LOOPMAMBA: RETHINKING STATE SPACE MODELS FROM AN ENERGY PERSPECTIVE

Anonymous authors
Paper under double-blind review

ABSTRACT

As a powerful alternative to Transformers, State Space Models (SSMs) have demonstrated remarkable effectiveness across a wide range of vision tasks. However, existing visual SSM architectures still rely substantially on empirically motivated design choices to adapt SSMs to visual tasks, with limited theoretical understanding of their underlying state transition mechanisms to guide architectural design. In this work, we rethink SSMs from an energy perspective, revealing that state transitions can be viewed as an iterative energy minimization process in which hidden states progressively move toward stable representations. Guided by this insight, we propose LoopMamba, a visual architecture with four Self-loop Layers that performs iterative hidden state refinement through Variable-Parameter State Transition. Extensive experiments and analyses on image classification, object detection, instance segmentation, and semantic segmentation demonstrate the effectiveness of LoopMamba. On ImageNet-1K, LoopMamba achieves a 1.8% improvement in Top-1 accuracy over Vim-S with 31.5% fewer parameters. Overall, LoopMamba offers an effective and interpretable paradigm for SSM-based visual representation learning.

1 INTRODUCTION

State Space Models (SSMs) have recently emerged as a powerful alternative to Transformers and have demonstrated remarkable effectiveness across a wide range of tasks, including language modeling and visual representation learning. Originally developed for sequential data modeling, SSMs provide an efficient mechanism for propagating information through hidden states with near-linear computational complexity and a compact parameterization. The introduction of Mamba (Gu & Dao, 2024) further enhances SSMs through input-dependent state transitions, achieving an attractive balance between modeling capability and computational efficiency. Building on this success, a series of vision SSMs (Zhu et al., 2024; Liu et al., 2024; Shi et al., 2024; Xie et al., 2024; Xiao et al., 2024; Yang et al., 2024a) have been proposed as effective visual backbones, demonstrating the strong potential of SSMs for visual representation learning.

Despite the growing theoretical understanding of SSMs, existing studies have primarily focused on characterizing their memory mechanisms (Gu et al., 2020), parameterization (Gu et al., 2021a; 2022), and selectivity (Muca Cirone et al., 2024; Gu & Dao, 2024), while relatively less attention has been devoted to translating these theoretical insights into SSM-based network architectural design. In the vision domain, existing vision SSMs mainly focus on adapting SSMs to visual tasks through empirically motivated designs, such as scanning strategies (Zhu et al., 2024; Liu et al., 2024), multi-scale feature extraction (Shi et al., 2024), and the integration of SSMs with other architectural components (Hatamizadeh & Kautz, 2025). As a result, architectural improvements often involve increasingly sophisticated modules or deeper networks, with relatively limited emphasis on deriving architectural principles from the underlying state transition dynamics. This disconnect between theoretical understanding and architectural design motivates us to revisit SSMs from a more interpretable perspective.

This raises a fundamental question: **can SSMs be understood from a more interpretable perspective, and can such an understanding inspire new architectural insights**? To address this question, we rethink the state transition mechanism of SSMs from an energy-based perspective. We show that an SSM state transition can be formulated as a gradient descent step of an Energy-

Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping the LLM and submitting your original text: we, and the authors, are much more interested in your unedited thoughts than in what an LLM has to say. AI-assisted or not, you are putting your name and reputation behind your review: LLM-generated falsehoods, hallucinations or misrepresentations are subject to disciplinary action, which may include desk-rejecting all papers you have authored.

Based Model (EBM), providing an energy-based view of iterative hidden state refinement. Under this interpretation, state transition is not merely a mechanism for propagating information across the sequence, but an iterative optimization process over the hidden representation. This perspective provides a new understanding of SSM architecture, in which one SSM layer corresponds to one optimization step, while stacking multiple SSM layers amounts to performing successive optimization steps with different transition parameters. Therefore, rather than increasing the complexity of SSM architectures by stacking more or increasingly complex layers, we ask whether this optimization process can instead be iterated through a shared transition process.

Motivated by this perspective, we propose LoopMamba, an SSM architecture that performs iterative hidden state optimization through variable-parameter self-loop refinement. Instead of stacking multiple independent layers, LoopMamba repeatedly applies a shared state transition within each stage, allowing the hidden state to undergo multiple optimization steps with a substantially reduced parameter overhead. Specifically, to enable iterative refinement with shared state transitions, we introduce Variable-Parameter State Transition (VPST) in the standard SSM layer, which shares the SSM state transition parameters across iterations while incorporating lightweight iteration-specific parameters. To achieve iteration-specific adaptation, we further design an Iteration-specific Lightweight State Fusion (ILSF) module in VPST to enhance the hidden state representation capability and the flexibility of the optimization trajectory through multi-step state fusion and iteration-specific low-rank adaptation. Through this design, different optimization iterations can learn distinct refinement directions while retaining the lightweight nature of parameter sharing. We conduct extensive experiments and analyses on image classification, object detection, instance segmentation, and semantic segmentation, which demonstrate the effectiveness of LoopMamba and the proposed energy-guided iterative optimization perspective. Our contributions can be summarized as follows:

- We provide a new energy-based interpretation of SSMs, revealing that the state transition can be viewed as an energy-guided iterative optimization process, where each transition progressively drives the hidden state toward a more stable representation.
- Based on this perspective, we propose LoopMamba, a novel SSM architecture that enables self-loop iterative optimization of the hidden state through Variable-Parameter State Transition (VPST), which provides both optimization flexibility and enhanced hidden state representation capability with limited parameter overhead.
- We conduct extensive experiments on various tasks. The results demonstrate that LoopMamba achieves competitive performance against state-of-the-art methods, validating the effectiveness of the proposed self-loop design.

2 Related Work

2.1 State Space Models

State Space Models (SSMs) originate from control theory and signal processing, where they describe dynamical systems by modeling the evolution of hidden states. To adapt SSMs to deep learning, LSSL (Gu et al., 2021b) employs the generalized bilinear transform to discretize continuous SSMs, while S4 (Gu et al., 2021a) introduces a structured parameterization that enables efficient long-sequence modeling. S5 (Smith et al., 2023) further simplifies the parameterization and improves training stability. These advances have established SSMs as an efficient alternative to attention-based architectures and enabled their applications across various tasks (Islam & Bertasius, 2022; Wang et al., 2023; Islam et al., 2023).

Focusing on SSM architecture design, Vision Mamba (Vim) (Zhu et al., 2024) extends Mamba with bidirectional scanning to capture spatial context, while VMamba (Liu et al., 2024) introduces 2D selective scan to better model image structures. PlainMamba (Yang et al., 2024a) replaces raster scanning with continuous scanning to preserve spatial dependencies. Subsequent works explore more adaptive scanning strategies, including multi-scale scanning in MSVMamba (Shi et al., 2024), content-aware scanning granularity in QuadMamba (Xie et al., 2024), and input-adaptive tree-based scanning in MambaTree (Xiao et al., 2024). More recently, DefMamba (Liu et al., 2025) replaces fixed scanning paths with deformable scanning, allowing state-space interactions to adapt to image content, while MobileMamba (He et al., 2025) explores lightweight multi-receptive-field designs to improve the efficiency of visual Mamba architectures. MambaVision (Hatamizadeh & Kautz, 2025)

adopts a hybrid Mamba-Transformer architecture that combines efficient state space modeling with self-attention to enhance global spatial representation. In contrast, PVMamba (Xie et al., 2025) and Spatial-Mamba (Xiao et al., 2025) focus on dynamic and structure-aware state aggregation. Overall, existing visual SSM studies mainly focus on adapting SSMs to visual data through different scanning, propagation, and aggregation strategies, while the underlying state transition mechanism is rarely leveraged to guide architectural design.

2.2 Energy-Based Models

Energy-Based Models (EBMs) provide a general framework for modeling complex data distributions by assigning an energy value to each configuration, where lower energy corresponds to more plausible states. Unlike explicit probabilistic models, EBMs do not require normalized likelihoods, providing a flexible framework for representation and structured prediction (LeCun et al., 2006). Recent studies have further explored the role of energy functions in modern generative and representation learning paradigms. Energy-based diffusion models incorporate energy functions into the diffusion process to improve generation and sampling (Geng et al., 2024), while Energy-based Diffusion Language Models (Xu et al., 2025) extend this idea to sequence generation by modeling the energy of an entire sequence at each diffusion step. Beyond generative modeling, EnergyMoGen (Zhang et al., 2025) employs energy-based diffusion to facilitate semantic composition through adaptive gradient-based optimization. More recently, Energy-Based Transformers (Gladstone et al., 2026) formulate prediction as energy minimization and perform iterative gradient-based inference, demonstrating the scalability of energy-based optimization across both language and visual modalities.

3 Methods

3.1 Preliminary

3.1.1 State Space Model

State Space Models (SSMs) originate from classical control theory and model sequential dynamics through latent state transitions (Kalman, 1960). Given an input $u(t) \in \mathbb{R}$, output $y(t) \in \mathbb{R}$, and latent state $h(t) \in \mathbb{R}^N$, a continuous-time SSM is defined as

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

where $\mathbf{A}$, $\mathbf{B}$, and $\mathbf{C}$ denote the state transition, input, and output matrices, respectively, and $\mathbf{D}$ is the residual matrix. Modern SSMs (Gu et al., 2021a; Gu & Dao, 2024) discretize the continuous system using zero-order hold (ZOH) with step size $\mathbf{\Delta}$:

$$\bar{\mathbf{A}} = \exp(\mathbf{\Delta A}), \qquad \bar{\mathbf{B}} = (\mathbf{\Delta A})^{-1}\left(\exp(\mathbf{\Delta A}) - \mathbf{I}\right)\mathbf{\Delta B}. \tag{2}$$

The resulting discrete SSM is given by

$$h_t = \bar{\mathbf{A}}h_{t-1} + \bar{\mathbf{B}}x_t, \qquad y_t = \mathbf{C}h_t + \mathbf{D}x_t. \tag{3}$$

3.1.2 Energy-Based Model

Energy-Based Models (EBMs) represent a system through an energy function that assigns a scalar energy to every possible configuration of variables, where lower energy corresponds to more favorable and stable configurations (LeCun et al., 2006; 2022). Formally, given an input context x, EBMs define an energy function $E_\theta(x, \hat{y})$, where $\hat{y}$ denotes the prediction to be optimized and θ represents the learnable parameters of the energy function. Starting from an initial prediction $\hat{y}_0$, the optimization is performed via gradient descent:

$$\hat{y}_{i+1} = \hat{y}_i - \eta \nabla_{\hat{y}_i} E_\theta(x, \hat{y}_i), \tag{4}$$

where η is the optimization step size and $\nabla_{\hat{y}_i} E_\theta(x, \hat{y}_i)$ represents the gradient of the energy function with respect to $\hat{y}_i$. Through successive optimization steps, $\hat{y}$ progressively converges toward a local minimum of the energy landscape, yielding a lower-energy and more stable prediction.

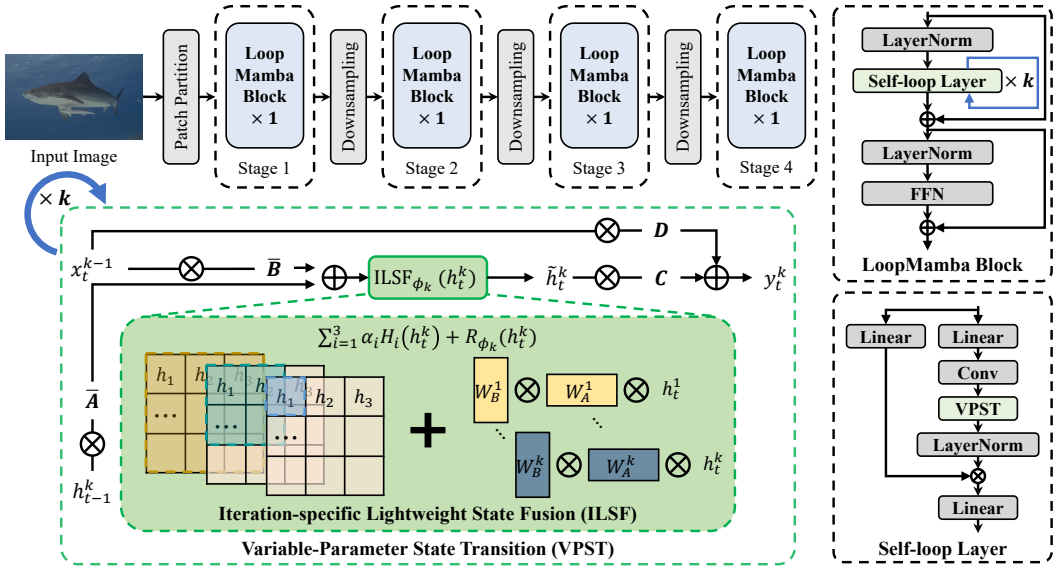

Figure 1: Illustration of LoopMamba's architecture and Variable-Parameter State Transition (VPST).

3.2 Energy-based Interpretation of State Space Models

In this section, we reinterpret the state transition of SSMs from an energy-based perspective. Consider the discrete state transition of h in Eq. 3. Given an input x and a optimization step size η, we define the following quadratic energy function:

$$E_\theta(x, h) = \frac{1}{2} h^\top \mathbf{M} h - b(x)^\top h, \tag{5}$$

where $\mathbf{M} = \frac{1}{\eta}(\mathbf{I} - \bar{\mathbf{A}})$ and $b(x) = \frac{1}{\eta}\bar{\mathbf{B}}x$.

Theorem 1 (Energy-based interpretation of SSM state transition)**.** *Suppose that* $\mathbf{I} - \bar{\mathbf{A}}$ *is symmetric. Then the SSM state transition in Eq. (3) is equivalent to one gradient descent step on the quadratic energy function in Eq. (5), i.e.,*

$$h_t = h_{t-1} - \eta \nabla_{h_{t-1}} E_\theta(x_t, h_{t-1}). \tag{6}$$

The equivalence follows by taking the gradient of the quadratic energy function with respect to h:

$$\nabla_h E_\theta(x, h) = \frac{1}{\eta}(\mathbf{I} - \bar{\mathbf{A}})h - \frac{1}{\eta}\bar{\mathbf{B}}x. \tag{7}$$

Substituting Eq. 7 into Eq. 6 recovers the original SSM state transition. When $\mathbf{I} - \bar{\mathbf{A}} \succ 0$, the energy function is strictly convex and coercive, and thus admits a unique global minimizer given by

$$h^* = (\mathbf{I} - \bar{\mathbf{A}})^{-1}\bar{\mathbf{B}}x. \tag{8}$$

For stable diagonal SSMs, both conditions hold, yielding the above energy interpretation and a unique global minimizer. Detailed derivations are provided in Appendix A.

3.3 LoopMamba

3.3.1 Architecture Design of LoopMamba

Theorem 1 establishes a correspondence between SSM state transitions and gradient descent in EBMs, providing an energy-based interpretation of the hidden state refinement process. Motivated by this perspective, we propose LoopMamba, a hierarchical architecture that iteratively refines representations by repeatedly applying a shared optimization process within each stage. As illustrated in Fig. 1, LoopMamba adopts patch embedding and progressive downsampling, with each stage consisting of a LoopMamba block based on the standard SSM architecture. The Self-loop Layer within

each block also follows the standard SSM design, while introducing a Variable-Parameter State Transition (VPST) equipped with an Iteration-specific Lightweight State Fusion (ILSF) module to enhance the representation capability of the hidden state and the flexibility of iterative refinement. The detailed configurations are provided in Appendix B.

3.3.2 VARIABLE-PARAMETER STATE TRANSITION

According to the energy-based interpretation, a state transition can be regarded as a single gradient descent step in Eq. 4, which indicates that repeatedly applying SSM state transitions corresponds to performing iterative optimization on the hidden state. Thus, we introduce Variable-Parameter State Transition (VPST, shown in Fig. 1) in LoopMamba, which recursively refines the hidden state through the shared state transition, while Iteration-specific Lightweight State Fusion enhances the flexibility of the optimization trajectory with a compact parameter budget.

Shared State Transition. Specifically, to achieve iterative optimization, VPST first performs the shared state transition:

$$h_t^k = \bar{\mathbf{A}} h_{t-1}^k + \bar{\mathbf{B}} x_t^{k-1}, \tag{9}$$

where t denotes the sequence step (i.e., patch position), k represents the optimization iteration (starting from 1; x^0 is the partition image), h_t^k is the hidden state at the current iteration, and x_t^{k-1} is the corresponding input representation from the previous self-loop iteration.

To improve the flexibility and representation capability of each optimization step, the updated hidden state is further processed by an Iteration-specific Lightweight State Fusion (ILSF) module,

$$\tilde{h}_t^k = \mathrm{ILSF}_{\phi_k}(h_t^k), \tag{10}$$

where ϕ_k denotes the iteration-specific adaptive parameters. Finally, the output of the current optimization step is obtained by

$$y_t^k = \mathbf{C}\tilde{h}_t^k + \mathbf{D} x_t^{k-1}, \tag{11}$$

where the output y_t^k is fed into the subsequent optimization step as the input for the next iteration.

Iteration-specific Lightweight State Fusion. Although the shared state transition effectively refines the hidden state iteratively, using identical transition parameters across all optimization steps may limit the diversity of optimization trajectories. Different optimization steps usually correspond to different refinement phases, where identical transition functions may be suboptimal for capturing diverse update trajectories. Moreover, the state transitions produce a sequence of intermediate states encoding diverse information at different steps, yet these states are not explicitly integrated, limiting the potential of iterative refinement.

To address these issues, we propose an Iteration-specific Lightweight State Fusion (ILSF) module, which jointly performs multi-step state fusion and iteration-specific low-rank adaptation, achieving effective multi-step state aggregation and flexible iteration-specific optimization. Given the intermediate state representation h_t^k, ILSF is formulated as:

$$\mathrm{ILSF}_{\phi_k}(h_t^k) = \sum_{i=1}^{3} \alpha_i H_i(h_t^k) + R_{\phi_k}(h_t^k), \tag{12}$$

where the first term denotes the multi-step state fusion operation, and $H_i(\cdot)$ denotes depth-wise convolution with different receptive fields:

$$H_i(h_t^k) = DWConv_i(h_t^k), \tag{13}$$

which use dilation rates of 1, 3, and 5, respectively. The learnable coefficients α_i adaptively aggregate multi-step state features. In practice, h_i^k contains the preceding states $h_{0\sim i}^k$, allowing multi-step state aggregation to be performed directly on h_i^k. While the second term of Eq. 12 denotes the iteration-specific low-rank adaptation:

$$R_{\phi_k}(h_t^k) = \mathbf{W}_B^k(\mathbf{W}_A^k(h_t^k)), \tag{14}$$

where $\mathbf{W}_A^k \in \mathbb{R}^{C\times r}$ and $\mathbf{W}_B^k \in \mathbb{R}^{r\times C}$ are two low-rank projection matrices with an intermediate rank r satisfying $r = \max(4, \lfloor C/32 \rfloor)$, where C denotes the number of channels (defined in Appendix B.). Compared with full-rank transformations, the low-rank design introduces negligible parameter overhead while allowing different optimization steps to learn distinct refinement directions. Overall, ILSF simultaneously enhances state representation capability and iteration flexibility, enabling VPST to perform more effective iterative optimization.

Table 1: Comparison of LoopMamba with state-of-the-art models on ImageNet-1K. LoopMamba achieves competitive performance with a favorable parameter-efficiency trade-off. P.: Parameters. F.: FLOPs. White : CNNs, Gray : Transformers, Yellow : SSMs, Purple : Ours. **Bold**: Best. Underline: Second-best.

Model	P. (M)	F. (G)	Top-1 acc. (%)
LSNet-T (Wang et al., 2025a) (CVPR'25)	11.4	0.3	74.9
SHViT-S2 (Yun & Ro, 2024) (CVPR'24)	11.4	0.4	75.2
SHViT-S4 (Yun & Ro, 2024) (CVPR'24)	16.1	1.0	79.4
Vim-T (Zhu et al., 2024) (ICML'24)	7.0	1.5	76.1
PlainMamba-L1 (Yang et al., 2024a) (BMVC'24)	7.0	3.0	77.9
QuadMamba-T (Xie et al., 2024) (NeurIPS'24)	10.0	2.0	78.2
MobileMamba-S6 (He et al., 2025) (CVPR'25)	15.0	0.7	78.0
Mamba®-T (Wang et al., 2025b) (CVPR'25)	9.0	1.6	77.4
EfficientVMamba-S (Pei et al., 2025) (AAAI'25)	11.0	1.3	78.7
FastVim-T (Kapse et al., 2026) (WACV'26)	7.0	1.2	75.4
LoopMamba-T-4 (Ours)	8.1	2.9	79.7
LoopMamba-T-8 (Ours)	8.5	4.4	**80.2**
ConvNeXt-T (Liu et al., 2022) (CVPR'22)	29.0	4.5	82.1
LSNet-S (Wang et al., 2025a) (CVPR'25)	16.1	0.5	77.8
LSNet-B (Wang et al., 2025a) (CVPR'25)	23.2	1.3	80.3
ViT-B/16 (Dosovitskiy et al., 2021) (ICLR'21)	86.0	55.4	77.9
DeiT-S (Touvron et al., 2021) (ICML'21)	22.0	4.6	79.8
Swin-T (Liu et al., 2021) (ICCV'21)	28.0	4.5	81.2
Vim-S (Zhu et al., 2024) (ICML'24)	26.0	5.1	80.3
PlainMamba-L2 (Yang et al., 2024a) (BMVC'24)	25.0	8.1	81.6
EfficientViM-M4 (Lee et al., 2025) (CVPR'25)	19.6	1.1	79.4
MobileMamba-B2 (He et al., 2025) (CVPR'25)	17.1	2.4	81.6
Mamba®-S (Wang et al., 2025b) (CVPR'25)	28.0	5.5	81.1
EfficientVMamba-B (Pei et al., 2025) (AAAI'25)	33.0	4.0	81.8
FastVim-S (Kapse et al., 2026) (WACV'26)	26.0	4.4	81.1
LoopMamba-B-4 (Ours)	17.8	6.4	82.1
LoopMamba-B-8 (Ours)	18.6	9.6	**82.4**

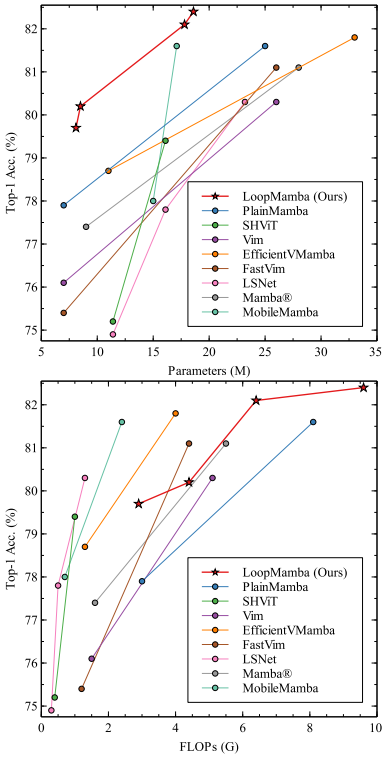

Figure 2: Accuracy–Parameters & Accuracy-FLOPs comparison of different methods on ImageNet-1K. LoopMamba achieves higher accuracy with fewer parameters and competitive complexity.

4 Experiments

4.1 Image Classification

To validate LoopMamba for image classification, we compare it with representative CNNs, Transformers, and SSM-based architectures on ImageNet-1K (Deng et al., 2009), detailed experimental settings are provided in Appendix C. As shown in Table 1 and Fig. 2, LoopMamba provides a favorable accuracy–efficiency trade-off, achieving superior accuracy with substantially fewer parameters while incurring only moderate computational costs.

Specifically, LoopMamba-T-4 achieves 79.7% top-1 accuracy with only 8.1M parameters, surpassing other models such as QuadMamba-T by over 1.0% while using fewer parameters and comparable FLOPs. Similarly, at a larger scale, LoopMamba-B-4 reaches 82.1% accuracy with 17.8M parameters and 6.4G FLOPs, outperforming all compared methods. Moreover, increasing the loop depth consistently improves performance, with LoopMamba-T-8 and LoopMamba-B-8 achieving 80.2% and 82.4%, respectively. These results demonstrate that, unlike stacking independent layers, self-loop optimization reuses the shared SSM state transition, enabling effective representation refinement with minimal parameter overhead.

4.2 Object Detection and Instance Segmentation

To evaluate the generalization ability of LoopMamba, we use pretrained LoopMamba as backbones for Mask R-CNN (He et al., 2017) on COCO 2017 (Lin et al., 2014). As shown in Table 2, LoopMamba achieves strong performance across model scales for both object detection and instance segmentation.

Table 2: Performance of object detection and instance segmentation on the COCO 2017 dataset. LoopMamba achieves strong performance on both object detection and instance segmentation tasks, demonstrating its generalization capability.

Backbone	Mask R-CNN 1×							
	P. (M)	F. (G)	AP^b	AP^b_{50}	AP^b_{75}	AP^m	AP^m_{50}	AP^m_{75}
LSNet-T (CVPR'25)	11.4	0.3	35.0	57.0	37.3	32.7	53.8	34.3
SHViT-S4 (CVPR'24)	16.1	1.0	39.0	61.2	41.9	35.9	57.9	37.9
PlainMamba-L1 (BMVC'24)	7.0	3.0	44.1	64.8	47.9	39.1	61.6	41.9
QuadMamba-T (NIPS'24)	10.0	2.0	42.3	64.6	46.2	38.8	61.6	41.4
EfficientVMamba-S (AAAI'25)	11.0	1.3	39.3	61.8	42.6	36.7	58.9	39.2
TinyViM-B (Ma et al., 2025) (ICCV'25)	11.0	1.5	42.3	64.2	46.3	38.7	61.1	41.3
LoopMamba-T-4 (Ours)	8.1	2.9	43.7	65.3	47.6	39.9	62.4	42.8
LoopMamba-T-8 (Ours)	8.5	4.4	**44.3**	**65.9**	**48.5**	**40.1**	**63.0**	**42.9**
LSNet-B (CVPR'25)	23.2	1.3	40.8	63.4	44.0	37.8	60.5	40.1
Swin-T (ICCV'21)	28.0	4.5	43.7	66.6	47.7	39.8	63.3	42.7
MSVMamba-M (Shi et al., 2024) (NIPS'24)	12.0	1.5	43.8	65.8	47.7	39.9	62.9	42.9
EfficientVMamba-B (AAAI'25)	33.0	4.0	43.7	66.2	47.9	40.2	63.3	42.9
MobileMamba-B1 (CVPR'25)	17.1	1.1	40.6	61.8	43.8	37.4	58.9	39.9
LoopMamba-B-4 (Ours)	17.8	6.4	45.5	67.2	50.0	41.4	64.3	44.7
LoopMamba-B-8 (Ours)	18.6	9.6	**46.1**	**67.8**	**50.2**	**41.9**	**64.9**	**45.1**

Table 3: Results of semantic segmentation on ADE20K. LoopMamba achieves competitive performance on semantic segmentation, demonstrating its effectiveness in downstream vision tasks.

Backbone	mIoU	P. (M)	F. (G)
LSNet-T (CVPR'25)	40.1	11.4	0.3
Vim-T (ICML'24)	41.0	7.0	1.5
FastVim-T (WACV'26)	41.8	7.0	1.2
LoopMamba-T-4 (Ours)	41.6	8.1	2.9
LoopMamba-T-8 (Ours)	**41.9**	8.5	4.4
LSNet-B (CVPR'25)	43.1	16.1	0.5
SWin-T (ICCV'21)	44.5	28.0	4.5
Vim-S (ICML'24)	44.9	26.0	5.1
MSVMamba-M (NIPS'24)	45.1	12.0	1.5
FastVim-S (WACV'26)	44.9	26.0	4.4
LoopMamba-B-4 (Ours)	44.7	17.8	6.4
LoopMamba-B-8 (Ours)	**45.8**	18.6	9.6

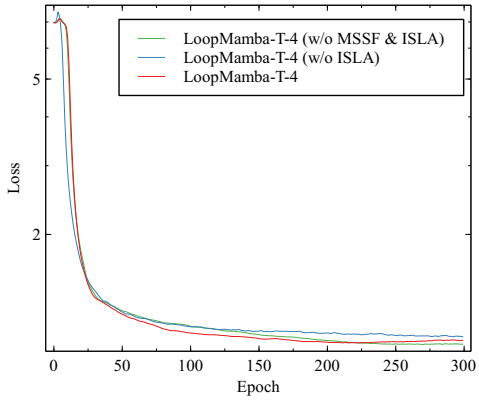

Figure 3: Training loss curves of different LoopMamba-T-4 variants on ImageNet-1K. Showing the effects of MSSF and ISLA.

At the tiny scale, LoopMamba-T-4 achieves 43.7% AP^b and 39.9% AP^m, while LoopMamba-T-8 further improves them to a best of 44.3% and 40.1%, respectively. LoopMamba-T-8 outperforms other methods such as QuadMamba-T by 2.0% in AP^b and 1.3% in AP^m with fewer parameters. At the larger scale, LoopMamba-B-4 and LoopMamba-B-8 achieve 45.5%/41.4% and 46.1%/41.9% in AP^b/AP^m, respectively, achieving the best performance. These consistent improvements indicate that repeated self-loop optimization progressively refines hidden representations, benefiting both object localization and instance-level feature discrimination, and also demonstrate LoopMamba's generalization ability.

4.3 Semantic Segmentation

Consistent with object detection and instance segmentation, we further evaluate LoopMamba for semantic segmentation on ADE20K (Zhou et al., 2017) using UPerNet (Xiao et al., 2018). As shown in Table 3, LoopMamba achieves competitive performance across various model scales. Specifically, LoopMamba-T-4 and LoopMamba-T-8 achieve 41.6% and 41.9% mIoU, respectively, with the latter outperforming FastVim-T by 0.1%, and achieving the highest performance. At the larger scale, LoopMamba-B-4 and LoopMamba-B-8 reach 44.7% and 45.8% mIoU, respectively, with LoopMamba-B-8 achieving the best performance among the compared methods. Moreover, increasing the loop depth from 4 to 8 consistently improves mIoU by 0.3% and 1.1% for the T and B variants, respectively. Overall, LoopMamba consistently achieves strong performance across different scales, demonstrating effective generalization and the effectiveness of iterative state refinement to dense prediction tasks.

Table 4: Ablation study of Variable-Parameter State Transition on ImageNet-1K. MSSF: multi-step state fusion. ISLA: iteration-specific low-rank adaptation.

LoopMamba-T-4				
MSSF	ISLA	P. (M)	F. (G)	Top-1 (%)
×	×	7.7	2.78	78.8
✓	×	7.8	2.86	79.0 (+0.2)
✓	✓	8.1	2.91	79.7 (+0.9)

Table 5: Depth analysis of LoopMamba. The results demonstrate that increasing the loop depth consistently improves accuracy.

Depth	P. (M)	F. (G)	Top-1 (%)
2	7.9	2.1	78.8
4	8.1	2.9	79.7
6	8.3	3.6	79.8
8	8.5	4.4	80.2
4 (w/o ISLA)	7.8	2.8	79.0
8 (w/o ISLA)	7.8	4.3	79.3

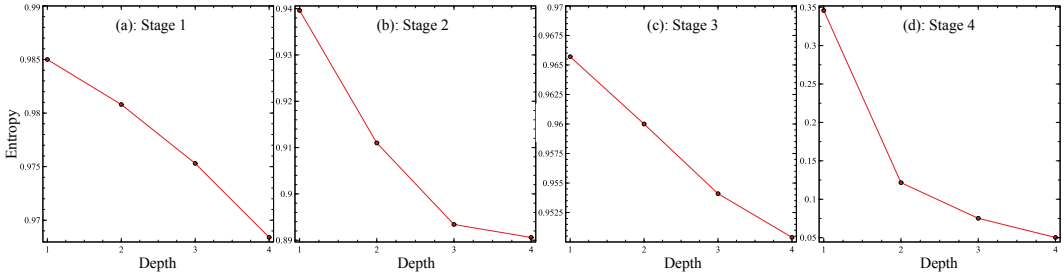

Figure 4: Entropy curves of different stages of LoopMamba-T-4 on ImageNet-1K. Providing empirical evidence of progressive hidden state refinement from an energy-based perspective.

4.4 Ablation Study

To investigate the effectiveness of each component in LoopMamba, we conduct ablation experiments on ImageNet-1K using LoopMamba-T-4. Specifically, we evaluate Multi-Step State Fusion (MSSF) and Iteration-Specific Low-rank Adaptation (ISLA), which together form the Iteration-specific Lightweight State Fusion (ILSF) mechanism. As shown in Table 4, the baseline achieves 78.8% Top-1 accuracy with 7.7M parameters and 2.78G FLOPs. Adding MSSF improves the accuracy to 79.0%, while further incorporating ISLA yields 79.7%, a 0.9% gain over the baseline with marginal computational overhead. We further visualize the corresponding training loss in Fig. 3. The complete model achieves lower training loss than variants without ISLA, indicating that iteration-specific adaptation improves the flexibility of state updates. Notably, the baseline exhibits comparable training loss but inferior accuracy, suggesting that the shared parameter self-loop state transition strategy without ILSF may overfit training samples, while MSSF and ISLA dynamically refine state representations by aggregating diverse state information and adapting low-rank modulation across iterations. Overall, these results and observations demonstrate that ILSF facilitates a more effective optimization trajectory and improves the representation learning capability of LoopMamba.

4.5 Depth Analysis

We further investigate the impact of loop depth on LoopMamba, with results shown in Table 5. Specifically, increasing the loop depth from 2 to 8 consistently improves Top-1 accuracy from 78.8% to 80.2%, while introducing only marginal parameter growth from 7.9M to 8.5M but increasing FLOPs from 2.1G to 4.4G. Notably, increasing the depth from 2 to 4 yields a 0.9% accuracy gain with only 0.2M additional parameters, demonstrating that additional state transitions progressively refine latent representations. This observation is consistent with LoopMamba's loop-based design, where greater loop depth adds state transition steps with marginal parameter growth but increased computational cost. To isolate the effect of repeated state transitions, we further evaluate variants without ISLA. Increasing the depth from 4 to 8 improves accuracy from 79.0% to 79.3% with unchanged parameters, confirming that repeated state transitions alone progressively refine visual representations, while the absence of ISLA limits their adaptive capability, resulting in limited improve of accuracy. Overall, these results demonstrate that increasing the loop depth provides a consistent accuracy improvement by trading additional computation for iterative representation refinement.

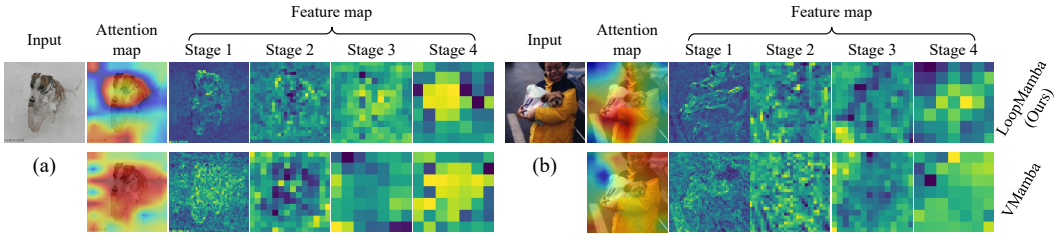

Figure 5: Visualization of LoopMamba-T-4 and VMamba-T on ImageNet-1K. Demonstrates LoopMamba's ability to maintain focus on important semantic regions.

4.6 State Evolution Analysis

To further investigate the optimization dynamics of LoopMamba from the energy perspective, we analyze the entropy evolution of intermediate hidden states. As illustrated in Fig. 4, the entropy consistently decreases with increasing loop depth across all four stages of LoopMamba-T-4 on ImageNet-1K, indicating progressive refinement of the latent representations. The reduction is relatively modest in early stages, with entropy decreasing from 0.985 to 0.968 in Stage 1 and from 0.939 to 0.890 in Stage 2, but becomes substantially larger in Stage 4, decreasing from 0.345 to 0.050. This stronger entropy reduction in deeper stages suggests that later semantic representations undergo more pronounced optimization and compression, consistent with their focus on extracting discriminative semantic patterns (stage 4 of LoopMamba in Fig. 5). Overall, the consistent entropy decay across loop iterations provides evidence that LoopMamba progressively refines hidden states through iterative state transitions, supporting its energy-based interpretation. We provide more entropy visualizations across LoopMamba variants in Appendix. D, which exhibit consistent trends.

4.7 Visualization

To intuitively evaluate LoopMamba's ability to learn effective visual representations, we visualize the attention maps of LoopMamba-T-4 and VMamba-T in Fig. 5 to compare their ability to capture crucial regions. LoopMamba produces more concentrated responses on the main semantic objects while suppressing irrelevant backgrounds. For example, it focuses on the dog in Fig. 5(a) and the main subject in the complex scene of Fig. 5(b), whereas VMamba exhibits more dispersed attention. In addition, we further visualize the feature maps across stages. LoopMamba consistently preserves key semantic regions, whereas VMamba shows scattered responses and loses crucial semantics at several stages. Specifically, in Fig. 5(a), LoopMamba maintains responses on the dog's body across stages, while VMamba exhibits scattered responses in stages 2 & 3. In the more complex case of Fig. 5(b), VMamba fails to preserve the semantic regions across stages, whereas LoopMamba retains meaningful responses at stages 1 & 4. These results demonstrate that LoopMamba can effectively capture and maintain focus on meaningful visual representations and regions.

5 Discussion and Conclusion

In this work, we introduce LoopMamba, which provides an energy-based perspective for understanding and designing SSM architectures. By interpreting state transitions as iterative energy minimization, LoopMamba reuses a shared state transition mechanism to progressively refine the hidden state with marginal parameter growth. Extensive experiments and analyses demonstrate the effectiveness of this iterative refinement across various vision tasks, while revealing a favorable trade-off between representation refinement and computational cost. Despite these advantages, the current design employs a fixed loop depth for all inputs, which may result in redundant computation for simple samples or insufficient refinement for complex ones. This limitation motivates future work on adaptive self-loop mechanisms that dynamically determine the depth of state transition steps based on input complexity or state convergence. Extending the self-loop paradigm to NLP and other sequence modeling tasks also provides a promising direction for exploring iterative state refinement beyond vision. More broadly, we believe that viewing state evolution as an iterative optimization process offers a promising framework for developing effective and interpretable SSMs across different domains. Further discussion on ACT and Looped Models is provided in Appendix E.

AI USE STATEMENT

In this work, we used generative AI tools to assist with improving the language and readability of the manuscript. We have not used generative AI tools to generate research ideas, design experiments, conduct experiments, analyze experimental results, generate mathematical derivations, or generate the scientific claims or conclusions of this work, and the remaining required disclosure tasks are not applicable to this work. We have reviewed and verified all AI-assisted work, including carefully reviewing all AI-assisted revisions to ensure their correctness, accuracy, and consistency with the intended content. We take full responsibility for the final content of this work, including all text produced with the aid of generative AI.

REPRODUCIBILITY STATEMENT

To facilitate the reproducibility of our work, we provide detailed supporting information in the appendix. Specifically, we provide detailed derivations of the theoretical results (Appendix. A), complete model architecture configurations (Appendix. B), and comprehensive experimental settings (Appendix. C). These materials are intended to provide sufficient information for reproducing and verifying our theoretical analyses and experimental results. Upon acceptance, we will release the complete source code and implementation of our work to facilitate further reproduction and research.

REFERENCES

Sangmin Bae, Yujin Kim, Reza Bayat, Sungnyun Kim, Jiyoun Ha, Tal Schuster, Adam Fisch, Hrayr Harutyunyan, Ziwei Ji, Aaron Courville, and Se-Young Yun. Mixture-of-recursions: Learning dynamic recursive depths for adaptive token-level computation. In D. Belgrave, C. Zhang, H. Lin, R. Pascanu, P. Koniusz, M. Ghassemi, and N. Chen (eds.), *Advances in Neural Information Processing Systems*, volume 38, Main Conference, pp. 96572–96617. Curran Associates, Inc., 2025. doi: 10.52202/085713-3229. URL https://proceedings.neurips.cc/paper_files/paper/2025/file/8b08bbf8b420faa6eeb4020720582ec7-Paper-Conference.pdf.

Mostafa Dehghani, Stephan Gouws, Oriol Vinyals, Jakob Uszkoreit, and Lukasz Kaiser. Universal transformers. In *International Conference on Learning Representations*, 2019. URL https://openreview.net/forum?id=HyzdRiR9Y7.

Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. Imagenet: A large-scale hierarchical image database. In *2009 IEEE conference on computer vision and pattern recognition*, pp. 248–255. Ieee, 2009.

Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, et al. An image is worth 16x16 words: Transformers for image recognition at scale. *International Conference on Learning Representations*, 2021.

Jiawei Du, Daquan Zhou, Jiashi Feng, Vincent Tan, and Joey Tianyi Zhou. Sharpness-aware training for free. *Advances in Neural Information Processing Systems*, 35:23439–23451, 2022.

Ying Fan, Yilun Du, Kannan Ramchandran, and Kangwook Lee. Looped transformers for length generalization. In Y. Yue, A. Garg, N. Peng, F. Sha, and R. Yu (eds.), *International Conference on Learning Representations*, volume 2025, pp. 14502–14520, 2025. URL https://proceedings.iclr.cc/paper_files/paper/2025/file/25cc3adf8c85f7c70989cb8a97a691a7-Paper-Conference.pdf.

Jonas Geiping, Sean McLeish, Neel Jain, John Kirchenbauer, Siddharth Singh, Brian Bartoldson, Bhavya Kailkhura, Abhinav Bhatele, and Tom Goldstein. Scaling up test-time compute with latent reasoning: A recurrent depth approach. In D. Belgrave, C. Zhang, H. Lin, R. Pascanu, P. Koniusz, M. Ghassemi, and N. Chen (eds.), *Advances in Neural Information Processing Systems*, volume 38, Main Conference, pp. 41340–41391. Curran Associates, Inc., 2025. doi: 10.52202/085713-1380. URL https://proceedings.neurips.cc/paper_files/paper/2025/file/3b01972cf31e6fa0fe29e4b8b5c2a0a1-Paper-Conference.pdf.

Cong Geng, Tian Han, Peng-Tao Jiang, Hao Zhang, Jinwei Chen, Søren Hauberg, and Bo Li. Improving adversarial energy-based model via diffusion process. In *Forty-first International Conference on Machine Learning*, 2024. URL https://openreview.net/forum?id=QXEx16jWdN.

Alexi Gladstone, Ganesh Nanduru, Md Mofijul Islam, Peixuan Han, Hyeonjeong Ha, Aman Chadha, Yilun Du, Heng Ji, Jundong Li, and Tariq Iqbal. Energy-based transformers are scalable learners and thinkers. In *International Conference on Learning Representations*, volume 2026, pp. 139312–139355, 2026.

Alex Graves. Adaptive computation time for recurrent neural networks. *arXiv preprint arXiv:1603.08983*, 2016.

Albert Gu and Tri Dao. Mamba: Linear-time sequence modeling with selective state spaces. In *First Conference on Language Modeling*, 2024. URL https://openreview.net/forum?id=tEYskw1VY2.

Albert Gu, Tri Dao, Stefano Ermon, Atri Rudra, and Christopher Ré. Hippo: Recurrent memory with optimal polynomial projections. *Advances in neural information processing systems*, 33: 1474–1487, 2020.

Albert Gu, Karan Goel, and Christopher Ré. Efficiently modeling long sequences with structured state spaces. *arXiv preprint arXiv:2111.00396*, 2021a.

Albert Gu, Isys Johnson, Karan Goel, Khaled Saab, Tri Dao, Atri Rudra, and Christopher Ré. Combining recurrent, convolutional, and continuous-time models with linear state space layers. *Advances in neural information processing systems*, 34:572–585, 2021b.

Albert Gu, Karan Goel, Ankit Gupta, and Christopher Ré. On the parameterization and initialization of diagonal state space models. *Advances in neural information processing systems*, 35:35971–35983, 2022.

Ali Hatamizadeh and Jan Kautz. Mambavision: A hybrid mamba-transformer vision backbone. In *Proceedings of the Computer Vision and Pattern Recognition Conference*, pp. 25261–25270, 2025.

Haoyang He, Jiangning Zhang, Yuxuan Cai, Hongxu Chen, Xiaobin Hu, Zhenye Gan, Yabiao Wang, Chengjie Wang, Yunsheng Wu, and Lei Xie. Mobilemamba: Lightweight multi-receptive visual mamba network. In *2025 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 4497–4507. IEEE, 2025.

Kaiming He, Georgia Gkioxari, Piotr Dollár, and Ross Girshick. Mask r-cnn. In *Proceedings of the IEEE international conference on computer vision*, pp. 2961–2969, 2017.

Md Mohaiminul Islam and Gedas Bertasius. Long movie clip classification with state-space video models. In *European Conference on Computer Vision*, pp. 87–104. Springer, 2022.

Md Mohaiminul Islam, Mahmudul Hasan, Kishan Shamsundar Athrey, Tony Braskich, and Gedas Bertasius. Efficient movie scene detection using state-space transformers. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 18749–18758, 2023.

Alexia Jolicoeur-Martineau. Less is more: Recursive reasoning with tiny networks. *arXiv preprint arXiv:2510.04871*, 2025.

Rudolph Emil Kalman. A new approach to linear filtering and prediction problems. 1960.

Saarthak Kapse, Robin Betz, and Srinivasan Sivanandan. Fast vision mamba: Pooling spatial dimensions for accelerated processing. In *2026 IEEE/CVF Winter Conference on Applications of Computer Vision (WACV)*, pp. 2923–2933. IEEE, 2026.

Yann LeCun, Sumit Chopra, Raia Hadsell, M Ranzato, Fujie Huang, et al. A tutorial on energy-based learning. *Predicting structured data*, 1(0), 2006.

Yann LeCun et al. A path towards autonomous machine intelligence version 0.9. 2, 2022-06-27. *Open Review*, 62(1):1–62, 2022.

Sanghyeok Lee, Joonmyung Choi, and Hyunwoo J Kim. Efficientvim: Efficient vision mamba with hidden state mixer based state space duality. In *2025 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 14923–14933. IEEE, 2025.

Tsung-Yi Lin, Michael Maire, Serge Belongie, James Hays, Pietro Perona, Deva Ramanan, Piotr Dollár, and C Lawrence Zitnick. Microsoft coco: Common objects in context. In *Computer vision–ECCV 2014: 13th European conference, zurich, Switzerland, September 6-12, 2014, proceedings, part v 13*, pp. 740–755. Springer, 2014.

Leiye Liu, Miao Zhang, Jihao Yin, Tingwei Liu, Wei Ji, Yongri Piao, and Huchuan Lu. Defmamba: Deformable visual state space model. In *2025 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 8838–8847. IEEE, 2025.

Yue Liu, Yunjie Tian, Yuzhong Zhao, Hongtian Yu, Lingxi Xie, Yaowei Wang, Qixiang Ye, Jianbin Jiao, and Yunfan Liu. Vmamba: Visual state space model. *Advances in neural information processing systems*, 37:103031–103063, 2024.

Ze Liu, Yutong Lin, Yue Cao, Han Hu, Yixuan Wei, Zheng Zhang, Stephen Lin, and Baining Guo. Swin transformer: Hierarchical vision transformer using shifted windows. In *2021 IEEE/CVF international conference on computer vision (ICCV)*, pp. 9992–10002. Ieee, 2021.

Zhuang Liu, Hanzi Mao, Chao-Yuan Wu, Christoph Feichtenhofer, Trevor Darrell, and Saining Xie. A convnet for the 2020s. In *2022 IEEE/CVF conference on computer vision and pattern recognition (CVPR)*, pp. 11966–11976. IEEE, 2022.

Ilya Loshchilov and Frank Hutter. Decoupled weight decay regularization. *International Conference on Learning Representations*, 2018.

Xiaowen Ma, Zhenliang Ni, and Xinghao Chen. Tinyvim: Frequency decoupling for tiny hybrid vision mamba. In *2025 IEEE/CVF International Conference on Computer Vision (ICCV)*, pp. 23519–23529. IEEE, 2025.

Nicola Muca Cirone, Antonio Orvieto, Benjamin Walker, Cristopher Salvi, and Terry Lyons. Theoretical foundations of deep selective state-space models. *Advances in Neural Information Processing Systems*, 37:127226–127272, 2024.

Xiaohuan Pei, Tao Huang, and Chang Xu. Efficientvmamba: Atrous selective scan for light weight visual mamba. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 39, pp. 6443–6451, 2025.

Nikunj Saunshi, Nishanth Dikkala, Zhiyuan Li, Sanjiv Kumar, and Sashank J. Reddi. Reasoning with latent thoughts: On the power of looped transformers. In Y. Yue, A. Garg, N. Peng, F. Sha, and R. Yu (eds.), *International Conference on Learning Representations*, volume 2025, pp. 14855–14881, 2025. URL https://proceedings.iclr.cc/paper_files/paper/2025/file/2676109d49d1eb26d6bc584a8f556305-Paper-Conference.pdf.

Yuheng Shi, Minjing Dong, and Chang Xu. Multi-scale vmamba: Hierarchy in hierarchy visual state space model. *Advances in Neural Information Processing Systems*, 37:25687–25708, 2024.

Jimmy TH Smith, Andrew Warrington, and Scott W Linderman. Simplified state space layers for sequence modeling. *International Conference on Learning Representations*, 2023.

Hugo Touvron, Matthieu Cord, Matthijs Douze, Francisco Massa, Alexandre Sablayrolles, and Hervé Jégou. Training data-efficient image transformers & distillation through attention. In *International conference on machine learning*, pp. 10347–10357. PMLR, 2021.

Ao Wang, Hui Chen, Zijia Lin, Jungong Han, and Guiguang Ding. Lsnet: See large, focus small. In *2025 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 9718–9729. IEEE, 2025a.

Feng Wang, Jiahao Wang, Sucheng Ren, Guoyizhe Wei, Jieru Mei, Wei Shao, Yuyin Zhou, Alan Yuille, and Cihang Xie. Mamba-reg: Vision mamba also needs registers. In *2025 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 14944–14953. IEEE, 2025b.

Guan Wang, Jin Li, Yuhao Sun, Xing Chen, Changling Liu, Yue Wu, Meng Lu, Sen Song, and Yasin Abbasi Yadkori. Hierarchical reasoning model, 2025c. URL https://arxiv.org/abs/2506.21734.

Jue Wang, Wentao Zhu, Pichao Wang, Xiang Yu, Linda Liu, Mohamed Omar, and Raffay Hamid. Selective structured state-spaces for long-form video understanding. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 6387–6397, 2023.

Chaodong Xiao, Minghan Li, Deyu Meng, Lei Zhang, et al. Spatial-mamba: Effective visual state space models via structure-aware state fusion. In *International Conference on Learning Representations*, volume 2025, pp. 73777–73795, 2025.

Tete Xiao, Yingcheng Liu, Bolei Zhou, Yuning Jiang, and Jian Sun. Unified perceptual parsing for scene understanding. In *European conference on computer vision*, pp. 432–448. Springer, 2018.

Yicheng Xiao, Lin Song, Shaoli Huang, Jiangshan Wang, Siyu Song, Yixiao Ge, Xiu Li, and Ying Shan. Mambatree: Tree topology is all you need in state space model. *Advances in Neural Information Processing Systems*, 37:75329–75354, 2024.

Fei Xie, Weijia Zhang, Zhongdao Wang, and Chao Ma. Quadmamba: Learning quadtree-based selective scan for visual state space model. *Advances in Neural Information Processing Systems*, 37:117682–117707, 2024.

Fei Xie, Zhongdao Wang, Weijia Zhang, and Chao Ma. Pvmamba: Parallelizing vision mamba via dynamic state aggregation. In *2025 IEEE/CVF International Conference on Computer Vision (ICCV)*, pp. 10218–10228. IEEE, 2025.

Kevin Xu and Issei Sato. On expressive power of looped transformers: Theoretical analysis and enhancement via timestep encoding. In Aarti Singh, Maryam Fazel, Daniel Hsu, Simon Lacoste-Julien, Felix Berkenkamp, Tegan Maharaj, Kiri Wagstaff, and Jerry Zhu (eds.), *Proceedings of the 42nd International Conference on Machine Learning*, volume 267 of *Proceedings of Machine Learning Research*, pp. 69613–69646. PMLR, 13–19 Jul 2025. URL https://proceedings.mlr.press/v267/xu25x.html.

Minkai Xu, Tomas Geffner, Karsten Kreis, Weili Nie, Yilun Xu, Jure Leskovec, Stefano Ermon, and Arash Vahdat. Energy-based diffusion language models for text generation. In *International Conference on Learning Representations*, volume 2025, pp. 33769–33789, 2025.

Chenhongyi Yang, Zehui Chen, Miguel Espinosa, Linus Ericsson, Zhenyu Wang, Jiaming Liu, and Elliot J Crowley. Plainmamba: Improving non-hierarchical mamba in visual recognition. *British Machine Vision Conference (BMVC)*, 2024a.

Liu Yang, Kangwook Lee, Robert Nowak, and Dimitris Papailiopoulos. Looped transformers are better at learning learning algorithms. In B. Kim, Y. Yue, S. Chaudhuri, K. Fragkiadaki, M. Khan, and Y. Sun (eds.), *International Conference on Learning Representations*, volume 2024, pp. 42195–42214, 2024b. URL https://proceedings.iclr.cc/paper_files/paper/2024/file/b8402301e7f06bdc97a31bfaa653dc32-Paper-Conference.pdf.

Seokju Yun and Youngmin Ro. Shvit: Single-head vision transformer with memory efficient macro design. In *2024 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 5756–5767. IEEE, 2024.

Jianrong Zhang, Hehe Fan, and Yi Yang. Energymogen: Compositional human motion generation with energy-based diffusion model in latent space. In *2025 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 17592–17602. IEEE, 2025.

Bolei Zhou, Hang Zhao, Xavier Puig, Sanja Fidler, Adela Barriuso, and Antonio Torralba. Scene parsing through ade20k dataset. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 633–641, 2017.

Lianghui Zhu, Bencheng Liao, Qian Zhang, Xinlong Wang, Wenyu Liu, and Xinggang Wang. Vision mamba: Efficient visual representation learning with bidirectional state space model. In *Forty-first International Conference on Machine Learning*, 2024.

In this appendix, we provide the following materials:

A: Detailed derivation of Theorem 1 (referring to Sec. 3.2 in the main paper);

B: Detailed architecture configurations of LoopMamba (referring to Sec. 3.3.1 in the main paper);

C: Detailed experimental settings (referring to Sec. 4 in the main paper).

D: Additional entropy visualization (referring to Sec. 4.6 in the main paper).

E: More discussion on the relationship between LoopMamba, ACT, and Looped Models, as well as future directions (referring to Sec. 5 in the main paper).

A Detailed Derivation of the Energy-based Interpretation of SSMs

In this section, we provide a detailed derivation and theoretical analysis of the energy-based interpretation of SSMs. The analysis is organized into three parts. First, we derive the quadratic energy formulation and establish its equivalence to the SSM state transition through a gradient descent step. Second, we verify the symmetry of matrix $\mathbf{M}$ for diagonal SSMs, thereby establishing the validity of the proposed energy formulation. Finally, we analyze the optimization properties of the resulting energy function, including its strict convexity, coercivity, and the existence and uniqueness of its global minimizer, and further verify these conditions for SSMs.

A.1 Derivation of the Energy Interpretation of SSMs

We start by establishing the connection between the SSM's state transition and the gradient descent process of EBMs. The recursive update of h_t in Eq. (3) can first be rewritten into the following residual form:

$$h_t = h_{t-1} + (\bar{\mathbf{A}} - \mathbf{I})h_{t-1} + \bar{\mathbf{B}}x_t. \tag{A-1}$$

Then, we rewrite the gradient descent step in Eq. (4), replacing the prediction y with the hidden state h:

$$h_t = h_{t-1} - \eta \nabla_{h_{t-1}} E_\theta(x_t, h_{t-1}). \tag{A-2}$$

Comparing Eq. (A-1) and Eq. (A-2), the two formulations become identical if:

$$\boxed{-\eta \nabla_{h_{t-1}} E_\theta(x_t, h_{t-1}) = (\bar{\mathbf{A}} - \mathbf{I})h_{t-1} + \bar{\mathbf{B}}x_t.} \tag{A-3}$$

Therefore, our objective is to construct an energy function whose gradient satisfies Eq. (A-3). For the convenience of discussion, we will ignore the subscripts of h and x below.

Since the desired gradient is linear with respect to the hidden state, we consider the following general quadratic energy function,

$$\boxed{E_\theta(x, h) = \frac{1}{2} h^\top \mathbf{M} h - b(x)^\top h,} \tag{A-4}$$

where $\mathbf{M} \in \mathbb{R}^{N \times N}$ is an undetermined symmetric matrix and $b(x) \in \mathbb{R}^N$ is an input-dependent mapping.

The gradient of Eq. (A-4) with respect to h is computed as:

$$\nabla_h E_\theta(x, h) = \nabla_h E^1_\theta(x, h) + \nabla_h E^2_\theta(x, h), \tag{A-5}$$

where $E^1_\theta(x, h) = \frac{1}{2} h^\top \mathbf{M} h$ and $E^2_\theta(x, h) = -b(x)^\top h$. For $\nabla_h E^1_\theta(x, h)$, using the $\frac{\partial (h^\top \mathbf{M} h)}{\partial h} = (\mathbf{M} + \mathbf{M}^\top)h$, we have:

$$\nabla_h E^1_\theta(x, h) = \frac{1}{2}(\mathbf{M} + \mathbf{M}^\top)h. \tag{A-6}$$

And for $E^2_\theta(x, h) = -b(x)^\top h$, it can be expanded as $E^2_\theta(x, h) = -\sum_{i=1}^N b_i(x) h_i$, where $b_i(x)$ denotes the i-th element of $b(x)$. Since $b(x)$ is independent of h, we have $\frac{\partial E^2_\theta(x,h)}{\partial h_i} = -b_i(x)$. Therefore,

$$\nabla_h E^2_\theta(x, h) = -b(x). \tag{A-7}$$

Based on Eq. (A-6) and Eq. (A-7), the gradient of Eq. (A-4) with respect to h is present as follow:

$$\nabla_h E_\theta(x,h) = \frac{1}{2}(\mathbf{M}+\mathbf{M}^\top)h - b(x), \tag{A-8}$$

Due to the $\mathbf{M}$ is symmetric, namely $\mathbf{M} = \mathbf{M}^\top$, then substituting Eq. (A-8) into Eq. (A-3) gives

$$-\eta\left(\mathbf{M}h - b(x)\right) = (\bar{\mathbf{A}} - \mathbf{I})h + \bar{\mathbf{B}}x. \tag{A-9}$$

Matching the coefficients of the hidden state and the input yields

$$\boxed{\mathbf{M} = \frac{1}{\eta}(\mathbf{I} - \bar{\mathbf{A}}), \qquad b(x) = \frac{1}{\eta}\bar{\mathbf{B}}x.} \tag{A-10}$$

Substituting Eq. (A-10) into Eq. (A-4) yields the following energy function,

$$E_\theta(x,h) = \frac{1}{2\eta}h^\top(\mathbf{I} - \bar{\mathbf{A}})h - \frac{1}{\eta}(\bar{\mathbf{B}}x)^\top h. \tag{A-11}$$

According to Eq. (A-8), its gradient is therefore

$$\nabla_h E_\theta(x,h) = \frac{1}{\eta}(\mathbf{I} - \bar{\mathbf{A}})h - \frac{1}{\eta}\bar{\mathbf{B}}x. \tag{A-12}$$

Finally, substituting Eq. (A-12) into the EBM gradient descent step in Eq. (A-2) gives

$$\begin{aligned} h_t &= h_{t-1} - (\mathbf{I} - \bar{\mathbf{A}})h_{t-1} + \bar{\mathbf{B}}x_t \\ &= \bar{\mathbf{A}}h_{t-1} + \bar{\mathbf{B}}x_t, \end{aligned} \tag{A-13}$$

which recovers the original state transition in Eq. (3).

A.2 Symmetry of the Matrix $\mathbf{M}$

The quadratic energy formulation (Eq. (A-4)) requires the matrix $\mathbf{M}$ to be symmetric. To verify that the energy interpretation is valid for modern SSMs, we analyze whether the matrix induced by the SSM transition satisfies this constraint.

Specifically, modern diagonal SSMs such as Mamba Gu & Dao (2024) parameterize the continuous-time transition matrix $\mathbf{A}$ as a diagonal matrix. After ZOH discretization, the discrete transition matrix $\bar{\mathbf{A}}$ preserves this diagonal structure:

$$\bar{\mathbf{A}} = \mathrm{diag}(\alpha_1, \alpha_2, \cdots, \alpha_N), \tag{A-14}$$

where α_i denotes the discretized state transition coefficient. Since a diagonal matrix is always symmetric, we have

$$\boxed{\bar{\mathbf{A}}^\top = \bar{\mathbf{A}}.} \tag{A-15}$$

Consequently,

$$(\mathbf{I} - \bar{\mathbf{A}})^\top = \mathbf{I} - \bar{\mathbf{A}}^\top = \mathbf{I} - \bar{\mathbf{A}}, \tag{A-16}$$

indicating that $\mathbf{I} - \bar{\mathbf{A}}$ is also symmetric. Therefore, the matrix $\mathbf{M}$ in Eq. (A-10) satisfies

$$\mathbf{M}^\top = \mathbf{M}, \tag{A-17}$$

which validates the symmetry constraint required by the quadratic energy function. Thus, the state transition of diagonal SSMs, including Mamba, can be interpreted as one gradient descent step over the corresponding energy landscape.

A.3 Convexity and Global Minimization of the Energy

We further analyze the optimization properties of the quadratic energy function associated with SSMs.

Consider the discrete state transition

$$h_t = \bar{\mathbf{A}} h_{t-1} + \bar{\mathbf{B}} x_t, \tag{A-18}$$

and its corresponding quadratic energy function

$$E_\theta(x, h) = \frac{1}{2\eta} h^\top (\mathbf{I} - \bar{\mathbf{A}}) h - \frac{1}{\eta} (\bar{\mathbf{B}} x)^\top h, \tag{A-19}$$

where $\eta > 0$ is the step size. Since $\mathbf{I} - \bar{\mathbf{A}}$ is symmetric, the Hessian of the energy with respect to the hidden state h is

$$\nabla_h^2 E_\theta(x, h) = \frac{1}{\eta} (\mathbf{I} - \bar{\mathbf{A}}). \tag{A-20}$$

Therefore, if

$$\mathbf{I} - \bar{\mathbf{A}} \succ 0, \tag{A-21}$$

then

$$\nabla_h^2 E_\theta(x, h) \succ 0, \tag{A-22}$$

which establishes that $E_\theta(x, h)$ is strictly convex with respect to h.

We next show that the energy function is coercive under the same condition. Let $\lambda_{\min} > 0$ denote the smallest eigenvalue of $\mathbf{I} - \bar{\mathbf{A}}$. By the positive definiteness of $\mathbf{I} - \bar{\mathbf{A}}$, we have

$$h^\top (\mathbf{I} - \bar{\mathbf{A}}) h \geq \lambda_{\min} \|h\|^2. \tag{A-23}$$

Meanwhile, by the Cauchy–Schwarz inequality,

$$\left| (\bar{\mathbf{B}} x)^\top h \right| \leq \|\bar{\mathbf{B}} x\| \|h\|. \tag{A-24}$$

Combining Eqs. (A-23) and (A-24, we obtain the lower bound

$$\boxed{E_\theta(x, h) \geq \frac{\lambda_{\min}}{2\eta} \|h\|^2 - \frac{\|\bar{\mathbf{B}} x\|}{\eta} \|h\|.} \tag{A-25}$$

Since the first term grows quadratically in $\|h\|$, whereas the second term grows only linearly, the quadratic term dominates as $\|h\| \to \infty$. More explicitly,

$$\frac{\lambda_{\min}}{2\eta} \|h\|^2 - \frac{\|\bar{\mathbf{B}} x\|}{\eta} \|h\| = \|h\| \left(\frac{\lambda_{\min}}{2\eta} \|h\| - \frac{\|\bar{\mathbf{B}} x\|}{\eta} \right) \to +\infty. \tag{A-26}$$

Therefore,

$$\boxed{E_\theta(x, h) \to +\infty, \qquad \text{as } \|h\| \to \infty,} \tag{A-27}$$

which establishes that $E_\theta(x, h)$ is coercive.

Since $E_\theta(x, h)$ is continuous and coercive, it attains a global minimum in the finite-dimensional hidden state space. Moreover, strict convexity guarantees that this global minimizer is unique. Setting the gradient to zero gives

$$\nabla_h E_\theta(x, h^*) = 0, \tag{A-28}$$

and therefore

$$h^* = (\mathbf{I} - \bar{\mathbf{A}})^{-1} \bar{\mathbf{B}} x. \tag{A-29}$$

The positive-definiteness condition in Eq. (A-21) can be verified for commonly used stable diagonal SSMs. Specifically, consider the continuous-time state transition matrix

$$\mathbf{A} = \operatorname{diag}(a_1, \ldots, a_N), \qquad a_i < 0. \tag{A-30}$$

Under zero-order hold (ZOH) discretization with a positive step size $\Delta > 0$, the discrete transition matrix is

$$\bar{\mathbf{A}} = \exp(\Delta \mathbf{A}) = \operatorname{diag}\left(e^{\Delta a_1}, \ldots, e^{\Delta a_N} \right). \tag{A-31}$$

Since $a_i < 0$ and $\Delta > 0$, we have

$$0 < e^{\Delta a_i} < 1, \qquad i = 1, \ldots, N. \tag{A-32}$$

Thus, all eigenvalues of $\bar{\mathbf{A}}$ lie strictly within $(0, 1)$, which directly implies

$$\mathbf{I} - \bar{\mathbf{A}} \succ 0. \tag{A-33}$$

Therefore, the quadratic energy function associated with such stable diagonal SSMs is strictly convex and admits a unique global minimizer.

This result provides a general optimization interpretation of SSM state transitions, showing that each transition performs a gradient descent step on a strictly convex energy landscape and drives the hidden state toward the unique minimizer in Eq. (A-29) under the positive-definiteness condition. For Mamba, the continuous-time state matrix remains diagonal with negative entries, while the discretization step $\Delta > 0$ depends on the current input. Consequently, $\bar{\mathbf{A}} = \exp(\Delta \mathbf{A})$ has diagonal entries in $(0, 1)$ at each time step, and thus $\mathbf{I} - \bar{\mathbf{A}} \succ 0$.

B Architecture Configurations

LoopMamba adopts a hierarchical four-stage architecture, with each stage performing iterative refinement through a Self-loop Layer and Variable-Parameter State Transition. We consider two model scales, Tiny (T) and Base (B). The Tiny variants use channel dimensions of $[64, 128, 256, 512]$, while the Base variants use $[96, 192, 384, 768]$. For each model scale, we evaluate two iteration settings, with 4 or 8 self-loop iterations applied within each stage. The detailed configurations are summarized in Table A-1.

Table A-1: Architecture configurations of LoopMamba variants.

Model	Channel Dimensions	Loop Depth (k)	Stages	Blocks
LoopMamba-T-4	$[64, 128, 256, 512]$	4	4	[1, 1, 1, 1]
LoopMamba-T-8	$[64, 128, 256, 512]$	8	4	[1, 1, 1, 1]
LoopMamba-B-4	$[96, 192, 384, 768]$	4	4	[1, 1, 1, 1]
LoopMamba-B-8	$[96, 192, 384, 768]$	8	4	[1, 1, 1, 1]

C Detailed Experimental settings

For the image classification task, we follow the training configuration of Swin Transformer (Liu et al., 2021). Specifically, all LoopMamba variants are trained from scratch for 300 epochs with a batch size of 128. We adopt AdamW (Loshchilov & Hutter, 2018) as the optimizer with betas set to (0.9, 0.999), momentum set to 0.9, an initial learning rate of 1×10^{-3}, and a weight decay of 0.05. A cosine annealing learning rate scheduler with a 20-epoch warm-up strategy is employed. Following previous works (Swin, VMamba (Liu et al., 2024)), we apply common data augmentation strategies during training. Moreover, label smoothing (0.1), exponential moving average, and MESA (Du et al., 2022) are adopted to improve training stability and performance. The DropPath rates are set to 0.2 for all variants.

For downstream tasks, we follow the experimental settings of Swin Transformer and employ the pretrained LoopMamba models as backbones for object detection, instance segmentation, and semantic segmentation tasks. Similar to the image classification setting, AdamW is adopted as the optimizer for downstream tasks. For object detection and instance segmentation, we set the training epochs to 12 with an initial learning rate of 1×10^{-4}. For semantic segmentation, the initial learning rate is set to 6×10^{-5}. All experiments are conducted on a computing cluster equipped with 8 NVIDIA L20 and A100 40GB GPUs.

D More Entropy Visulization

In this section, we provide additional visualizations of the entropy evolution of intermediate hidden states across LoopMamba variants with different scales. As shown in Fig. A-1, all LoopMamba

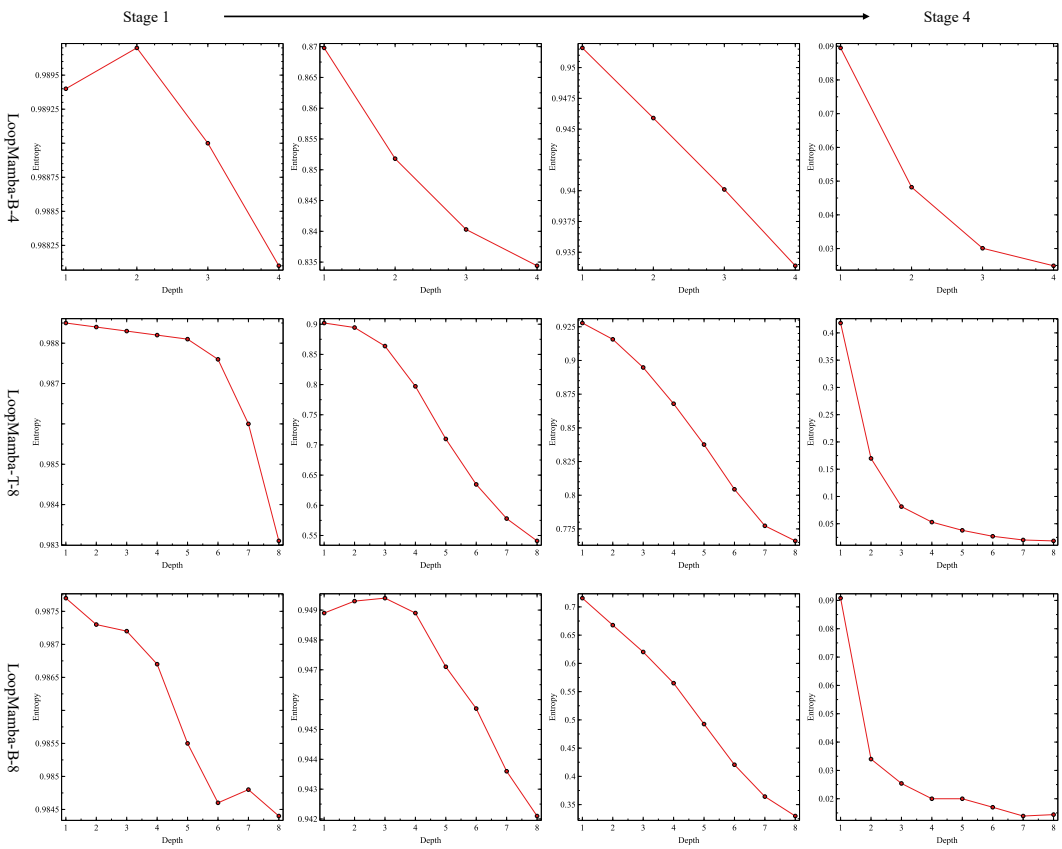

Figure A-1: More state entropy curves of LoopMamba variants. These results show consistent state evolution across different LoopMamba variants.

variants consistently exhibit a decreasing trend in state entropy across successive loop iterations. Although the magnitude of entropy reduction varies across model scales and stages, and intermediate states' entropy may exhibit slight fluctuations, the overall downward trend remains consistent, indicating that the hidden states undergo progressive refinement throughout the iterative state transitions. These observations are consistent with the entropy analysis of LoopMamba-T-4 in the main text and further support the iterative refinement behavior underlying the energy-based interpretation of LoopMamba.

E More Discussion

Relation to Adaptive Computation Time. LoopMamba is closely related to the broader idea of adaptive computation, but addresses a different aspect of iterative computation. Adaptive Computation Time (ACT) (Graves, 2016; Dehghani et al., 2019; Saunshi et al., 2025; Bae et al., 2025; Fan et al., 2025; Geiping et al., 2025; Wang et al., 2025c; Jolicoeur-Martineau, 2025) allows a recurrent model to dynamically determine the number of computational steps required for each input, thereby allocating more computation to samples that benefit from additional processing. Related recurrent architectures, such as the Universal Transformer (Dehghani et al., 2019), further demonstrate that recurrent depth can be combined with adaptive halting to obtain input-dependent computational depth.

In contrast, LoopMamba currently uses a fixed number of iterations and does not explicitly learn a halting policy. Its primary focus is not when to stop, but rather how the hidden state evolves during repeated transitions. Specifically, the proposed energy perspective interprets the state transition as a progressive refinement process, while VPST allows the transition dynamics to vary across iterations. Therefore, the two approaches are complementary: ACT provides a mechanism for determining

the computational budget, whereas LoopMamba provides a structured mechanism for performing computation within that budget. This distinction also suggests a natural extension of LoopMamba in which the number of self-loop iterations is dynamically determined according to the state evolution or its energy-related quantities.

Relation to Looped Models. LoopMamba is closely related to looped neural architectures, which reuse a compact set of parameters across multiple computational steps to increase the effective computational depth without proportionally increasing the number of parameters (Xu & Sato, 2025; Yang et al., 2024b; Saunshi et al., 2025; Fan et al., 2025). Such architectures demonstrate that computational depth can be decoupled from the number of parameterized layers, allowing a model to repeatedly refine its representations using a compact parameter set.

LoopMamba follows this principle by repeatedly applying the same Self-loop Layer, thereby increasing the effective depth through iterative state transitions while maintaining a small number of parameterized layers. However, LoopMamba does not rely on strict weight tying across iterations. Through Variable-Parameter State Transition (VPST), the transition dynamics can vary across iterations through iteration-specific parameters. Consequently, while the overall computational structure is shared across loop iterations, the state transition can progressively adapt along the refinement trajectory. This design provides a compromise between fully independent layers, where each computational step introduces a separate set of parameters, and strictly weight-tied looped models, where all iterations perform identical transformations.

From this perspective, LoopMamba extends the parameter-sharing principle of looped models to visual state-space models by introducing iteration-specific transition dynamics. This allows effective computational depth to be increased without linearly increasing the parameter count, while retaining the flexibility to specialize state transitions at different stages of the iterative refinement process.

Future Directions Inspired by Adaptive and Looped Computation. The connections to ACT and looped models suggest several natural directions for extending LoopMamba. First, the current architecture uses a fixed number of Self-loop iterations for all inputs. Inspired by ACT and looped architectures, future LoopMamba variants could further decouple the effective computational depth from the number of parameterized Self-loop Layers, allowing the same compact set of transition parameters to support a flexible number of iterations. In particular, an adaptive halting mechanism could determine the required number of state transitions for each input, allocating more computation to visually complex samples while avoiding unnecessary iterations for simpler ones. Such a design would extend LoopMamba from a fixed-depth iterative architecture to an input-adaptive computation framework, while preserving its parameter-efficient looped structure.

Second, the energy-based interpretation developed in this work provides a potential basis for determining when further state refinement is necessary. For example, the magnitude of state changes, the decrease of the associated energy, or the stabilization of hidden state statistics could serve as signals for adaptive halting. This could lead to an energy-guided LoopMamba that dynamically balances representation refinement and computational cost, providing a more principled alternative to manually specifying the loop depth.

More broadly, these directions raise an interesting question for future research: **how much computation should a visual state-space model perform, and how should that computation be allocated across iterative state transitions?** LoopMamba provides an initial answer by interpreting repeated SSM transitions as progressive hidden state refinement. Combining this perspective with adaptive computation and looped architectures may lead to visual SSMs that jointly optimize parameter sharing, computational depth, and input-dependent state refinement.