectures because it fails to handle technical jargon.
   - The authors should discuss the computational efficiency of the proposed algorithms and how they scale with dataset size.
   - If applicable, the authors should discuss possible limitations of their approach to address problems of privacy and fairness.
   - While the authors might fear that complete honesty about limitations might be used by reviewers as grounds for rejection, a worse outcome might be that reviewers discover limitations that aren't acknowledged in the paper. The authors should use their best judgment and recognize that individual actions in favor of transparency play an important role in developing norms that preserve the integrity of the community. Reviewers will be specifically instructed to not penalize honesty concerning limitations.

3. **Theory Assumptions and Proofs**

   Question: For each theoretical result, does the paper provide the full set of assumptions and a complete (and correct) proof?

   Answer: [Yes]

Justification: We provide detailed proofs for our tree topology scanning algorithm in both the main paper and the appendix.

Guidelines:

- The answer NA means that the paper does not include theoretical results.
- All the theorems, formulas, and proofs in the paper should be numbered and cross-referenced.
- All assumptions should be clearly stated or referenced in the statement of any theorems.
- The proofs can either appear in the main paper or the supplemental material, but if they appear in the supplemental material, the authors are encouraged to provide a short proof sketch to provide intuition.
- Inversely, any informal proof provided in the core of the paper should be complemented by formal proofs provided in appendix or supplemental material.
- Theorems and Lemmas that the proof relies upon should be properly referenced.

4. **Experimental Result Reproducibility**

   Question: Does the paper fully disclose all the information needed to reproduce the main experimental results of the paper to the extent that it affects the main claims and/or conclusions of the paper (regardless of whether the code and data are provided or not)?

   Answer: [Yes]

   Justification: We describe the experimental details both in the main paper and the appendix. We will make our code publicly available, along with detailed instructions for reproduction.

   Guidelines:

   - The answer NA means that the paper does not include experiments.
   - If the paper includes experiments, a No answer to this question will not be perceived well by the reviewers: Making the paper reproducible is important, regardless of whether the code and data are provided or not.
   - If the contribution is a dataset and/or model, the authors should describe the steps taken to make their results reproducible or verifiable.
   - Depending on the contribution, reproducibility can be accomplished in various ways. For example, if the contribution is a novel architecture, describing the architecture fully might suffice, or if the contribution is a specific model and empirical evaluation, it may be necessary to either make it possible for others to replicate the model with the same dataset, or provide access to the model. In general. releasing code and data is often one good way to accomplish this, but reproducibility can also be provided via detailed instructions for how to replicate the results, access to a hosted model (e.g., in the case of a large language model), releasing of a model checkpoint, or other means that are appropriate to the research performed.
   - While NeurIPS does not require releasing code, the conference does require all submissions to provide some reasonable avenue for reproducibility, which may depend on the nature of the contribution. For example
     (a) If the contribution is primarily a new algorithm, the paper should make it clear how to reproduce that algorithm.
     (b) If the contribution is primarily a new model architecture, the paper should describe the architecture clearly and fully.
     (c) If the contribution is a new model (e.g., a large language model), then there should either be a way to access this model for reproducing the results or a way to reproduce the model (e.g., with an open-source dataset or instructions for how to construct the dataset).
     (d) We recognize that reproducibility may be tricky in some cases, in which case authors are welcome to describe the particular way they provide for reproducibility. In the case of closed-source models, it may be that access to the model is limited in some way (e.g., to registered users), but it should be possible for other researchers to have some path to reproducing or verifying the results.

5. **Open access to data and code**

https://doi.org/10.52202/079017-2398

Question: Does the paper provide open access to the data and code, with sufficient instructions to faithfully reproduce the main experimental results, as described in supplemental material?

Answer: [Yes]

Justification: We provide the core part of our code in the supplementary material.

Guidelines:

- The answer NA means that paper does not include experiments requiring code.
- Please see the NeurIPS code and data submission guidelines (https://nips.cc/public/guides/CodeSubmissionPolicy) for more details.
- While we encourage the release of code and data, we understand that this might not be possible, so "No" is an acceptable answer. Papers cannot be rejected simply for not including code, unless this is central to the contribution (e.g., for a new open-source benchmark).
- The instructions should contain the exact command and environment needed to run to reproduce the results. See the NeurIPS code and data submission guidelines (https://nips.cc/public/guides/CodeSubmissionPolicy) for more details.
- The authors should provide instructions on data access and preparation, including how to access the raw data, preprocessed data, intermediate data, and generated data, etc.
- The authors should provide scripts to reproduce all experimental results for the new proposed method and baselines. If only a subset of experiments are reproducible, they should state which ones are omitted from the script and why.
- At submission time, to preserve anonymity, the authors should release anonymized versions (if applicable).
- Providing as much information as possible in supplemental material (appended to the paper) is recommended, but including URLs to data and code is permitted.

6. **Experimental Setting/Details**

   Question: Does the paper specify all the training and test details (e.g., data splits, hyperparameters, how they were chosen, type of optimizer, etc.) necessary to understand the results?

   Answer: [Yes]

   Justification: The complete experiment details are introduced in the main paper and the appendix.

   Guidelines:

   - The answer NA means that the paper does not include experiments.
   - The experimental setting should be presented in the core of the paper to a level of detail that is necessary to appreciate the results and make sense of them.
   - The full details can be provided either with the code, in appendix, or as supplemental material.

7. **Experiment Statistical Significance**

   Question: Does the paper report error bars suitably and correctly defined or other appropriate information about the statistical significance of the experiments?

   Answer: [Yes]

   Justification: We provide the statistical analysis results in the appendix.

   Guidelines:

   - The answer NA means that the paper does not include experiments.
   - The authors should answer "Yes" if the results are accompanied by error bars, confidence intervals, or statistical significance tests, at least for the experiments that support the main claims of the paper.
   - The factors of variability that the error bars are capturing should be clearly stated (for example, train/test split, initialization, random drawing of some parameter, or overall run with given experimental conditions).

- The method for calculating the error bars should be explained (closed form formula, call to a library function, bootstrap, etc.)
- The assumptions made should be given (e.g., Normally distributed errors).
- It should be clear whether the error bar is the standard deviation or the standard error of the mean.
- It is OK to report 1-sigma error bars, but one should state it. The authors should preferably report a 2-sigma error bar than state that they have a 96% CI, if the hypothesis of Normality of errors is not verified.
- For asymmetric distributions, the authors should be careful not to show in tables or figures symmetric error bars that would yield results that are out of range (e.g. negative error rates).
- If error bars are reported in tables or plots, The authors should explain in the text how they were calculated and reference the corresponding figures or tables in the text.

8. **Experiments Compute Resources**

   Question: For each experiment, does the paper provide sufficient information on the computer resources (type of compute workers, memory, time of execution) needed to reproduce the experiments?

   Answer: [Yes]

   Justification: We have described the resources required to perform our experiments in the appendix.

   Guidelines:

   - The answer NA means that the paper does not include experiments.
   - The paper should indicate the type of compute workers CPU or GPU, internal cluster, or cloud provider, including relevant memory and storage.
   - The paper should provide the amount of compute required for each of the individual experimental runs as well as estimate the total compute.
   - The paper should disclose whether the full research project required more compute than the experiments reported in the paper (e.g., preliminary or failed experiments that didn't make it into the paper).

9. **Code Of Ethics**

   Question: Does the research conducted in the paper conform, in every respect, with the NeurIPS Code of Ethics https://neurips.cc/public/EthicsGuidelines?

   Answer: [Yes]

   Justification: Our work conforms with the NeurIPS Code of Ethics in every respect

   Guidelines:

   - The answer NA means that the authors have not reviewed the NeurIPS Code of Ethics.
   - If the authors answer No, they should explain the special circumstances that require a deviation from the Code of Ethics.
   - The authors should make sure to preserve anonymity (e.g., if there is a special consideration due to laws or regulations in their jurisdiction).

10. **Broader Impacts**

    Question: Does the paper discuss both potential positive societal impacts and negative societal impacts of the work performed?

    Answer: [No]

    Justification: Our work mainly focuses on the research of basic neural network architecture. We have identified no potential negative social impacts.

    Guidelines:

    - The answer NA means that there is no societal impact of the work performed.
    - If the authors answer NA or No, they should explain why their work has no societal impact or why the paper does not address societal impact.

https://doi.org/10.52202/079017-2398

- Examples of negative societal impacts include potential malicious or unintended uses (e.g., disinformation, generating fake profiles, surveillance), fairness considerations (e.g., deployment of technologies that could make decisions that unfairly impact specific groups), privacy considerations, and security considerations.
- The conference expects that many papers will be foundational research and not tied to particular applications, let alone deployments. However, if there is a direct path to any negative applications, the authors should point it out. For example, it is legitimate to point out that an improvement in the quality of generative models could be used to generate deepfakes for disinformation. On the other hand, it is not needed to point out that a generic algorithm for optimizing neural networks could enable people to train models that generate Deepfakes faster.
- The authors should consider possible harms that could arise when the technology is being used as intended and functioning correctly, harms that could arise when the technology is being used as intended but gives incorrect results, and harms following from (intentional or unintentional) misuse of the technology.
- If there are negative societal impacts, the authors could also discuss possible mitigation strategies (e.g., gated release of models, providing defenses in addition to attacks, mechanisms for monitoring misuse, mechanisms to monitor how a system learns from feedback over time, improving the efficiency and accessibility of ML).

11. **Safeguards**

    Question: Does the paper describe safeguards that have been put in place for responsible release of data or models that have a high risk for misuse (e.g., pretrained language models, image generators, or scraped datasets)?

    Answer: [NA]

    Justification: Our paper poses no such risks.

    Guidelines:

    - The answer NA means that the paper poses no such risks.
    - Released models that have a high risk for misuse or dual-use should be released with necessary safeguards to allow for controlled use of the model, for example by requiring that users adhere to usage guidelines or restrictions to access the model or implementing safety filters.
    - Datasets that have been scraped from the Internet could pose safety risks. The authors should describe how they avoided releasing unsafe images.
    - We recognize that providing effective safeguards is challenging, and many papers do not require this, but we encourage authors to take this into account and make a best faith effort.

12. **Licenses for existing assets**

    Question: Are the creators or original owners of assets (e.g., code, data, models), used in the paper, properly credited and are the license and terms of use explicitly mentioned and properly respected?

    Answer: [Yes]

    Justification: The creators or original owners of assets (e.g., code, data, models), used in the paper, are properly credited as well as the license and terms of use explicitly are mentioned and properly respected.

    Guidelines:

    - The answer NA means that the paper does not use existing assets.
    - The authors should cite the original paper that produced the code package or dataset.
    - The authors should state which version of the asset is used and, if possible, include a URL.
    - The name of the license (e.g., CC-BY 4.0) should be included for each asset.
    - For scraped data from a particular source (e.g., website), the copyright and terms of service of that source should be provided.

- If assets are released, the license, copyright information, and terms of use in the package should be provided. For popular datasets, `paperswithcode.com/datasets` has curated licenses for some datasets. Their licensing guide can help determine the license of a dataset.
- For existing datasets that are re-packaged, both the original license and the license of the derived asset (if it has changed) should be provided.
- If this information is not available online, the authors are encouraged to reach out to the asset's creators.

13. **New Assets**

    Question: Are new assets introduced in the paper well documented and is the documentation provided alongside the assets?

    Answer: [NA]

    Justification: No new assets introduced.

    Guidelines:

    - The answer NA means that the paper does not release new assets.
    - Researchers should communicate the details of the dataset/code/model as part of their submissions via structured templates. This includes details about training, license, limitations, etc.
    - The paper should discuss whether and how consent was obtained from people whose asset is used.
    - At submission time, remember to anonymize your assets (if applicable). You can either create an anonymized URL or include an anonymized zip file.

14. **Crowdsourcing and Research with Human Subjects**

    Question: For crowdsourcing experiments and research with human subjects, does the paper include the full text of instructions given to participants and screenshots, if applicable, as well as details about compensation (if any)?

    Answer: [NA]

    Justification: Our work does not involve crowdsourcing nor research with human subjects.

    Guidelines:

    - The answer NA means that the paper does not involve crowdsourcing nor research with human subjects.
    - Including this information in the supplemental material is fine, but if the main contribution of the paper involves human subjects, then as much detail as possible should be included in the main paper.
    - According to the NeurIPS Code of Ethics, workers involved in data collection, curation, or other labor should be paid at least the minimum wage in the country of the data collector.

15. **Institutional Review Board (IRB) Approvals or Equivalent for Research with Human Subjects**

    Question: Does the paper describe potential risks incurred by study participants, whether such risks were disclosed to the subjects, and whether Institutional Review Board (IRB) approvals (or an equivalent approval/review based on the requirements of your country or institution) were obtained?

    Answer: [NA]

    Justification: Our work does not involve crowdsourcing nor research with human subjects.

    Guidelines:

    - The answer NA means that the paper does not involve crowdsourcing nor research with human subjects.
    - Depending on the country in which research is conducted, IRB approval (or equivalent) may be required for any human subjects research. If you obtained IRB approval, you should clearly state this in the paper.

https://doi.org/10.52202/079017-2398

- We recognize that the procedures for this may vary significantly between institutions and locations, and we expect authors to adhere to the NeurIPS Code of Ethics and the guidelines for their institution.
- For initial submissions, do not include any information that would break anonymity (if applicable), such as the institution conducting the review.