VISIONHOPE: VISUAL BACKBONES AS SELF-MODIFYING LEARNING SYSTEMS

Anonymous authors
Paper under double-blind review

ABSTRACT

Visual backbones have evolved from Convolutional Neural Networks (CNNs) with local aggregation to Vision Transformers (ViTs) with global interactions, State-Space Models (SSMs) with input-dependent state transitions, and Test-Time Training (TTT) layers that adapt an inner learner while processing an image. Across this progression, visual computation has become increasingly adaptive to each input, yet the rules governing that adaptation remain largely prescribed by the trained backbone. We introduce VisionHOPE, the first generic visual backbone formulated as a self-modifying learning system, in which what the model remembers and how it learns co-evolve within an image. Building on the self-referential construction of Nested Learning (NL), VisionHOPE realizes this co-evolution through five coupled memories that store content, generate key and value representations, and govern learning rate and retention. These memories evolve jointly as visual context accumulates along each scan. However, directly applying the unconstrained self-referential update to a visual backbone leads to instability. We therefore derive a stability-matched step-size control scheme that combines a soft cap on self-referential injection with a spectral clamp on the retained memory transition, and prove that the resulting memory dynamics are non-expansive along each scan. For two-dimensional feature maps, we adapt NL's chunk formulation by aligning chunks with image rows and columns across four directional scans. VisionHOPE achieves strong results on ImageNet-1K, COCO, and ADE20K, establishing self-modifying learning systems as a practical foundation for general-purpose visual backbones. The code is available at `https://anonymous.4open.science/r/ICLR_27_ID_25983`.

1 INTRODUCTION

Visual backbone design has repeatedly advanced by changing how information is aggregated and propagated across spatial locations. Convolutional Neural Networks (CNNs) use shared local kernels, introducing locality and translation equivariance (Krizhevsky et al., 2012; He et al., 2016), while modern ConvNets show that redesigned convolutional architectures remain highly competitive (Liu et al., 2022; Ding et al., 2024; Yu & Wang, 2025). Vision Transformers (ViTs) instead use content-dependent self-attention, allowing each token to interact directly with global visual context (Vaswani et al., 2017; Dosovitskiy et al., 2020). Subsequent work has introduced hierarchical architectures, refined spatial aggregation, improved attention efficiency, and revisited ViT block design (Liu et al., 2021; Fan et al., 2024; Xu et al., 2025; Wang et al., 2026). However, self-attention has quadratic complexity in token count, making high-resolution processing increasingly costly.

State-Space Models (SSMs) provide a recurrent alternative (Gu et al., 2021). Mamba makes its state transition input-dependent, allowing it to selectively propagate or forget information with linear sequence-length scaling (Gu & Dao, 2023). Visual adaptations develop scanning strategies to accommodate the two-dimensional structure of images (Zhu et al., 2024; Liu et al., 2024; Huang et al., 2024) and incorporate attention modules to improve performance (Hatamizadeh & Kautz, 2025). Together, these advances establish SSMs as a strong family of visual backbones. Nevertheless, their transition-generation rule remains fixed: each token determines a transition, but the token-to-transition mapping itself does not evolve within the image. Test-Time Training (TTT) goes further by using an inner learner as its recurrent state and updating it through self-supervised learning (Sun et al., 2025). In vision, ViT^3 studies full-image inner adaptation (Han et al., 2026), whereas

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.

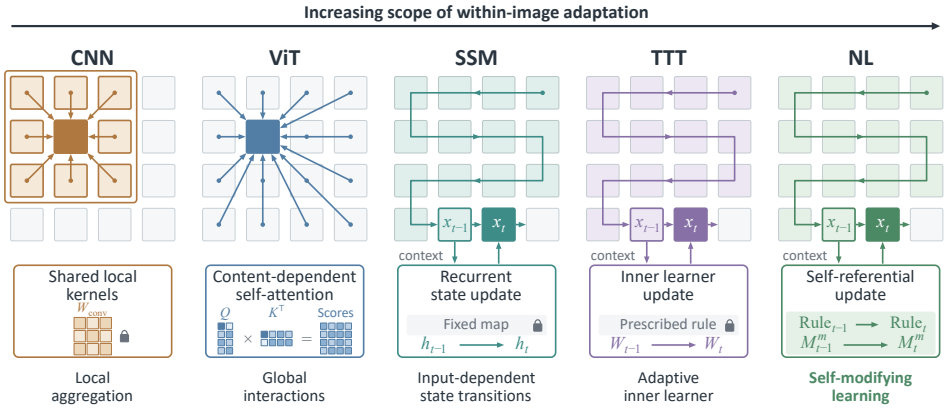

Figure 1: Progression of within-image adaptation from local aggregation to self-modifying learning.

Vision-TTT adapts the learner recurrently along visual token sequences (Kong et al., 2026). These methods show that TTT layers offer an alternative to attention and conventional recurrent layers.

Figure 1 summarizes this progression: from CNNs to TTT, computation becomes increasingly adaptive to each input, yet the rules governing that adaptation remain prescribed by the trained backbone. This raises a question: **Can a visual backbone modify not only what it remembers, but also how it learns while processing an image?** We address this question through the self-referential principle of Nested Learning (NL) (Behrouz et al., 2025a). NL describes a model as a collection of interconnected learning processes, each compressing its context flow into an internal state. Its self-referential construction couples memories that store content, generate the key and value representations used for updates, and govern learning rate and retention. From this perspective, a backbone can become a learning system whose stored content and learning rule co-evolve with accumulating visual context.

Based on this principle, we introduce VisionHOPE, the first generic visual backbone formulated as a self-modifying learning system. VisionHOPE couples five types of within-image memory: content, key, value, learning rate, and retention. As visual context accumulates along each scan, the content memory co-evolves with the memories that determine its updates. VisionHOPE therefore modifies not only what it remembers but also how it learns. This distinguishes it from TTT-based visual backbones, which adapt an inner learner while leaving its representation maps and update rule largely outer-parameterized. Directly applying NL's unconstrained self-referential update to a visual backbone, however, creates a stability barrier. Under this self-referential construction, the memories generate the quantities governing their own subsequent updates, creating a feedback loop in which expansive memory transitions can compound and destabilize training. We derive a stability-matched step-size control scheme that combines a soft cap on self-referential injection with a spectral clamp on the retained memory transition. We prove that these controls yield complementary bounds that jointly guarantee non-expansive memory dynamics along each scan. To process two-dimensional feature maps, VisionHOPE adapts NL's chunk formulation (Behrouz et al., 2025a) to four directional scans (Liu et al., 2024) by aligning chunks with image rows and columns. Forward and reverse row-major scans use row-aligned chunks, while forward and reverse column-major scans use column-aligned chunks. Each direction maintains independent states, and the directional outputs are fused channel-wise. The recurrent computation scales linearly with visual token count, while the chunk formulation facilitates parallel computation of token-dependent quantities. Our contributions are:

1. We introduce VisionHOPE, the first generic visual backbone formulated as a self-modifying learning system. Its content, key, value, learning-rate, and retention memories co-evolve along visual scans, allowing the backbone to modify both what it remembers and how it learns.
2. We identify the unconstrained self-referential update as a stability barrier in vision and derive a stability-matched step-size control scheme. Combining a soft cap on self-referential injection with a spectral clamp on the retained transition, it guarantees non-expansive memory dynamics.
3. We adapt NL's chunk formulation to four directional scans and instantiate the resulting VisionHOPE operator in hierarchical and plain backbones. Across model scales, VisionHOPE

achieves competitive results on ImageNet-1K (Deng et al., 2009), COCO (Lin et al., 2014), and ADE20K (Zhou et al., 2017), demonstrating its potential as a general-purpose visual backbone.

2 METHODOLOGY

VisionHOPE instantiates NL's self-referential principle as a stabilized Self-Referential Nested Learning (SRNL) module. We first review the theoretical foundations of SRNL, then introduce the stability-matched step-size control and prove non-expansion for the resulting token-wise and chunk-wise recurrences. Finally, we apply independent SRNL instances along four directional scans with spatially aligned chunks and embed the resulting VisionHOPE operator in a standard residual block.

2.1 NESTED LEARNING FOUNDATIONS

2.1.1 DELTA GRADIENT DESCENT

Following NL (Behrouz et al., 2025a), consider a linear associative memory with shared key and value dimension d. At online step t of the context flow, its preceding memory state $W_{t-1} \in \mathbb{R}^{d\times d}$ maps the current key $k_t \in \mathbb{R}^d$ to the response $z_t = W_{t-1}k_t \in \mathbb{R}^d$. Let $\mathcal{J}_t : \mathbb{R}^d \to \mathbb{R}$ be a differentiable local objective on a response $z \in \mathbb{R}^d$. Its gradient with respect to z at z_t defines the output-space learning signal $g_t = \nabla_z \mathcal{J}_t(z)|_{z=z_t} \in \mathbb{R}^d$. With learning rate $\eta_t > 0$, ordinary gradient descent produces the updated memory $W_t^{\mathrm{GD}} = W_{t-1} - \eta_t g_t k_t^\top$. Here, k_t specifies the write address and $-g_t$ provides the write signal. Appendix B.1 explains the associative-memory formulation.

NL introduces Delta Gradient Descent (DGD) to incorporate an explicit correction based on the memory response at the current key. It uses the write signal $u_t = -g_t$ as the target of an auxiliary L_2 regression objective. We denote the proximal parameter by $\lambda_t > 0$ to distinguish it from the effective step η_t used in the recurrence. With k_t and u_t held fixed, the updated memory W_t^{DGD} is

$$W_t^{\mathrm{DGD}} = \arg\min_{W\in\mathbb{R}^{d\times d}} \left[\frac{1}{2}\left\|Wk_t - u_t\right\|_2^2 + \frac{1}{2\lambda_t}\left\|W - W_{t-1}\right\|_F^2\right], \qquad \lambda_t > 0. \tag{1}$$

Here, W is a candidate memory. The first term measures the squared regression error between the response Wk_t and the write target u_t at the current key, while the second penalizes changes from the previous state W_{t-1}. Solving this proximal problem gives the effective DGD step size

$$\eta_t = \frac{\lambda_t}{1 + \lambda_t\|k_t\|_2^2}, \tag{2}$$

which reduces to $\eta_t = \lambda_t/(1+\lambda_t)$ for unit-norm keys. The resulting closed-form recurrence is

$$W_t^{\mathrm{DGD}} = W_{t-1} - \eta_t\left(W_{t-1}k_t - u_t\right)k_t^\top = W_{t-1}\left(I_d - \eta_t k_t k_t^\top\right) - \eta_t g_t k_t^\top. \tag{3}$$

Here, $I_d \in \mathbb{R}^{d\times d}$ is the identity matrix. Appendix B.2 provides the derivation. Compared with ordinary gradient descent, DGD adds the key-dependent correction $-\eta_t W_{t-1}k_t k_t^\top$. Equivalently, it writes the residual $u_t - W_{t-1}k_t$, correcting the write signal using what the memory stores at the current key. This provides a mechanism for revising existing associations as successive, correlated context elements arrive. The self-referential construction of NL augments this DGD recurrence with a context-dependent retention factor denoted by $\alpha_t \in [0, 1)$, yielding the updated memory state W_t:

$$W_t = W_{t-1}\left(\alpha_t I_d - \eta_t k_t k_t^\top\right) - \eta_t g_t k_t^\top. \tag{4}$$

Here, α_t controls how much of the previous memory is retained at each step of the recurrence, and η_t scales both the key-dependent correction and the current gradient write. This retained DGD recurrence is the inner learning rule applied to the coupled memories introduced next.

2.1.2 SELF-REFERENTIAL NESTED LEARNING

The retained DGD recurrence in Equation (4) specifies how a memory is updated at each online step given its current key, learning signal, step size, and retention factor. NL makes this process self-referential by using evolving memories to generate the update quantities and allowing each memory to produce its own target. NL permits arbitrary memory architectures. In the original HOPE

architecture in NL, these memories are instantiated as Multi-Layer Perceptrons (MLPs), and their mappings are learned with an L_2 regression objective. The corresponding L_2 recurrence, however, is derived explicitly only for matrix-valued linear memories. VisionHOPE accordingly adopts this linear L_2 recurrence for memory updates, yielding the five-memory SRNL module used throughout the model. For an ordered visual context, let $x_t \in \mathbb{R}^d$ denote the token at step t, where d is the input dimension of an SRNL instance. The self-referential state contains the following five memories:

$$\mathcal{S}_t = \left\{ M_t^m, M_t^k, M_t^v, m_t^\eta, m_t^\alpha \right\}. \tag{5}$$

Here, $M_t^m \in \mathbb{R}^{d\times d}$ stores content, $M_t^k, M_t^v \in \mathbb{R}^{d\times d}$ generate key and value representations, and $m_t^\eta, m_t^\alpha \in \mathbb{R}^{1\times d}$ govern learning rate and retention. For compactness, $M_t^\square$ denotes any of these states for $\square \in \{m, k, v, \eta, \alpha\}$, with $M_t^\eta = m_t^\eta$ and $M_t^\alpha = m_t^\alpha$. At step t, the update quantities are generated from the preceding memory states using the current token x_t:

$$k_t = M_{t-1}^k x_t, \qquad v_t = M_{t-1}^v x_t, \qquad \eta_t = \phi_\eta\left(m_{t-1}^\eta x_t\right), \qquad \alpha_t = \phi_\alpha\left(m_{t-1}^\alpha x_t\right). \tag{6}$$

Here, $k_t, v_t \in \mathbb{R}^d$ are the key and value representations. The functions $\phi_\eta : \mathbb{R} \to \mathbb{R}_{>0}$ and $\phi_\alpha : \mathbb{R} \to [0, 1)$ map the scalar memory outputs to a positive DGD step η_t and a retention factor α_t. Their forms are specified in Appendix C.1. The learning-rate memory directly parameterizes η_t, while λ_t is used only in the preceding proximal derivation. Self-reference enters through each memory's update target. Rather than sharing v_t as the target, memory $\square$ transforms it using its current state:

$$\widehat{v}_t^\square = M_{t-1}^\square v_t, \qquad \square \in \{m, k, v, \eta, \alpha\}. \tag{7}$$

Each memory uses its self-generated target $\widehat{v}_t^\square$ to define a learning signal at the current key k_t. Following the output-space formulation in Section 2.1.1, we define the local L_2 regression objective

$$\mathcal{J}_t^\square(z) = \frac{1}{2}\left\| z - \widehat{v}_t^\square \right\|_2^2, \tag{8}$$

where z denotes the memory response. Both the response and target are vectors for matrix memories and scalars for row-vector memories. Holding the target fixed and evaluating the gradient with respect to z at the current response gives the output-space learning signal for each memory

$$g_t^\square = \nabla_z \mathcal{J}_t^\square(z)\Big|_{z=M_{t-1}^\square k_t} = M_{t-1}^\square k_t - \widehat{v}_t^\square. \tag{9}$$

Substituting this signal into retained DGD gives the Self-Referential DGD (SR-DGD) update

$$M_t^\square = M_{t-1}^\square\left(\alpha_t I_d - \eta_t k_t k_t^\top\right) - \eta_t g_t^\square k_t^\top. \tag{10}$$

The same expression applies to row-vector memories as $1 \times d$ linear maps. Because all five states generate quantities used to determine their subsequent updates, stored content, update representations, and learning dynamics evolve as a coupled system. The recurrence is explicit and differentiable, so outer gradients can pass through the memory trajectory without an iterative inner solver.

The query projection remains outside the self-referential update loop. Following NL's construction, an outer-parameterized projection generates the query to read the output from the content memory:

$$q_t = W_q x_t, \qquad y_t = M_{t-1}^m q_t. \tag{11}$$

Here, $W_q \in \mathbb{R}^{d\times d}$ is the query projection, while $q_t, y_t \in \mathbb{R}^d$ are the query and output. The projection is learned through outer training but remains fixed during within-context memory evolution.

2.1.3 Chunk-Wise Linear Recurrence

The fully token-indexed SRNL recurrence in Equation (10) must update its state at step t before generating the quantities for step $t+1$, serializing the state updates and the five memory mappings across tokens. NL reduces this cost through chunking. Within each chunk, fixed boundary states generate token-dependent quantities in parallel, while the SR-DGD state updates accumulate in token order. Chunking refreshes the states used to generate update quantities only at chunk boundaries. We partition the ordered context into chunks of fixed length C. For token t, the state of memory $\square \in \{m, k, v, \eta, \alpha\}$ at the beginning of its chunk is $M_{C\times\lfloor\frac{t-1}{C}\rfloor}^\square$. These boundary states generate

$$k_t = M_{C\times\lfloor\frac{t-1}{C}\rfloor}^k x_t, \quad v_t = M_{C\times\lfloor\frac{t-1}{C}\rfloor}^v x_t, \quad \eta_t = \phi_\eta\left(m_{C\times\lfloor\frac{t-1}{C}\rfloor}^\eta x_t\right), \quad \alpha_t = \phi_\alpha\left(m_{C\times\lfloor\frac{t-1}{C}\rfloor}^\alpha x_t\right). \tag{12}$$

With $q_t = W_q x_t$, the content readout, self-generated target, and output-space learning signal are

$$y_t = M_{C\times\lfloor\frac{t-1}{C}\rfloor}^m q_t, \qquad \widehat{v}_t^\square = M_{C\times\lfloor\frac{t-1}{C}\rfloor}^\square v_t, \qquad g_t^\square = M_{C\times\lfloor\frac{t-1}{C}\rfloor}^\square k_t - \widehat{v}_t^\square. \tag{13}$$

2.2 STABILITY-MATCHED STEP-SIZE CONTROL

The unconstrained SR-DGD update couples every memory to quantities generated by the evolving system itself within each image. To isolate the resulting feedback, we first analyze the fully token-indexed recurrence. The resulting step-size control is applied at each token and has the same form under chunking. Let $\delta_t = k_t - v_t \in \mathbb{R}^d$ denote the key-value discrepancy. Equations (7) and (9) then give $g_t^{\square} = M_{t-1}^{\square}\delta_t$. Substituting this relation into Equation (10) yields the recurrence:

$$M_t^{\square} = M_{t-1}^{\square}\left(\alpha_t I_d - \eta_t k_t k_t^\top - \eta_t \delta_t k_t^\top\right). \tag{14}$$

The matrix multiplying $M_{t-1}^{\square}$ is the one-step memory transition. For $k_t \neq 0$, evaluating this transition along the unit direction of k_t and applying the reverse triangle inequality gives

$$\left\|\alpha_t I_d - \eta_t k_t k_t^\top - \eta_t \delta_t k_t^\top\right\|_2 = \left\|\alpha_t I_d - \eta_t (k_t + \delta_t) k_t^\top\right\|_2 \geq \eta_t \|k_t\|_2 \|k_t + \delta_t\|_2 - \alpha_t. \tag{15}$$

Thus, $\eta_t\|k_t\|_2\|k_t + \delta_t\|_2 > 1 + \alpha_t$ is sufficient for an expansive transition, and the recurrence provides no general non-expansion guarantee. Because the updated memories generate the quantities governing subsequent updates, such amplification can feed back and compound along the context. Under chunking, the same feedback passes through the boundary states propagated between chunks. To guarantee non-expansion, we bound the spectral norm of the retained memory transition $\alpha_t I_d - \eta_t k_t k_t^\top$ by α_t and keep that of the self-referential injection $-\eta_t \delta_t k_t^\top$ below $1 - \alpha_t$. We implement these complementary constraints with a soft injection cap followed by a spectral clamp.

Definition 1 (Stability-matched step-size control)**.** *VisionHOPE applies stabilized L_2 normalization to each key, ensuring $\|k_t\|_2 \leq 1$. Let $\epsilon > 0$ be a constant for numerical stability. For the raw step $\eta_t > 0$ and retention factor $\alpha_t \in [0, 1)$ generated by their memories, define the soft injection cap:*

$$r_t = \sqrt{\|\delta_t\|_2^2 + \epsilon^2} + \epsilon, \qquad \eta_t^{\mathrm{inj}} = \frac{1 - \alpha_t}{r_t}, \qquad \bar{\eta}_t = \eta_t^{\mathrm{inj}}\left(1 - \exp\left(-\frac{\eta_t}{\eta_t^{\mathrm{inj}}}\right)\right), \tag{16}$$

where r_t is the stabilized magnitude of the key-value discrepancy, η_t^{inj} is the step-size limit for self-referential injection, and $\bar{\eta}_t$ is the smoothly capped candidate step. We then define the spectral clamp, which enforces the retained-transition bound and produces the final executed step $\widetilde{\eta}_t$:

$$\eta_t^{\mathrm{spec}} = \frac{2\alpha_t}{\|k_t\|_2^2}, \qquad \widetilde{\eta}_t = \min(\bar{\eta}_t, \eta_t^{\mathrm{spec}}), \tag{17}$$

where η_t^{spec} is the spectral limit for the retained memory transition and is set to $+\infty$ when $k_t = 0$. The soft injection cap smoothly limits self-referential injection while preserving small raw steps to first order. The spectral clamp leaves the candidate unchanged unless it exceeds η_t^{spec}.

Replacing η_t in Equation (14) with $\widetilde{\eta}_t$ gives

$$M_t^{\square} = M_{t-1}^{\square} T_t, \qquad T_t = A_t + B_t, \qquad A_t = \alpha_t I_d - \widetilde{\eta}_t k_t k_t^\top, \qquad B_t = -\widetilde{\eta}_t \delta_t k_t^\top. \tag{18}$$

Here, A_t is the retained memory transition, B_t is the self-referential injection operator, and T_t is the complete controlled transition governing the memory update at each token.

Proposition 1 (Complementary operator bounds)**.** *Under Definition 1, the two transition components satisfy the following complementary operator bounds:*

$$\|A_t\|_2 \leq \alpha_t, \qquad \|B_t\|_2 < 1 - \alpha_t. \tag{19}$$

Appendix B.3 provides the proof. The first bound limits the spectral norm of the retained memory transition to α_t. The second scales the admissible self-referential injection with the contraction margin left by retention, allowing less additional feedback as α_t approaches one. Together, these bounds control both potential sources of one-step amplification in the complete transition.

Corollary 1 (Token-wise non-expansion)**.** *For the fully token-indexed recurrence, the following transition and state bounds hold for every memory $\square \in \{m, k, v, \eta, \alpha\}$ and token t:*

$$\|T_t\|_2 < 1, \qquad \|M_t^{\square}\|_F \leq \|M_{t-1}^{\square}\|_F. \tag{20}$$

The corollary establishes non-expansion of memory norms along the token-wise recurrence and is proved in Appendix B.3. The same step-size control gives the complementary operator bounds at each token for quantities generated from chunk boundary states. Appendix B.4 derives a shared-gain representation and proves non-expansion relative to the boundary states of each chunk.

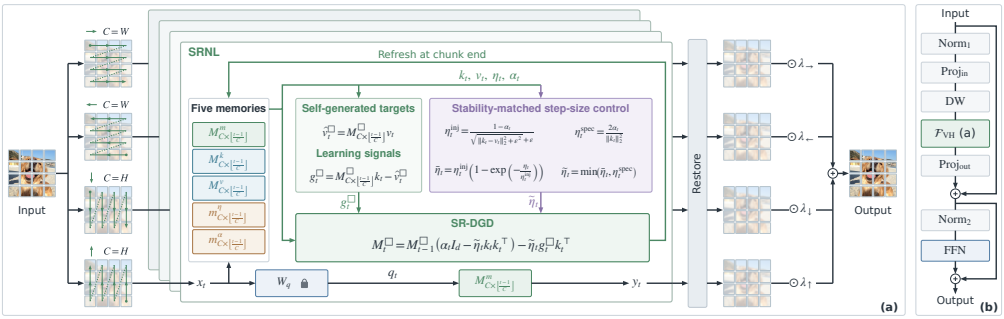

Figure 2: (a) The VisionHOPE operator applies SRNL along four directional scans with spatially aligned chunks, followed by spatial restoration and channel-wise fusion. Five coupled memories evolve through SR-DGD with step-size control. (b) The VisionHOPE block integrates the operator with projections, depthwise convolution, and an FFN in two pre-normalized residual branches.

2.3 VisionHOPE Architecture

Four-directional VisionHOPE operator. Sections 2.1 and 2.2 define the SRNL module over an ordered visual context and equip its recurrence with the stability-matched step-size control. The VisionHOPE operator extends SRNL to two-dimensional feature maps through four directional scans (Liu et al., 2024), followed by spatial restoration and channel-wise fusion (Figure 2(a)). Along each route, chunk boundaries are aligned with complete rows or columns. Let $Z \in \mathbb{R}^{B\times D_m\times H\times W}$ denote the internal feature map for scanning, where B is the batch size, D_m is the internal width, and H and W are the spatial dimensions. With $N = HW$, VisionHOPE serializes Z as

$$\mathcal{D} = \{\rightarrow, \leftarrow, \downarrow, \uparrow\}, \qquad Z_r = \mathcal{P}_r(Z) \in \mathbb{R}^{B\times N\times D_m}, \qquad r \in \mathcal{D}, \tag{21}$$

where $\mathcal{P}_r$ serializes the feature grid in either direction under row-major or column-major ordering. Each direction runs an independent SRNL instance with the five-memory state defined in Equation (5), initialized from learned values at the beginning of each image. Chunks follow the rows or columns traversed by each route. Let C_r denote the chunk length for direction r. VisionHOPE sets

$$C_\rightarrow = C_\leftarrow = W, \qquad C_\downarrow = C_\uparrow = H. \tag{22}$$

The row-major routes contain H row-aligned chunks, while the column-major routes contain W column-aligned chunks. Let $Y_r \in \mathbb{R}^{B\times N\times D_m}$ denote the directional output for route r. The four sequences are restored to the common feature grid and fused using learned channel-wise scales:

$$Y = \sum_{r\in\mathcal{D}} \lambda_r \odot \mathcal{P}_r^{-1}(Y_r), \qquad \lambda_r \in \mathbb{R}^{D_m}, \tag{23}$$

where $\mathcal{P}_r^{-1}$ reverses the directional serialization, λ_r weights the output of route r channel-wise, and $\odot$ denotes element-wise multiplication. These operations define the VisionHOPE operator $\mathcal{F}_{\mathrm{VH}}(Z) = Y$, which preserves the feature-map shape and scales linearly with token count N.

Block design. A VisionHOPE block combines the VisionHOPE operator with input and output projections, local depthwise convolution, and a Feed-Forward Network (FFN) in two pre-normalized residual branches (Figure 2(b)). For a block input $X \in \mathbb{R}^{B\times D\times H\times W}$, the structural form is

$$Z = \mathrm{DW}(\mathrm{Proj}_{\mathrm{in}}(\mathrm{Norm}_1(X))), \quad X' = X + \mathrm{Proj}_{\mathrm{out}}(\mathcal{F}_{\mathrm{VH}}(Z)), \quad X_{\mathrm{out}} = X' + \mathrm{FFN}\big(\mathrm{Norm}_2(X')\big), \tag{24}$$

where Norm_1 and Norm_2 denote normalization layers, $\mathrm{Proj}_{\mathrm{in}}$ and $\mathrm{Proj}_{\mathrm{out}}$ are the pointwise input and output projections that map channels from D to D_m and from D_m to D, respectively, and DW denotes depthwise convolution. Appendix B.5 provides further implementation details.

3 Experiments

We evaluate VisionHOPE on ImageNet-1K classification (Deng et al., 2009), COCO object detection and instance segmentation (Lin et al., 2014), and ADE20K semantic segmentation (Zhou et al., 2017). We further examine computational efficiency and key design choices. Model configurations, training protocols, and visual analysis are provided in Appendices C.1, C.2, and C.6, respectively.

Table 1: ImageNet-1K classification of hierarchical backbones at 224×224. The panels correspond to increasing model scales. Parameters, FLOPs, and Top-1 accuracy are reported in M, G, and %.

Method	Type	#P	FLOPs	Acc.
ConvNeXt-T	CNN	29	4.5	82.1
InternImage-T	CNN	30	5.0	83.5
MambaOut-T	CNN	27	4.5	82.7
FasterViT-1	ViT	53	5.3	83.2
TransNeXt-T	ViT	28	5.7	84.0
RMT-S	ViT	27	4.5	84.1
SOFT++-S	ViT	27	4.5	82.6
MILA-T	ViT	25	4.2	83.5
VMamba-T	SSM	30	4.9	82.6
LocalVMamba-T	SSM	26	5.7	82.7
MambaVision-T2	SSM	35	5.1	82.7
H-ViT^3-T	TTT	29	4.9	84.0
VisionHOPE-T	NL	27	4.9	84.1

Method	Type	#P	FLOPs	Acc.
ConvNeXt-S	CNN	50	8.7	83.1
InternImage-S	CNN	50	8.0	84.2
MambaOut-S	CNN	48	9.0	84.1
FasterViT-2	ViT	76	8.7	84.2
TransNeXt-S	ViT	50	10.3	84.7
RMT-B	ViT	54	9.7	85.0
SOFT++-M	ViT	48	8.7	83.7
MILA-S	ViT	43	7.3	84.4
VMamba-S	SSM	50	8.7	83.6
LocalVMamba-S	SSM	50	11.4	83.7
MambaVision-S	SSM	50	7.5	83.3
H-ViT^3-S	TTT	54	8.8	84.9
VisionHOPE-S	NL	53	9.8	85.2

Method	Type	#P	FLOPs	Acc.
ConvNeXt-B	CNN	89	15.4	83.8
InternImage-B	CNN	97	16.0	84.9
MambaOut-B	CNN	85	15.8	84.2
FasterViT-3	ViT	160	18.2	84.9
TransNeXt-B	ViT	90	18.4	84.8
RMT-L	ViT	95	18.2	85.5
SOFT++-L	ViT	85	15.4	84.1
MILA-B	ViT	96	16.2	85.3
VMamba-B	SSM	89	15.4	83.9
MambaVision-B	SSM	98	15.0	84.2
VSSD-B	SSM	89	16.1	85.4
H-ViT^3-B	TTT	94	16.7	85.5
VisionHOPE-B	NL	91	17.3	85.6

Table 2: ImageNet-1K classification of plain backbones at 224×224 resolution. The panels correspond to increasing scales. Parameters, FLOPs, and Top-1 accuracy are reported in M, G, and %.

Method	Type	#P	FLOPs	Acc.
DeiT-T	ViT	6	1.3	72.2
XCiT-T12/16	ViT	7	1.2	77.1
Agent-DeiT-T	ViT	6	1.2	74.9
Vim-T	SSM	7	1.6	76.1
Mamba®-T	SSM	9	1.4	77.4
ViT^3-T	TTT	6	1.2	76.5
Vision-TTT-T	TTT	7	1.4	77.7
P-VisionHOPE-T	NL	6	1.2	78.4

Method	Type	#P	FLOPs	Acc.
ConvNeXt-S (iso.)	CNN	22	4.3	79.7
DeiT-S	ViT	22	4.6	79.8
Agent-DeiT-S	ViT	23	4.4	80.5
Vim-S	SSM	26	5.3	80.3
Mamba®-S	SSM	28	5.1	81.4
ViT^3-S	TTT	24	4.8	81.6
Vision-TTT-S	TTT	26	5.3	81.8
P-VisionHOPE-S	NL	22	4.7	82.3

Method	Type	#P	FLOPs	Acc.
ConvNeXt-B (iso.)	CNN	87	16.9	82.0
DeiT-B	ViT	87	17.6	81.8
Agent-DeiT-B	ViT	87	17.6	82.0
Vim-B	SSM	98	19.0	81.9
Mamba®-B	SSM	99	19.8	83.0
ViT^3-B	TTT	90	18.0	82.6
Vision-TTT-B	TTT	102	20.3	82.7
P-VisionHOPE-B	NL	88	19.0	83.4

3.1 Image Classification

We train VisionHOPE from scratch on ImageNet-1K, which contains 1.28M training images and 50K validation images from 1,000 classes. We report single-crop Top-1 accuracy at 224×224 resolution. Tables 1 and 2 compare hierarchical and plain backbones, respectively. The compared backbones include CNN-based ConvNeXt (Liu et al., 2022), InternImage (Wang et al., 2023), and MambaOut (Yu & Wang, 2025); ViT-based DeiT (Touvron et al., 2021), XCiT (Ali et al., 2021), FasterViT (Hatamizadeh et al., 2024), TransNeXt (Shi, 2024), RMT (Fan et al., 2024), SOFT++ (Lu et al., 2024), Agent-DeiT (Han et al., 2024b), and MILA (Han et al., 2024a); SSM-based Vim (Zhu et al., 2024), VMamba (Liu et al., 2024), LocalVMamba (Huang et al., 2024), MambaVision (Hatamizadeh & Kautz, 2025), VSSD (Shi et al., 2025), and Mamba® (Wang et al., 2025); and TTT-based ViT^3 (including H-ViT^3) (Han et al., 2026) and Vision-TTT (Kong et al., 2026). Across both backbone layouts, VisionHOPE achieves the best or tied-best Top-1 accuracy in every scale panel, demonstrating the strong potential of self-modifying learning systems.

3.2 Object Detection and Instance Segmentation

We transfer ImageNet-pretrained hierarchical backbones to COCO using Mask R-CNN (He et al., 2017) and evaluate on val2017. The comparisons include the CNN-based backbones ConvNeXt (Liu et al., 2022), InternImage (Wang et al., 2023), and MambaOut (Yu & Wang, 2025); the ViT-based backbones Swin (Liu et al., 2021), CSWin (Dong et al., 2022), SOFT++ (Lu et al., 2024), and MILA (Han et al., 2024a); the SSM-based backbones VMamba (Liu et al., 2024), LocalVMamba (Huang et al., 2024), and VSSD (Shi et al., 2025); and the TTT-based H-ViT^3 (Han et al., 2026). Table 3 reports bounding-box and mask AP under the $1\times$ schedule, with FLOPs measured at 1280×800 resolution. Detailed results under the $3\times$ schedule are provided in Appendix C.4. VisionHOPE achieves the best or tied-best box and mask AP across all three evaluated model scales, showing that its gains transfer consistently to object detection and instance segmentation.

3.3 Semantic Segmentation

We transfer ImageNet-pretrained hierarchical backbones to ADE20K using UPerNet (Xiao et al., 2018) and evaluate on the validation set. The comparisons include the CNN-based backbones Con-

Table 3: COCO object detection and instance segmentation with Mask R-CNN under the $1\times$ schedule. The panels correspond to increasing backbone scales. FLOPs and AP are reported in G and %.

Method	FLOPs	AP^b	AP^m
ConvNeXt-T	262	44.2	40.1
InternImage-T	270	47.2	42.5
MambaOut-T	262	45.1	41.0
CSWin-T	279	46.7	42.2
SOFT++-S	261	43.8	40.1
MILA-T	255	46.8	42.1
VMamba-T	271	47.3	42.7
VSSD-T	265	46.9	42.6
H-ViT^3-T	271	47.3	42.8
VisionHOPE-T	266	47.9	43.1

Method	FLOPs	AP^b	AP^m
ConvNeXt-S	348	45.4	41.8
InternImage-S	340	47.8	43.3
MambaOut-S	354	47.4	42.7
CSWin-S	342	47.9	43.2
SOFT++-M	349	46.6	42.0
MILA-S	319	49.2	44.2
VMamba-S	349	48.7	43.7
VSSD-S	325	48.4	43.5
H-ViT^3-S	349	49.1	44.1
VisionHOPE-S	365	49.5	44.2

Method	FLOPs	AP^b	AP^m
ConvNeXt-B	486	47.0	42.7
InternImage-B	501	48.8	44.0
MambaOut-B	495	47.4	43.0
Swin-B	496	46.9	42.3
CSWin-B	526	48.7	43.9
SOFT++-L	481	47.0	42.2
MILA-B	502	50.5	45.0
VMamba-B	485	49.2	44.1
H-ViT^3-B	510	50.0	44.6
VisionHOPE-B	516	50.5	45.0

Table 4: ADE20K semantic segmentation with UPerNet. The panels correspond to increasing backbone scales. Complete-model parameters, FLOPs, and mIoU are reported in M, G, and %.

Backbone	#P	FLOPs	mIoU
ConvNeXt-T	60	939	46.0
MambaOut-T	54	938	47.4
VVT-S	56	960	46.8
SOFT++-S	60	948	46.5
MambaVision-T	55	945	46.0
H-ViT^3-T	58	946	48.0
VisionHOPE-T	55	942	49.4

Backbone	#P	FLOPs	mIoU
ConvNeXt-S	82	1027	48.7
MambaOut-S	76	1032	49.5
VVT-M	78	1040	48.1
SOFT++-M	81	1040	48.9
MambaVision-S	84	1135	48.2
H-ViT^3-S	84	1026	50.2
VisionHOPE-S	82	1043	50.3

Backbone	#P	FLOPs	mIoU
ConvNeXt-B	122	1170	49.1
MambaOut-B	112	1178	49.6
VVT-L	92	1068	48.8
SOFT++-L	121	1204	49.2
MambaVision-B	126	1342	49.1
H-ViT^3-B	124	1195	51.7
VisionHOPE-B	121	1200	51.8

vNeXt (Liu et al., 2022) and MambaOut (Yu & Wang, 2025); the ViT-based backbones VVT (Sun et al., 2023) and SOFT++ (Lu et al., 2024); the SSM-based MambaVision (Hatamizadeh & Kautz, 2025); and the TTT-based H-ViT^3 (Han et al., 2026). Table 4 reports mIoU, complete-model parameters, and FLOPs (512×2048). VisionHOPE achieves the best mIoU, showing its effectiveness.

3.4 Efficiency Analysis

Computational efficiency. For a square feature map containing N tokens of width D, we compare the per-layer interaction costs of DeiT (Touvron et al., 2021), Vim (Zhu et al., 2024), and VisionHOPE, including the associated projections, under the configurations used in our experiments:

$$\begin{aligned} \mathcal{C}(\mathrm{DeiT}) &= 4ND^2 + 2N^2D, \\ \mathcal{C}(\mathrm{Vim}) &= 6ND^2 + 576ND, \\ \mathcal{C}(\mathrm{VisionHOPE}) &= \frac{5}{4}ND^2 + 232ND + 1600D\sqrt{N}. \end{aligned} \tag{25}$$

Self-attention introduces a quadratic interaction term in N, whereas Vim and VisionHOPE remain linear in visual token count and scale more favorably as the input image resolution increases.

Memory efficiency. For batch size B, the corresponding activation-memory complexities are

$$\mathcal{M}(\mathrm{DeiT}) = O(BND + BN^2), \qquad \mathcal{M}(\mathrm{Vim}) = O(BND), \qquad \mathcal{M}(\mathrm{VisionHOPE}) = O(BND). \tag{26}$$

When materialized explicitly, DeiT's attention matrix introduces a quadratic memory term in N, whereas Vim and VisionHOPE retain linear memory scaling as the visual token count increases.

Figure 3 empirically confirms these theoretical trends. As input resolution increases, P-VisionHOPE-B maintains linear scaling with visual token count, whereas DeiT-B's computation and memory grow substantially faster. With similar FLOPs to Vim-B, P-VisionHOPE-B achieves higher throughput and lower memory use across all tested resolutions. These results demonstrate the scalability and practical efficiency of VisionHOPE. Appendix C.3 provides the complete derivations of Equations (25) and (26), together with additional efficiency results for the Tiny and Small models.

3.5 Ablation Study

We conduct the following ablations on P-VisionHOPE-S while keeping the backbone and training protocol fixed. The first group examines the contributions of the coupled SRNL memories and

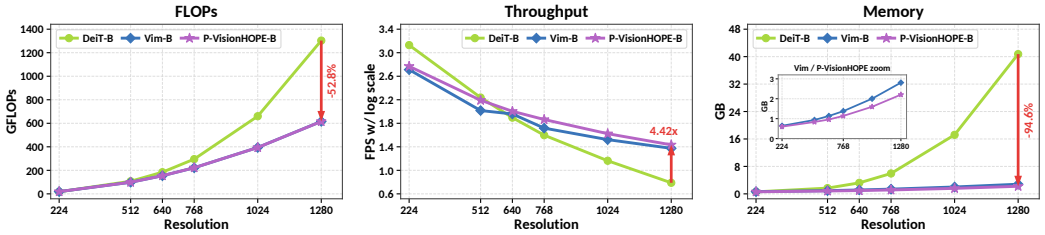

Figure 3: Efficiency comparison among DeiT-B, Vim-B, and P-VisionHOPE-B. We plot FLOPs, throughput, and memory footprint for input resolutions ranging from 224×224 to 1280×1280. Hardware measurements use one NVIDIA A100 GPU with BF16 and a fixed batch size of 8.

Table 5: Ablations on P-VisionHOPE-S. The left, middle, and right panels examine the coupled SRNL memories and DGD, stability-matched step-size control, and directional scanning and fusion. Top-1 accuracy is reported in %. A NaN entry indicates numerical divergence during training.

Variant	Acc.
w/o adaptive K/V & dynamics	81.7
w/o adaptive K/V	81.9
w/o adaptive dynamics	82.0
w/o DGD	81.9
Full model	82.3

Variant	Acc.
w/o soft injection cap	NaN
w/o spectral clamp	82.3
w/ injection clamp	81.8
w/ soft spectral cap	81.9
Full model	82.3

Variant	Acc.
1-way scan	81.6
2-way scan	81.9
4-way fixed sum	82.1
Shared initial states	82.0
Full model	82.3

their DGD update. We use *adaptive K/V* to denote within-image evolution of the key and value memories, and *adaptive dynamics* for that of the learning-rate and retention memories. Removing either leaves the corresponding mappings learned through outer training but fixed within the image, while the content memory continues to evolve. The DGD ablation removes the additional key-dependent correction $-\widetilde{\eta}_t M^{\square}_{t-1} k_t k_t^\top$, while retaining the output-space learning signal, retention factor, and stability-matched step-size control. The second group isolates the complementary roles of the soft injection cap and spectral clamp in stabilization. Removing either control leaves the other unchanged. The injection clamp replaces the soft injection cap with a hard clamp at the same injection limit, whereas the soft spectral cap replaces the spectral clamp with a smooth map to the same spectral limit. The final group examines the two-dimensional realization: the one-way variant retains only the forward row-major route, the two-way variant adds its reverse, and the four-way fixed-sum variant uses four routes with unit channel-wise scales. The variant with shared initial states ties the learned initial memories across directions while maintaining separate memory trajectories.

Table 5 shows consistent benefits from the coupled SRNL memories and their DGD update. Removing the soft injection cap causes the forward pass to diverge numerically early in training, demonstrating its importance for stable optimization. The injection clamp and soft spectral cap remain stable throughout training but are less effective than the complete control scheme. Removing the spectral clamp leaves accuracy unchanged in this setting, which may be related to its extremely low activation rate, as detailed in Appendix C.5. Increasing directional coverage improves performance, while channel-wise scales and direction-specific initial states provide further gains. Additional analyses of step-size control behavior and configuration choices are provided in Appendix C.5.

4 Conclusion

Building on NL, we introduced VisionHOPE, a self-modifying visual backbone whose stored content and learning rule co-evolve through five coupled memories within each image. Our stability-matched step-size control combines a soft injection cap with a spectral clamp to guarantee non-expansive memory dynamics. Four directional scans with spatially aligned chunks enable two-dimensional processing with linear scaling in token count. Experiments on ImageNet-1K, COCO, and ADE20K demonstrate strong performance in classification and dense prediction. These results establish self-modifying learning as a practical foundation for general-purpose visual backbones.

AI USE STATEMENT

We used LLM-based tools to polish the manuscript. We also used these tools to standardize code formatting. We have reviewed the AI-assisted text and code changes and take responsibility for the final manuscript and code.

REFERENCES

Alaaeldin Ali, Hugo Touvron, Mathilde Caron, Piotr Bojanowski, Matthijs Douze, Armand Joulin, Ivan Laptev, Natalia Neverova, Gabriel Synnaeve, Jakob Verbeek, and Herve Jegou. Xcit: Cross-covariance image transformers. In *Advances in Neural Information Processing Systems (NeurIPS)*, volume 34, pp. 20014–20027, 2021.

Marcin Andrychowicz, Misha Denil, Sergio Gómez, Matthew W Hoffman, David Pfau, Tom Schaul, Brendan Shillingford, and Nando de Freitas. Learning to learn by gradient descent by gradient descent. In *Advances in Neural Information Processing Systems (NeurIPS)*, volume 29, 2016.

Jimmy Ba, Geoffrey E Hinton, Volodymyr Mnih, Joel Z Leibo, and Catalin Ionescu. Using fast weights to attend to the recent past. In *Advances in Neural Information Processing Systems (NeurIPS)*, volume 29, 2016.

Ali Behrouz, Meisam Razaviyayn, Peilin Zhong, and Vahab Mirrokni. Nested learning: The illusion of deep learning architectures. In *Advances in Neural Information Processing Systems (NeurIPS)*, volume 38, Main Conference, pp. 46968–47002, 2025a.

Ali Behrouz, Peilin Zhong, and Vahab Mirrokni. Titans: Learning to memorize at test time. In *Advances in Neural Information Processing Systems (NeurIPS)*, volume 38, Main Conference, pp. 113506–113543, 2025b.

Han Cai, Junyan Li, Muyan Hu, Chuang Gan, and Song Han. Efficientvit: Multi-scale linear attention for high-resolution dense prediction. *arXiv preprint arXiv:2205.14756*, 2022.

Xiangxiang Chu, Zhi Tian, Bo Zhang, Xinlong Wang, and Chunhua Shen. Conditional positional encodings for vision transformers. In *Proceedings of the International Conference on Learning Representations (ICLR)*, 2023.

Tri Dao and Albert Gu. Transformers are SSMs: Generalized models and efficient algorithms through structured state space duality. In *Proceedings of the 41st International Conference on Machine Learning (ICML)*, volume 235, pp. 10041–10071, 21–27 Jul 2024.

Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. Imagenet: A large-scale hierarchical image database. In *Proceedings of the 2009 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 248–255, 2009.

Xiaohan Ding, Xiangyu Zhang, Jungong Han, and Guiguang Ding. Scaling up your kernels to 31x31: Revisiting large kernel design in cnns. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 11963–11975, June 2022.

Xiaohan Ding, Yiyuan Zhang, Yixiao Ge, Sijie Zhao, Lin Song, Xiangyu Yue, and Ying Shan. Unireplknet: A universal perception large-kernel convnet for audio video point cloud time-series and image recognition. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 5513–5524, June 2024.

Xiaoyi Dong, Jianmin Bao, Dongdong Chen, Weiming Zhang, Nenghai Yu, Lu Yuan, Dong Chen, and Baining Guo. Cswin transformer: A general vision transformer backbone with cross-shaped windows. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 12114–12124, 2022.

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. *arXiv preprint arXiv:2010.11929*, 2020.

JIAWEI DU, Daquan Zhou, Jiashi Feng, Vincent Tan, and Joey Tianyi Zhou. Sharpness-aware training for free. In *Advances in Neural Information Processing Systems (NeurIPS)*, volume 35, pp. 23439–23451, 2022.

Qihang Fan, Huaibo Huang, Mingrui Chen, Hongmin Liu, and Ran He. Rmt: Retentive networks meet vision transformers. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 5641–5651, June 2024.

Anthony Fuller, Yousef Yassin, Daniel Kyrollos, Evan Shelhamer, and James Green. Thicker and quicker: The jumbo token for fast plain vision transformers. In *Proceedings of the International Conference on Learning Representations (ICLR)*, volume 2026, pp. 50189–50211, 2026.

Albert Gu and Tri Dao. Mamba: Linear-time sequence modeling with selective state spaces. *arXiv preprint arXiv:2312.00752*, 2023.

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

David Ha, Andrew Dai, and Quoc V. Le. Hypernetworks. In *Proceedings of the International Conference on Learning Representations (ICLR)*, 2017.

Dongchen Han, Ziyi Wang, Zhuofan Xia, Yizeng Han, Yifan Pu, Chunjiang Ge, Jun Song, Shiji Song, Bo Zheng, and Gao Huang. Demystify mamba in vision: A linear attention perspective. In *Advances in Neural Information Processing Systems (NeurIPS)*, volume 37, pp. 127181–127203, 2024a.

Dongchen Han, Tianzhu Ye, Yizeng Han, Zhuofan Xia, Siyuan Pan, Pengfei Wan, Shiji Song, and Gao Huang. Agent attention: On the integration of softmax and linear attention. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pp. 124–140, 2024b.

Dongchen Han, Yining Li, Tianyu Li, Zixuan Cao, Ziming Wang, Jun Song, Yu Cheng, Bo Zheng, and Gao Huang. Vit^3: Unlocking test-time training in vision. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 51–61, June 2026.

Ali Hatamizadeh and Jan Kautz. Mambavision: A hybrid mamba-transformer vision backbone. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 25261–25270, June 2025.

Ali Hatamizadeh, Greg Heinrich, Hongxu Yin, Andrew Tao, Jose M. Alvarez, Jan Kautz, and Pavlo Molchanov. Fastervit: Fast vision transformers with hierarchical attention. In *Proceedings of the International Conference on Learning Representations (ICLR)*, volume 2024, pp. 29368–29391, 2024.

Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2016.

Kaiming He, Georgia Gkioxari, Piotr Dollar, and Ross Girshick. Mask r-cnn. In *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*, Oct 2017.

Tao Huang, Xiaohuan Pei, Shan You, Fei Wang, Chen Qian, and Chang Xu. Localmamba: Visual state space model with windowed selective scan. In *European Conference on Computer Vision (ECCV) Workshops*, pp. 12–22, 2024.

Kazuki Irie, Imanol Schlag, Róbert Csordás, and Jürgen Schmidhuber. Going beyond linear transformers with recurrent fast weight programmers. In *Advances in Neural Information Processing Systems (NeurIPS)*, volume 34, pp. 7703–7717, 2021.

Kazuki Irie, Imanol Schlag, Róbert Csordás, and Jürgen Schmidhuber. A modern self-referential weight matrix that learns to modify itself. In *Proceedings of the International Conference on Machine Learning (ICML)*, pp. 9660–9677. PMLR, 2022.

Quan Kong, Yanru Xiao, Yuhao Shen, and Cong Wang. Vision-ttt: Efficient and expressive visual representation learning with test-time training. *arXiv preprint arXiv:2603.00518*, 2026.

Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. Imagenet classification with deep convolutional neural networks. In *Advances in Neural Information Processing Systems (NeurIPS)*, volume 25, 2012.

Shufan Li, Harkanwar Singh, and Aditya Grover. Mamba-nd: Selective state space modeling for multi-dimensional data. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pp. 75–92, 2024a.

Siyuan Li, Zedong Wang, Zicheng Liu, Cheng Tan, Haitao Lin, Di Wu, Zhiyuan Chen, Jiangbin Zheng, and Stan Z Li. Moganet: Multi-order gated aggregation network. In *Proceedings of the International Conference on Learning Representations (ICLR)*, volume 2024, pp. 37393–37427, 2024b.

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 *Proceedings of the European Conference on Computer Vision (ECCV)*, pp. 740–755, 2014.

Yue Liu, Yunjie Tian, Yuzhong Zhao, Hongtian Yu, Lingxi Xie, Yaowei Wang, Qixiang Ye, Jianbin Jiao, and Yunfan Liu. Vmamba: Visual state space model. In *Advances in Neural Information Processing Systems (NeurIPS)*, volume 37, pp. 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 *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, pp. 9992–10002, 2021.

Zhuang Liu, Hanzi Mao, Chao-Yuan Wu, Christoph Feichtenhofer, Trevor Darrell, and Saining Xie. A convnet for the 2020s. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 11976–11986, June 2022.

Ilya Loshchilov and Frank Hutter. Decoupled weight decay regularization. In *Proceedings of the International Conference on Learning Representations (ICLR)*, 2019.

Jiachen Lu, Junge Zhang, Xiatian Zhu, Jianfeng Feng, Tao Xiang, and Li Zhang. Softmax-free linear transformers. *International Journal of Computer Vision*, 132(8):3355–3374, 2024.

Wenjie Luo, Yujia Li, Raquel Urtasun, and Richard Zemel. Understanding the effective receptive field in deep convolutional neural networks. In *Advances in Neural Information Processing Systems (NeurIPS)*, volume 29, 2016.

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 (AAAI)*, volume 39, pp. 6443–6451, 2025.

Anvitha Ramachandran, Dhruv Parikh, Haoyang Fan, Rajgopal Kannan, and Viktor Prasanna. Vision non-causal trapezoidal mamba: Eliminating directional scanning in vision ssms with second-order dynamics. *arXiv preprint arXiv:2607.03589*, 2026.

Imanol Schlag, Kazuki Irie, and Jürgen Schmidhuber. Linear transformers are secretly fast weight programmers. In *Proceedings of the 38th International Conference on Machine Learning (ICML)*, volume 139, pp. 9355–9366, 18–24 Jul 2021.

Dai Shi. Transnext: Robust foveal visual perception for vision transformers. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 17773–17783, June 2024.

Yuheng Shi, Mingjia Li, Minjing Dong, and Chang Xu. Vssd: Vision mamba with non-causal state space duality. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, pp. 10819–10829, October 2025.

Weixuan Sun, Zhen Qin, Hui Deng, Jianyuan Wang, Yi Zhang, Kaihao Zhang, Nick Barnes, Stan Birchfield, Lingpeng Kong, and Yiran Zhong. Vicinity vision transformer. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 45(10):12635–12649, 2023.

Yu Sun, Xinhao Li, Karan Dalal, Jiarui Xu, Arjun Vikram, Genghan Zhang, Yann Dubois, Xinlei Chen, Xiaolong Wang, Sanmi Koyejo, Tatsunori Hashimoto, and Carlos Guestrin. Learning to (Learn at test time): RNNs with expressive hidden states. In *Proceedings of the 42nd International Conference on Machine Learning (ICML)*, volume 267, pp. 57503–57522, 13–19 Jul 2025.

Hugo Touvron, Matthieu Cord, Matthijs Douze, Francisco Massa, Alexandre Sablayrolles, and Herve Jegou. Training data-efficient image transformers & distillation through attention. In *Proceedings of the 38th International Conference on Machine Learning (ICML)*, volume 139, pp. 10347–10357, 18–24 Jul 2021.

Zhengzhong Tu, Hossein Talebi, Han Zhang, Feng Yang, Peyman Milanfar, Alan Bovik, and Yinxiao Li. Maxvit: Multi-axis vision transformer. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pp. 459–479, 2022.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. In *Advances in Neural Information Processing Systems (NeurIPS)*, volume 30, 2017.

Ao Wang, Hui Chen, Zijia Lin, Jungong Han, and Guiguang Ding. Repvit: Revisiting mobile cnn from vit perspective. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 15909–15920, June 2024.

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 *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 14944–14953, June 2025.

Feng Wang, Sucheng Ren, Tiezheng Zhang, Predrag Neskovic, Anand Bhattad, Cihang Xie, and Alan Yuille. Vit-5: Vision transformers for the mid-2020s. *arXiv preprint arXiv:2602.08071*, 2026.

Wenhai Wang, Jifeng Dai, Zhe Chen, Zhenhang Huang, Zhiqi Li, Xizhou Zhu, Xiaowei Hu, Tong Lu, Lewei Lu, Hongsheng Li, Xiaogang Wang, and Yu Qiao. Internimage: Exploring large-scale vision foundation models with deformable convolutions. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 14408–14419, June 2023.

Sanghyun Woo, Shoubhik Debnath, Ronghang Hu, Xinlei Chen, Zhuang Liu, In So Kweon, and Saining Xie. Convnext v2: Co-designing and scaling convnets with masked autoencoders. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 16133–16142, June 2023.

Tete Xiao, Yingcheng Liu, Bolei Zhou, Yuning Jiang, and Jian Sun. Unified perceptual parsing for scene understanding. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pp. 432–448, 2018.

Guoan Xu, Wenfeng Huang, Wenjing Jia, Jiamao Li, Guangwei Gao, and Guo-Jun Qi. S2aformer: Strip self-attention for efficient vision transformer. *IEEE Transactions on Image Processing*, 34: 8243–8256, 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. *arXiv preprint arXiv:2403.17695*, 2024.

Weihao Yu and Xinchao Wang. Mambaout: Do we really need mamba for vision? In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 4484–4496, June 2025.

Weihao Yu, Mi Luo, Pan Zhou, Chenyang Si, Yichen Zhou, Xinchao Wang, Jiashi Feng, and Shuicheng Yan. Metaformer is actually what you need for vision. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 10819–10829, June 2022.

Jingwei Zhang, Xi Han, Hong Qin, Mahdi S Hosseini, and Dimitris Samaras. Lbmamba: Locally bi-directional mamba. *arXiv preprint arXiv:2506.15976*, 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 (CVPR)*, July 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 *Proceedings of the 41st International Conference on Machine Learning (ICML)*, volume 235, pp. 62429–62442, 21–27 Jul 2024.

A Related Work

A.1 Convolutional and Transformer Backbones

CNNs established the modern visual backbone through local connectivity, shared spatial kernels, and translation-equivariant feature extraction (Krizhevsky et al., 2012; He et al., 2016). Modern ConvNets revisit this design space through updated architectural choices in ConvNeXt (Liu et al., 2022) and ConvNeXt V2 (Woo et al., 2023), enlarged receptive fields in RepLKNet (Ding et al., 2022) and UniRepLKNet (Ding et al., 2024), deformable spatial aggregation in InternImage (Wang et al., 2023), multi-order gated aggregation in MogaNet (Li et al., 2024b), and efficient mobile design in RepViT (Wang et al., 2024). MambaOut (Yu & Wang, 2025) further builds gated convolutional backbones by removing the state-space component from Mamba blocks (Gu & Dao, 2023). These models demonstrate the continued competitiveness of local aggregation. However, the mappings governing that aggregation are learned during training and remain fixed during image processing.

ViT replaces local convolution with content-dependent global self-attention (Vaswani et al., 2017; Dosovitskiy et al., 2020), while Swin Transformer (Liu et al., 2021) combines hierarchical representations with shifted-window attention. Subsequent backbones broaden this design space by revisiting both backbone organization and the mechanisms used for visual token interaction. MetaFormer (Yu et al., 2022) abstracts the common backbone structure beyond a particular token mixer, and MaxViT (Tu et al., 2022) integrates window and grid attention for local and global interaction. EfficientViT (Cai et al., 2022) develops multi-scale linear attention for high-resolution dense prediction. RMT (Fan et al., 2024) introduces a retentive mechanism for spatially aware representation, while TransNeXt (Shi, 2024) develops foveal token aggregation. More recent work accelerates plain ViTs through a wider global token (Fuller et al., 2026), develops strip self-attention for efficient multi-scale perception (Xu et al., 2025), and systematically modernizes the canonical ViT block (Wang et al., 2026). Although these models make token interactions dependent on the current image, the learned mechanism generating those interactions does not evolve during inference.

A.2 Visual State-Space Models

Structured SSMs provide long-range sequence modeling through recurrent state updates (Gu et al., 2021). Mamba (Gu & Dao, 2023) introduces input-dependent state transitions and a hardware-aware parallel algorithm, achieving linear sequence-length scaling. Mamba-2 (Dao & Gu, 2024) connects structured state spaces and attention through state-space duality. Adapting these models to images requires resolving the mismatch between ordered sequences and two-dimensional spatial structure. Vim (Zhu et al., 2024) uses bidirectional state-space processing, and VMamba (Liu et al., 2024) introduces four-directional selective scanning over two-dimensional feature maps. Related visual SSMs explore non-hierarchical layouts (Yang et al., 2024), multidimensional traversal (Li et al., 2024a), windowed or atrous selective scans (Huang et al., 2024; Pei et al., 2025), and non-causal state-space duality (Shi et al., 2025). More recent work combines redesigned Mamba blocks with attention in a hierarchical backbone (Hatamizadeh & Kautz, 2025), introduces locally bidirectional recurrence without an additional global reverse scan (Zhang et al., 2025), and explores second-order non-causal dynamics that eliminate directional scanning (Ramachandran et al., 2026). These methods make the state transition input-dependent, but their transition-generation rule remains fixed: tokens determine transitions through an outer-parameterized mapping unchanged within the image.

A.3 Test-Time Training in Vision

TTT layers use an inner learner as their recurrent state and update it through self-supervised learning (Sun et al., 2025). The learner compresses input-derived key-value pairs into its parameters and produces representations from queries. In vision, ViT^3 (Han et al., 2026) studies full-image inner adaptation, focusing on the learner architecture and inner training procedure. Vision-TTT (Kong et al., 2026) adapts the learner recurrently along visual token sequences. Its dual-dataset strategy pairs forward and reverse token sequences, while convolutional preprocessing incorporates local spatial information. Together, they establish TTT-based visual backbones in both full-image and token-indexed forms. VisionHOPE shares within-image adaptation with these methods but differs in what evolves. TTT-based visual backbones adapt an inner learner while their representation maps and update dynamics remain largely outer-parameterized. VisionHOPE follows NL's self-referential construction by coupling content, key, value, learning-rate, and retention memories (Behrouz et al., 2025a). This coupling allows stored content and the learning rule to co-evolve with visual context.

A.4 Nested Learning and Self-Modifying Systems

Fast-weight models update short-term parameters from the current context to capture recent history (Ba et al., 2016), while HyperNetworks generate parameters of one network using another (Ha et al., 2017). Learned optimizers make update rules trainable (Andrychowicz et al., 2016), and fast-weight programmers connect linear attention and recurrent sequence modeling with online associative memory (Schlag et al., 2021; Irie et al., 2021). Self-referential weight matrices generate their update quantities and modify themselves through the delta rule (Irie et al., 2022). Titans (Behrouz et al., 2025b) treats neural memory as a learner updated at test time to retain information over long contexts. These works establish key ingredients of adaptive computation: context-dependent parameters, trainable update rules, and memory implemented as a learning process during inference.

Building on these ideas, NL describes a model as a collection of interconnected learning processes, each compressing its context flow into an internal state (Behrouz et al., 2025a). Its self-referential construction couples five memories that store content, generate key and value representations, and govern learning rate and retention. This coupling allows stored content and the learning rule to co-evolve as context accumulates. NL also derives a chunk formulation and combines its self-referential component with a Continuum Memory System (CMS) to form the HOPE architecture. CMS updates its memory modules at different frequencies to retain information over multiple timescales. In NL's formulation, the learning signal for these modules comes from a task objective. Language modeling provides this signal through next-token prediction, whereas discriminative vision has no canonical token-wise target along a visual scan. Incorporating CMS into a visual backbone would therefore require choosing an additional online learning objective. VisionHOPE focuses on the self-referential component and adapts it to generic visual backbones. In vision, the unconstrained self-referential update creates a stability barrier, which VisionHOPE addresses with a step-size control scheme.

B Additional Methodological Details

B.1 Associative Memory and Gradient Descent

Following NL, we detail the associative-memory formulation summarized in Section 2.1.1.

Associative memory. Let $\mathcal{K} = \{k_i\}_{i=1}^n$ and $\mathcal{V} = \{v_i\}_{i=1}^n$ denote paired keys and values, with $k_i \in \mathbb{R}^{d_k}$ and $v_i \in \mathbb{R}^{d_v}$. Here, n is the number of associations, while d_k and d_v are the respective key and value dimensions. An associative memory is a parameterized map $M : \mathbb{R}^{d_k} \to \mathbb{R}^{d_v}$ that compresses these associations into its parameters. Given an internal objective $\widetilde{\mathcal{L}}$ that measures how well M maps each key to its paired value, learning the memory is formulated as follows:

$$M^\star = \arg\min_M \widetilde{\mathcal{L}}\left(M(\mathcal{K}); \mathcal{V}\right), \tag{27}$$

where $M(\mathcal{K}) = \{M(k_i)\}_{i=1}^n$ denotes predictions for all keys, and $M^\star$ is the optimized memory. The parameters of M form its memory state, whereas optimizing this objective is the associated learning process. The keys and values are not restricted to input tokens and may represent data samples, gradients, intermediate representations, subsequences, or other elements of a context flow.

Gradient descent as associative memory. To instantiate the general memory M in Equation (27), set $d_k = d_v = d$ and consider a linear associative memory $M_W : \mathbb{R}^d \to \mathbb{R}^d$. The memory is parameterized by $W \in \mathbb{R}^{d\times d}$ and maps a key $k \in \mathbb{R}^d$ as $M_W(k) = Wk$. At online step t of the context flow, W_{t-1} denotes the memory state after $t-1$ elements and $k_t \in \mathbb{R}^d$ is the current key. In VisionHOPE, each step corresponds to a visual token. The response is $z_t = M_{W_{t-1}}(k_t) = W_{t-1}k_t \in \mathbb{R}^d$. Let $\mathcal{J}_t : \mathbb{R}^d \to \mathbb{R}$ be a differentiable local objective on a response $z \in \mathbb{R}^d$. Its gradient with respect to z at z_t defines the output-space learning signal $g_t = \nabla_z \mathcal{J}_t(z)|_{z=z_t} \in \mathbb{R}^d$. Holding k_t fixed, the chain rule gives the gradient with respect to the memory parameters

$$\nabla_W \mathcal{J}_t(Wk_t)|_{W=W_{t-1}} = g_t k_t^\top. \tag{28}$$

Here, ∇_W denotes differentiation with respect to the memory parameters W, and $g_t k_t^\top \in \mathbb{R}^{d\times d}$ is a rank-one outer product. Ordinary gradient descent then updates the memory state as

$$W_t^{\mathrm{GD}} = W_{t-1} - \eta_t g_t k_t^\top, \qquad \eta_t > 0, \tag{29}$$

where W_t^{GD} is the updated memory state and η_t is the learning rate at the current step. The resulting rank-one term $-\eta_t g_t k_t^\top$ admits an associative-memory interpretation: k_t specifies the write address, while $-g_t$ provides the write signal. In this view, W serves as the memory state, while gradient descent is the learning process that compresses input-dependent learning signals into it.

Dot-product proximal formulation. With k_t, g_t, and η_t held fixed, NL equivalently expresses Equation (29) as the solution to a proximal problem with a dot-product mapping objective:

$$W_t^{\mathrm{GD}} = \arg\min_{W\in\mathbb{R}^{d\times d}} \left[\langle Wk_t, g_t\rangle + \frac{1}{2\eta_t} \|W - W_{t-1}\|_F^2 \right], \tag{30}$$

where $W \in \mathbb{R}^{d\times d}$ is a candidate updated memory, and $\langle\cdot,\cdot\rangle$ is the Euclidean inner product. The dot-product term encourages the response Wk_t to move along the write signal $-g_t$, while the quadratic penalty discourages large changes from W_{t-1}. Differentiating the objective with respect to W gives

$$\nabla_W \langle Wk_t, g_t\rangle = g_t k_t^\top, \qquad \nabla_W \frac{1}{2\eta_t} \|W - W_{t-1}\|_F^2 = \eta_t^{-1}(W - W_{t-1}). \tag{31}$$

Setting their sum to zero at the minimizer W_t^{GD} yields

$$g_t k_t^\top + \eta_t^{-1}(W_t^{\mathrm{GD}} - W_{t-1}) = 0 \quad \Longrightarrow \quad W_t^{\mathrm{GD}} = W_{t-1} - \eta_t g_t k_t^\top, \tag{32}$$

which recovers Equation (29). Since $\eta_t > 0$, the objective is strictly convex and the minimizer is unique. The dot-product formulation yields an additive rank-one write with no additional correction based on the current response $W_{t-1}k_t$. NL argues that such an additional correction is useful for context flows with highly correlated elements. Replacing the dot-product term with an L_2 regression objective for the target $u_t = -g_t$ yields the DGD formulation derived in Appendix B.2.

B.2 Proximal Derivation of Delta Gradient Descent

We derive the recurrence in Equation (3) from the proximal L_2 objective in Equation (1). Here, $u_t = -g_t$ is the regression target and $\lambda_t > 0$ is the proximal parameter. With k_t, u_t, and λ_t held fixed, differentiating the two terms of the objective with respect to the candidate memory W gives

$$\nabla_W \frac{1}{2} \|Wk_t - u_t\|_2^2 = (Wk_t - u_t)\, k_t^\top, \qquad \nabla_W \frac{1}{2\lambda_t} \|W - W_{t-1}\|_F^2 = \lambda_t^{-1}\,(W - W_{t-1})\,. \tag{33}$$

Since $\lambda_t > 0$, the objective is strictly convex with respect to W. Setting the sum of these gradients to zero therefore gives the first-order condition for its unique minimizer W_t^{DGD}:

$$\left(W_t^{\mathrm{DGD}} k_t - u_t\right) k_t^\top + \lambda_t^{-1}\left(W_t^{\mathrm{DGD}} - W_{t-1}\right) = 0. \tag{34}$$

Multiplying by λ_t and collecting the terms containing W_t^{DGD} gives

$$W_t^{\mathrm{DGD}}\left(I_d + \lambda_t k_t k_t^\top\right) = W_{t-1} + \lambda_t u_t k_t^\top. \tag{35}$$

Here, $I_d \in \mathbb{R}^{d\times d}$ is the identity matrix. The matrix $I_d + \lambda_t k_t k_t^\top$ is positive definite and therefore invertible. Applying the Sherman-Morrison identity gives

$$\left(I_d + \lambda_t k_t k_t^\top\right)^{-1} = I_d - \frac{\lambda_t}{1+\lambda_t\|k_t\|_2^2} k_t k_t^\top = I_d - \eta_t k_t k_t^\top, \tag{36}$$

where $\eta_t = \lambda_t/(1+\lambda_t\|k_t\|_2^2)$ is the effective DGD step in Equation (2). NL simplifies this coefficient for normalized keys, while we keep the key norm explicit to cover the unnormalized case. Right-multiplying both sides of Equation (35) by this inverse and using $\lambda_t(1-\eta_t\|k_t\|_2^2) = \eta_t$ gives

$$W_t^{\mathrm{DGD}} = W_{t-1}\left(I_d - \eta_t k_t k_t^\top\right) + \eta_t u_t k_t^\top = W_{t-1} - \eta_t\left(W_{t-1}k_t - u_t\right)k_t^\top. \tag{37}$$

Substituting $u_t = -g_t$ into this expression recovers Equation (3).

B.3 Properties of the Stability-Matched Step-Size Control

This subsection establishes the token-wise guarantees of the stability-matched step-size control. We first characterize the soft injection cap, then prove complementary operator bounds for the retained memory transition and self-referential injection, and finally combine them to obtain token-wise non-expansion. We conclude with the finite-precision safeguards used in the practical implementation.

Since $\eta_t > 0$ and $\eta_t^{\mathrm{inj}} > 0$, their ratio is positive, giving $0 < 1 - \exp(-\eta_t/\eta_t^{\mathrm{inj}}) < 1$ and hence $0 < \bar{\eta}_t < \eta_t^{\mathrm{inj}}$. Differentiating with respect to the raw step while holding η_t^{inj} fixed gives

$$\frac{\partial\bar{\eta}_t}{\partial\eta_t} = \exp\left(-\frac{\eta_t}{\eta_t^{\mathrm{inj}}}\right) \in (0,1). \tag{38}$$

Thus, the soft injection cap is smooth and strictly increasing in η_t. Expanding around $\eta_t = 0$ yields

$$\bar{\eta}_t = \eta_t - \frac{\eta_t^2}{2\eta_t^{\mathrm{inj}}} + O\left(\frac{\eta_t^3}{(\eta_t^{\mathrm{inj}})^2}\right), \tag{39}$$

which proves first-order preservation of small raw steps. The spectral clamp in Equation (17) is piecewise smooth and differentiable almost everywhere with respect to its two scalar inputs.

Proof of Proposition 1. If $k_t = 0$, then $A_t = \alpha_t I_d$ and $B_t = 0$, so both required operator bounds hold directly. It therefore remains to consider $k_t \neq 0$. With $\kappa_t = \|k_t\|_2^2$, the spectral clamp ensures

$$0 \le \widetilde{\eta}_t\kappa_t \le \eta_t^{\mathrm{spec}}\kappa_t = 2\alpha_t. \tag{40}$$

The rank-one matrix $k_t k_t^\top$ has eigenvalue κ_t along k_t and zero on its orthogonal complement. Consequently, the retained transition $A_t = \alpha_t I_d - \widetilde{\eta}_t k_t k_t^\top$ has eigenvalue $\alpha_t - \widetilde{\eta}_t\kappa_t$ along k_t and α_t in every orthogonal direction. Its spectral bound is therefore equivalent to the following condition:

$$\|A_t\|_2 \le \alpha_t \quad\Longleftrightarrow\quad |\alpha_t - \widetilde{\eta}_t\kappa_t| \le \alpha_t \quad\Longleftrightarrow\quad 0 \le \widetilde{\eta}_t\kappa_t \le 2\alpha_t. \tag{41}$$

Together with Equation (40), this proves $\|A_t\|_2 \le \alpha_t$. For the self-referential injection, the rank-one matrix $\delta_t k_t^\top$ has spectral norm $\|\delta_t\|_2\|k_t\|_2$, giving

$$\|B_t\|_2 = \widetilde{\eta}_t\|\delta_t\|_2\|k_t\|_2 \le \bar{\eta}_t\|\delta_t\|_2\|k_t\|_2 \le \eta_t^{\mathrm{inj}}\|\delta_t\|_2\|k_t\|_2 \le (1-\alpha_t)\frac{\|\delta_t\|_2}{r_t} < 1-\alpha_t. \tag{42}$$

Here, the first two inequalities follow from $\widetilde{\eta}_t \le \bar{\eta}_t < \eta_t^{\mathrm{inj}}$, the third uses $\|k_t\|_2 \le 1$ and $\eta_t^{\mathrm{inj}} = (1-\alpha_t)/r_t$, and the final inequality is strict because $r_t > \|\delta_t\|_2$. The argument also covers $\delta_t = 0$, for which $B_t = 0$. This proves the strict bound $\|B_t\|_2 < 1-\alpha_t$ at every token. □

Proof of Corollary 1. By Proposition 1 and the triangle inequality,

$$\|T_t\|_2 \le \|A_t\|_2 + \|B_t\|_2 < \alpha_t + (1-\alpha_t) = 1. \tag{43}$$

Since $M_t^{\square} = M_{t-1}^{\square}T_t$, submultiplicativity gives

$$\|M_t^{\square}\|_F \le \|M_{t-1}^{\square}\|_F\|T_t\|_2 \le \|M_{t-1}^{\square}\|_F. \tag{44}$$

Applying this argument at every token proves non-expansion along the fully token-indexed trajectory, so each memory-state norm remains bounded by its initial value. The result applies to all five memories, with the learning-rate and retention memories treated as $1 \times d$ linear maps. □

In finite precision, VisionHOPE applies fixed multiplicative inward factors of $1 - 10^{-3}$ and $1 - 10^{-6}$ to the injection and spectral limits, respectively. The squared key norm in the denominator of η_t^{spec} is explicitly lower-bounded by 10^{-20}, and the resulting spectral limit is rounded to the next representable value toward zero. These safeguards make the implemented limits slightly more conservative and provide additional margins against roundoff errors near both step-size limits.

B.4 Chunk-Wise Stability of VisionHOPE

We apply the step-size control to the chunk-wise recurrence in Section 2.1.3. Fix a chunk spanning tokens $b+1, \ldots, b+C$, where b is a nonnegative multiple of C. For these tokens, $C \times \left\lfloor \frac{t-1}{C} \right\rfloor = b$, so the boundary state of memory $\square \in \{m, k, v, \eta, \alpha\}$ is $M_b^{\square}$. Let $\delta_t = k_t - v_t$ and obtain $\widetilde{\eta}_t$ by applying Equations (16) and (17) to the quantities generated from these boundary states. We use A_t and B_t defined in Equation (18), whose bounds in Proposition 1 remain valid for these quantities. Because $g_t^{\square} = M_b^{\square}\delta_t$, substituting these quantities and the executed step $\widetilde{\eta}_t$ into Equation (10) gives

$$M_t^{\square} = M_{t-1}^{\square} A_t + M_b^{\square} B_t, \qquad t = b+1, \ldots, b+C. \tag{45}$$

Proposition 2 (Shared right-multiplicative closure)**.** *For this fixed chunk, every memory $\square \in \{m, k, v, \eta, \alpha\}$ admits the following shared right-multiplicative representation:*

$$M_t^{\square} = M_b^{\square} G_t, \qquad G_b = I_d, \qquad G_t = G_{t-1} A_t + B_t, \tag{46}$$

where the recurrence holds for $t = b+1, \ldots, b+C$ and the same gain $G_t \in \mathbb{R}^{d\times d}$ is shared by all five memories. The gain is defined relative to the boundary of this fixed chunk.

Proof. The claim holds at $t = b$ because $M_b^{\square} = M_b^{\square} I_d$. For the inductive step, suppose $M_{t-1}^{\square} = M_b^{\square} G_{t-1}$. Substituting this relation into Equation (45) gives

$$M_t^{\square} = M_b^{\square} \left(G_{t-1} A_t + B_t\right) = M_b^{\square} G_t. \tag{47}$$

The matrices A_t and B_t do not depend on $\square$, so the resulting gain is shared by all memories. □

Theorem 1 (Non-expansion of the chunk-wise recurrence)**.** *Under the stability-matched step-size control, every within-chunk gain and memory state satisfies*

$$\|G_t\|_2 \le 1, \qquad \|M_t^{\square}\|_F \le \|M_b^{\square}\|_F, \qquad t = b, \ldots, b+C. \tag{48}$$

Proof. The case $t = b$ follows from $G_b = I_d$. For $b < t \le b + C$, Proposition 2 reduces all five memory trajectories to the shared-gain recurrence in Equation (46). Unrolling this recurrence gives

$$G_t = A_{b+1} \cdots A_t + \sum_{j=b+1}^{t} B_j A_{j+1} \cdots A_t, \tag{49}$$

where empty matrix and scalar products are understood as I_d and 1, respectively, and the multiplication order is inherited from the recurrence. Applying submultiplicativity and Proposition 1 yields

$$\|G_t\|_2 < \prod_{s=b+1}^{t} \alpha_s + \sum_{j=b+1}^{t} (1 - \alpha_j) \prod_{s=j+1}^{t} \alpha_s. \tag{50}$$

The inequality is strict because the final injection term, corresponding to $j = t$, retains the strict bound $\|B_t\|_2 < 1 - \alpha_t$. The accumulated injection terms telescope because

$$(1 - \alpha_j) \prod_{s=j+1}^{t} \alpha_s = \prod_{s=j+1}^{t} \alpha_s - \prod_{s=j}^{t} \alpha_s. \tag{51}$$

Therefore,

$$\prod_{s=b+1}^{t} \alpha_s + \sum_{j=b+1}^{t} (1 - \alpha_j) \prod_{s=j+1}^{t} \alpha_s = 1, \tag{52}$$

and Equation (50) gives $\|G_t\|_2 < 1$ for every $b < t \le b+C$. Finally, $M_t^{\square} = M_b^{\square} G_t$ and Frobenius-spectral submultiplicativity give $\|M_t^{\square}\|_F \le \|M_b^{\square}\|_F \|G_t\|_2 \le \|M_b^{\square}\|_F$. In particular, the state at the end of the chunk satisfies $\|M_{b+C}^{\square}\|_F \le \|M_b^{\square}\|_F$. This state becomes the boundary state for the next chunk, where the same argument applies with a new gain initialized to I_d. Repeating the argument bounds the norm of every memory state along the scan by that of its initial state. □

Table 6: Configurations of hierarchical VisionHOPE models. For an input of size $H \times W$, $\downarrow n$ denotes downsampling by a factor of n. $\mathrm{B}(D, D_m, n_h) \times L$ denotes L repeated VisionHOPE blocks with block width D, internal width D_m, and n_h heads per direction. The head dimension is $d = 16$.

	Size	VisionHOPE-T	VisionHOPE-S	VisionHOPE-B
Stage 1	$\left(\frac{H}{4} \times \frac{W}{4}\right)$	Stem $\downarrow 4$ $\mathrm{B}(64, 32, 2) \times 3$	Stem $\downarrow 4$ $\mathrm{B}(64, 32, 2) \times 4$	Stem $\downarrow 4$ $\mathrm{B}(96, 48, 3) \times 4$
Stage 2	$\left(\frac{H}{8} \times \frac{W}{8}\right)$	Down $\downarrow 2$ $\mathrm{B}(128, 64, 4) \times 4$	Down $\downarrow 2$ $\mathrm{B}(160, 80, 5) \times 8$	Down $\downarrow 2$ $\mathrm{B}(192, 96, 6) \times 8$
Stage 3	$\left(\frac{H}{16} \times \frac{W}{16}\right)$	Down $\downarrow 2$ $\mathrm{B}(256, 128, 8) \times 18$	Down $\downarrow 2$ $\mathrm{B}(320, 160, 10) \times 25$	Down $\downarrow 2$ $\mathrm{B}(448, 224, 14) \times 25$
Stage 4	$\left(\frac{H}{32} \times \frac{W}{32}\right)$	Down $\downarrow 2$ $\mathrm{B}(512, 256, 16) \times 4$	Down $\downarrow 2$ $\mathrm{B}(512, 256, 16) \times 8$	Down $\downarrow 2$ $\mathrm{B}(640, 320, 20) \times 8$
Classifier	–	Global average pooling, linear		

B.5 VisionHOPE Architecture Details

Section 2.3 presents the structural form of the VisionHOPE operator and block. In the reported models, the block input first undergoes Conditional Positional Encoding (CPE) (Chu et al., 2023) and pre-normalization before a pointwise input projection maps it to the internal width. The DW operation in Equation (24) is implemented as a gated depthwise-convolutional residual branch. It supplies local visual context and produces the feature map Z used by the VisionHOPE operator.

Along each route, we divide the D_m channels of Z_r into $n_h = D_m/d$ heads of dimension d. Each head maintains an independent five-memory SRNL state for each direction. In the multi-head implementation, we apply a $D_m \times D_m$ outer query projection once to Z and reuse the resulting queries across all four scan directions. For each direction, we reorder the queries into scan order, split them into heads, and apply L_2 normalization within each head before content readout.

After the recurrent scans, the head-wise content reads are concatenated and combined with a learned channel-wise skip connection from Z_r to form Y_r. The sequences are restored to the feature grid and fused using the direction scales in Equation (23), yielding Y. A depthwise local positional branch processes Z and adds its output to Y, forming a skip connection around the VisionHOPE operator. A pointwise output projection then maps the combined output back to the original width D.

At the block level, both residual branches use LayerNorm, with stochastic depth applied during training. The hierarchical backbone arranges VisionHOPE blocks across progressively downsampled stages, whereas the plain backbone stacks them at a single resolution. Row- and column-aligned chunk lengths follow each stage's feature resolution. Appendix C.1 provides model configurations.

C Additional Experimental Details

C.1 Model Configurations

Hierarchical models. VisionHOPE-T/S/B distribute VisionHOPE blocks across four progressively downsampled stages. Model scale is determined by the stage widths and depths. The FFN expansion ratio is set to 4 throughout. Table 6 specifies the complete hierarchical configurations.

Plain models. P-VisionHOPE-T/S/B retain a single feature resolution throughout the backbone after the convolutional stem. We scale these models by increasing the block width and total depth. The FFN expansion ratio is set to 4 throughout. Table 7 specifies the complete plain architectures.

Shared VisionHOPE settings. At 224×224 resolution, row and column chunk lengths coincide: $(56, 28, 14, 7)$ for the four hierarchical stages and 14 for the plain models. At other resolutions, row chunks span the feature map's width and column chunks its height. For the scalar outputs s_t^η and s_t^α of the learning-rate and retention memories, the maps in Equations (6) and (12) are given by

$$\eta_t = \phi_\eta(s_t^\eta) = \gamma_\eta \, \mathrm{softplus}(s_t^\eta), \qquad \alpha_t = \phi_\alpha(s_t^\alpha) = \sigma\left(s_t^\alpha + \log\frac{\alpha_{\mathrm{init}}}{1-\alpha_{\mathrm{init}}}\right). \tag{53}$$

Table 7: Configurations of plain VisionHOPE models. For an input of size $H \times W$, Stem $\downarrow$ 16 produces a feature map at $\left(\frac{H}{16} \times \frac{W}{16}\right)$. $\mathrm{B}(D, D_m, n_h) \times L$ denotes L repeated VisionHOPE blocks with block width D, internal width D_m, and n_h heads per direction. The head dimension is $d = 16$.

	Size	P-VisionHOPE-T	P-VisionHOPE-S	P-VisionHOPE-B
Backbone	$\left(\frac{H}{16} \times \frac{W}{16}\right)$	Stem $\downarrow 16$ $\mathrm{B}(192, 96, 6) \times 12$	Stem $\downarrow 16$ $\mathrm{B}(384, 192, 12) \times 13$	Stem $\downarrow 16$ $\mathrm{B}(768, 384, 24) \times 14$
Classifier	–	Global average pooling, linear		

Here, σ denotes the logistic sigmoid, $\gamma_\eta > 0$ is the raw-step scale, and $\alpha_{\text{init}} \in (0, 1)$ is the initial retention value. We set $\gamma_\eta = 0.025$ and $\alpha_{\text{init}} = 0.9$. The stability analysis requires no additional property of these scalar maps beyond $\eta_t > 0$ and $\alpha_t \in [0, 1)$. Their particular parameterization therefore does not alter the complementary operator bounds or the resulting non-expansion guarantees for the coupled memory dynamics. Before outer training, M_0^m, M_0^k, and M_0^v are initialized from a truncated normal distribution with standard deviation 0.01, whereas m_0^η and m_0^α are initialized to zero. Zero-initializing the scalar memories does not make the generated step zero: the initial raw step and retention factor are $\gamma_\eta \log 2$ and α_{init}, respectively. Setting $\alpha_{\text{init}} = 0.9$ preserves most of the existing memory while leaving the initial stability margin $1 - \alpha_{\text{init}} = 0.1$ for self-referential injection. Stabilized key normalization uses $k_t = k_t^{\text{raw}} / \sqrt{\|k_t^{\text{raw}}\|_2^2 + 10^{-6}}$, where k_t^{raw} denotes the key memory output before normalization. The step-size control in Definition 1 uses $\epsilon = 10^{-6}$.

C.2 Training Protocols

Image classification. We train our ImageNet models with AdamW (Loshchilov & Hutter, 2019). The 300-epoch schedule begins with 20 epochs of linear learning-rate warmup, followed by cosine decay. The learning rate is scaled linearly from 10^{-3} at a global batch size of 1024. The initial warmup and minimum learning rates are 10^{-6} and 10^{-5}, respectively. The hierarchical models undergo 10 additional training epochs at the minimum learning rate. The stochastic-depth rates are $(0.2, 0.4, 0.4)$ for VisionHOPE-T/S/B and $(0, 0.1, 0.5)$ for P-VisionHOPE-T/S/B. The hierarchical models additionally use MESA training (DU et al., 2022). Beginning at epoch 75, we use an exponential moving average teacher with decay 0.9998 to provide soft class targets for the student model. The corresponding soft cross-entropy loss is added to the classification objective with weights 1.0, 1.5, and 1.5 for the Tiny, Small, and Base models. The plain models are trained without MESA.

Object detection and instance segmentation. We initialize each Mask R-CNN model from its corresponding ImageNet-pretrained hierarchical backbone and fine-tune it with AdamW and standard multi-scale augmentation. The base learning rate is 4×10^{-4} for a global batch size of 64 and is scaled linearly when the batch size changes. The $1\times$ and $3\times$ schedules run for 12 and 36 epochs, respectively, with linear warmup over the first 1000 iterations and tenfold learning-rate reductions at epochs $(8, 11)$ and $(27, 33)$. The stochastic-depth rates are $(0.2, 0.3, 0.3)$ for VisionHOPE-T/S/B under the $1\times$ schedule and $(0.2, 0.4)$ for the Tiny and Small models under the $3\times$ schedule.

Semantic segmentation. We initialize UPerNet from the corresponding ImageNet-pretrained hierarchical backbones and optimize each model for 160K iterations with AdamW and a base learning rate of 6×10^{-5}. Training uses a global batch size of 16, linear warmup over the first 1500 iterations, and polynomial learning-rate decay with power 1.0. We follow the standard ADE20K data augmentation and use stochastic-depth rates of $(0.2, 0.4, 0.4)$ for VisionHOPE-T/S/B, respectively.

C.3 Efficiency Analysis Details

Computational complexity. Equation (25) directly compares the theoretical per-layer interaction costs for N tokens of width D. These estimates include the main linear projections and attention or recurrent computations, isolating token interaction from FFNs, local convolutions, and auxiliary block operations. In DeiT, the $4ND^2$ term comes from the standard query, key, value, and output projections, while $2N^2D$ accounts for constructing the dense attention matrix and applying it to the values. For Vim, let E denote the expanded inner width, S the hidden state dimension, and R the number of recurrent directions. The input and output projections contribute $2NDE$ and NDE,

respectively, while its recurrent selective state-space computation contributes $9RNES$. Thus,

$$\mathcal{C}(\text{Vim}) = 3NDE + 9RNES. \tag{54}$$

The Vim configurations considered here use $E = 2D$, $S = 16$, and $R = 2$, yielding

$$\mathcal{C}(\text{Vim}) = 6ND^2 + 576ND. \tag{55}$$

For VisionHOPE, let D_m denote the internal width, d the memory head dimension, $n_h = D_m/d$ the number of heads per direction, and C_r the chunk length along route $r \in \mathcal{D}$. The input and output projections, the outer query projection, and the SRNL instances applied along the four routes give

$$\mathcal{C}(\text{VisionHOPE}) = 2NDD_m + ND_m^2 + 4Nn_h(6d^2 + 20d) + n_hN(3d^3 + 2d^2)\sum_{r\in\mathcal{D}}\frac{1}{C_r}. \tag{56}$$

The first term counts the input and output projections, and the second counts the shared outer query projection applied once to the internal feature map. The third covers the generation of token-dependent quantities, the shared-gain recurrence with the executed step, and content reads across the four routes, while the final term accounts for updating the five memory states at chunk boundaries. The evaluated models use $D_m = D/2$, $d = 16$, and $n_h = D/32$. For an $H \times W$ map with $N = HW$, the row-major routes use $C_r = W$ and the column-major routes use $C_r = H$, making the boundary term $800D(H + W)$. For square maps with $H = W = \sqrt{N}$, Equation (56) becomes

$$\mathcal{C}(\text{VisionHOPE}) = \frac{5}{4}ND^2 + 232ND + 1600D\sqrt{N}. \tag{57}$$

Memory complexity. Equation (26) compares the theoretical per-layer activation-memory requirements for batch size B. DeiT stores $O(BND)$ intermediate token representations and, when materialized explicitly, an $O(BN^2)$ attention matrix. Using the same E, S, and R defined above, the token and direction-wise state-space activations in Vim give the following memory bound

$$\mathcal{M}(\text{Vim}) = O(BND + BRNES). \tag{58}$$

With $E = 2D$, $S = 16$, and $R = 2$, this reduces to the linear bound

$$\mathcal{M}(\text{Vim}) = O(BND). \tag{59}$$

For VisionHOPE, an upper bound on intermediate activation memory including token representations, directional token-dependent quantities, and chunk boundary storage is given by

$$\mathcal{M}(\text{VisionHOPE}) = O\left(BND + B|\mathcal{D}|ND_m + Bn_hNd^2\sum_{r\in\mathcal{D}}\frac{1}{C_r}\right). \tag{60}$$

The first two terms account for token representations and directional buffers. The third term bounds the storage of $d \times d$ gains, with one gain per chunk for each head and scan direction. Each gain is shared by the five memories within the corresponding SRNL instance. Computing the query projection once does not change the asymptotic storage required for the directional quantities. Substituting the model settings $|\mathcal{D}| = 4$, $D_m = D/2$, $d = 16$, and $n_h = D/32$ gives $O(BND + BD(H + W))$ for an $H \times W$ feature map. For square feature maps, this bound further simplifies to

$$\mathcal{M}(\text{VisionHOPE}) = O(BND + BD\sqrt{N}) = O(BND). \tag{61}$$

Resolution-scaling measurements. We compare DeiT, Vim, and P-VisionHOPE models at input resolutions from 224×224 to 1280×1280. All measurements use a single NVIDIA A100 GPU, BF16, and a fixed batch size of 8. Figure 4 reports the Tiny and Small results, complementing the Base comparison in Figure 3. Across both model scales, P-VisionHOPE maintains linear computation and memory growth with token count. Its efficiency advantage over DeiT widens substantially with resolution, yielding markedly lower FLOPs and memory use together with higher throughput at large inputs. Compared with Vim, P-VisionHOPE maintains a comparable FLOP profile, achieves higher throughput across all tested resolutions, and uses less memory at high resolutions.

C.4 COCO Results under the 3× Schedule

We additionally evaluate VisionHOPE on COCO under the longer $3\times$ training schedule to determine whether its transfer performance persists with extended optimization. Table 8 reports complete bounding-box and mask AP metrics, including the corresponding AP_{50} and AP_{75} results, for the Tiny and Small backbones. VisionHOPE retains competitive performance at both scales.

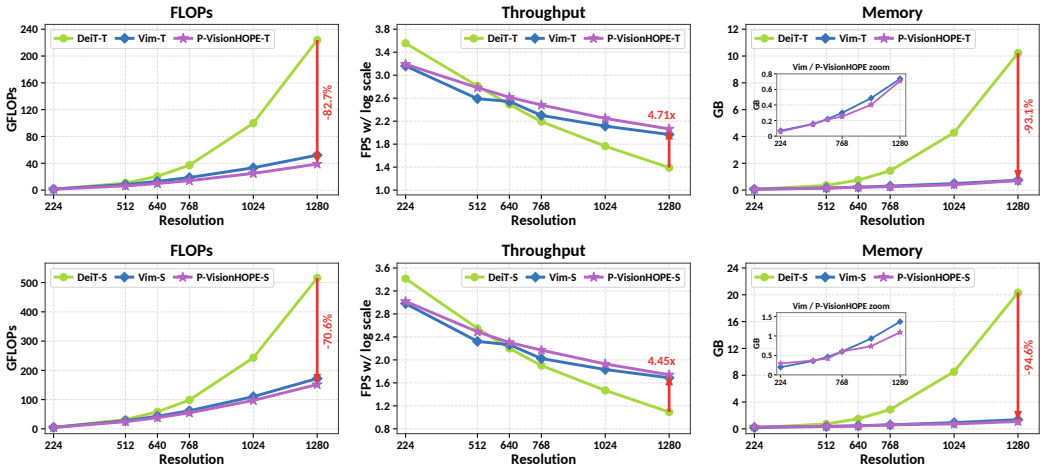

Figure 4: Efficiency comparison among DeiT, Vim, and P-VisionHOPE at the Tiny and Small scales. We plot FLOPs, throughput, and memory footprint for input resolutions from 224×224 to 1280×1280. Hardware measurements use one NVIDIA A100 GPU with BF16 and a fixed batch size of 8.

Table 8: COCO object detection and instance segmentation with Mask R-CNN under the $3\times$ schedule. FLOPs and AP are reported in G and %. VisionHOPE delivers competitive performance.

Method	FLOPs	AP^b	AP^b_{50}	AP^b_{75}	AP^m	AP^m_{50}	AP^m_{75}
ConvNeXt-T	262	46.2	67.9	50.8	41.7	65.0	44.9
InternImage-T	270	49.1	70.4	54.1	43.7	67.3	47.3
CSWin-T	279	49.0	70.7	53.7	43.6	67.9	46.6
MILA-T	255	48.8	71.0	53.6	43.8	68.0	46.8
VMamba-T	271	48.8	70.4	53.5	43.7	67.4	47.0
LocalVMamba-T	291	48.7	70.1	53.0	43.4	67.0	46.4
H-ViT^3-T	271	48.9	71.0	53.4	44.0	68.0	47.5
VisionHOPE-T	266	49.4	70.6	54.5	44.1	67.8	47.8

Method	FLOPs	AP^b	AP^b_{50}	AP^b_{75}	AP^m	AP^m_{50}	AP^m_{75}
ConvNeXt-S	348	47.9	70.0	52.7	42.9	66.9	46.2
InternImage-S	340	49.7	71.1	54.5	44.5	68.5	47.8
CSWin-S	342	50.0	71.3	54.7	44.5	68.4	47.7
MILA-S	319	50.5	71.8	55.2	44.9	69.1	48.2
VMamba-S	349	49.9	70.9	54.7	44.2	68.2	47.7
LocalVMamba-S	414	49.9	70.5	54.4	44.1	67.8	47.4
H-ViT^3-S	349	50.5	72.0	55.5	45.0	69.1	48.8
VisionHOPE-S	365	50.5	71.7	55.5	45.0	69.0	48.9

C.5 Additional Ablation Studies

Step-size control behavior. To interpret the results of the step-size control ablations in Table 5, we examine the step mappings and clamp activation rates. The soft injection cap is smooth and strictly increasing in the raw step η_t, so $\bar{\eta}_t$ preserves the graded modulation produced by the learning-rate memory while approaching the token-dependent limit η_t^{inj}. The injection clamp $\min(\eta_t, \eta_t^{\mathrm{inj}})$ is active when $\eta_t/\eta_t^{\mathrm{inj}} > 1$, a condition met by 49.68% of the token-wise updates during inference with the P-VisionHOPE-S injection-clamp variant on the ImageNet-1K validation set. When active, it replaces η_t with η_t^{inj} and removes direct dependence on raw-step magnitude, plausibly contributing to the lower accuracy. The spectral clamp is active when $\bar{\eta}_t/\eta_t^{\mathrm{spec}} = \bar{\eta}_t\|k_t\|_2^2/(2\alpha_t) > 1$. In the full P-VisionHOPE-S model, this condition holds for only 0.0044% of token-wise updates during inference on the same validation set and 0.0011% across the first ten warmup epochs of the standard training schedule. Since the spectral clamp leaves candidate steps below this limit unchanged, it modifies only these rare updates while enforcing the retained-transition bound, consistent with the matching accuracies of the full model and the variant without it. The soft spectral cap instead applies a second smooth map, $\eta_t^{\mathrm{spec}}\big(1 - \exp(-\bar{\eta}_t/\eta_t^{\mathrm{spec}})\big)$, to $\bar{\eta}_t$. Although this map respects the retained-transition bound and preserves small steps to first order, it modifies every positive candidate step, including those already within the limit. This additional remapping plausibly explains its lower accuracy. These comparisons help explain our choice of a soft injection cap and a spectral clamp.

Configuration sensitivity. We further examine four configuration choices using P-VisionHOPE-S. The ratio D_m/D controls the operator's internal width, chunk length determines the boundary-state update frequency, γ_η sets the scale of the raw step before step-size control, and d specifies the memory head dimension. Table 9 varies each setting and reports its effect on model size, computation, and ImageNet-1K accuracy. Together, these ablations support the defaults used throughout our

Table 9: Configuration ablations on P-VisionHOPE-S. The four panels vary D_m/D, chunk length, γ_η, and d, respectively. Parameters, FLOPs, and Top-1 accuracy are reported in M, G, and %.

D_m/D	#P	FLOPs	Acc.
1/4	20	4.2	81.5
1/2	22	4.7	82.3
1	27	6.1	82.3
2	44	9.8	82.5

Chunk	#P	FLOPs	Acc.
1	22	6.2	82.4
7	22	4.8	82.2
14	22	4.7	82.3
28	22	4.7	82.0

γ_η	#P	FLOPs	Acc.
0.0125	22	4.7	82.2
0.025	22	4.7	82.3
0.05	22	4.7	82.1
0.1	22	4.7	82.1

d	#P	FLOPs	Acc.
4	22	4.5	81.8
8	22	4.6	82.0
16	22	4.7	82.3
32	22	5.3	82.2

Table 10: ImageNet-1K classification of hierarchical models with and without MESA. The panels correspond to Tiny, Small, and Base models from left to right. The results for MILA and H-ViT^3 are taken from their respective papers (Han et al., 2024a; 2026). Top-1 accuracy is reported in %.

Method	Acc.
MILA-T (w/o MESA)	83.3
MILA-T (w/ MESA)	83.5
H-ViT^3-T (w/o MESA)	83.5
H-ViT^3-T (w/ MESA)	84.0
VisionHOPE-T (w/o MESA)	83.7
VisionHOPE-T (w/ MESA)	84.1

Method	Acc.
MILA-S (w/o MESA)	84.2
MILA-S (w/ MESA)	84.4
H-ViT^3-S (w/o MESA)	84.4
H-ViT^3-S (w/ MESA)	84.9
VisionHOPE-S (w/o MESA)	84.7
VisionHOPE-S (w/ MESA)	85.2

Method	Acc.
MILA-B (w/o MESA)	85.0
MILA-B (w/ MESA)	85.3
H-ViT^3-B (w/o MESA)	84.9
H-ViT^3-B (w/ MESA)	85.5
VisionHOPE-B (w/o MESA)	85.2
VisionHOPE-B (w/ MESA)	85.6

experiments, including the use of spatially aligned chunks that span a full row or column of visual tokens. Notably, the default chunk length $C = 14$ yields a $2.86\times$ inference speedup over the fully token-indexed setting $C = 1$, while maintaining comparable Top-1 accuracy. This gain highlights the practical importance of chunking for efficient recurrent computation in visual backbones.

Effect of MESA. Table 10 compares hierarchical VisionHOPE models with MILA and H-ViT^3 for image classification under training settings with and without MESA. MESA improves Top-1 accuracy for all compared model families. VisionHOPE obtains gains comparable to those of H-ViT^3 and achieves higher accuracy than both baselines at every scale under both settings. These results show that its advantage over the compared ViT and TTT models persists without MESA.

C.6 Visual Analysis

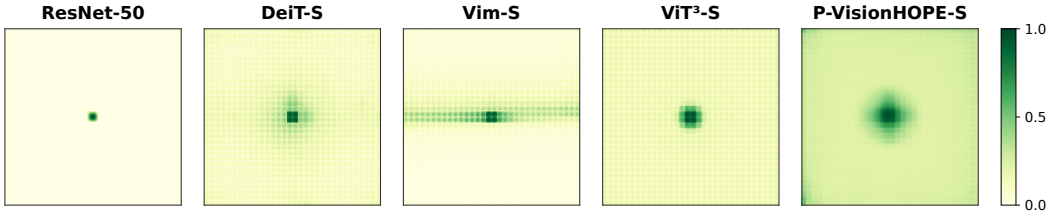

Figure 5: Qualitative comparison of global perception through Effective Receptive Fields (ERFs) at 512×512 resolution. The maps are computed from the geometrically centered 2×2 features at the native output of the second processing block and averaged across the complete ImageNet-1K validation set. Darker colors indicate stronger relative influence on the centered representation.

Global perception. The Effective Receptive Field (ERF) of an output representation measures the input region that influences its response, with each pixel magnitude indicating the strength of that influence (Luo et al., 2016). To compare global perception across visual backbones, we evaluate ImageNet-1K-pretrained models at 512×512 resolution and select the geometrically centered 2×2 features from the native output of the second processing block. We sum their positive activations, backpropagate the resulting scalar to the input, retain positive input gradients, sum across RGB channels, and average the maps across the full 50,000-image validation set. Figure 5 shows how strongly different input regions influence the centered representation, with darker colors indicating



Figure 6: Spatial variation of the final executed step in P-VisionHOPE-S across several ImageNet-1K validation images. Each two-row block pairs input images with the corresponding 14×14 maps of $\widetilde{\eta}_t$ from the final VisionHOPE block, restored to image coordinates and averaged across heads and four scan directions. Darker colors indicate relatively larger executed steps within each image.

larger relative responses. ResNet-50 (He et al., 2016) exhibits a localized receptive field, DeiT-S and ViT^3-S remain more concentrated around the center, and Vim-S presents a strongly directional pattern. In contrast, P-VisionHOPE-S distributes its influence broadly across the two-dimensional input while retaining a clear central response. This indicates stronger global perception in VisionHOPE.

Within-image learning dynamics. To examine how VisionHOPE modulates its within-image learning dynamics across spatial locations, we visualize the final executed step $\widetilde{\eta}_t$ from Equation (17)

in the final block of P-VisionHOPE-S. For each ImageNet-1K validation image, the token-wise values from the four directional scan orders are restored to the common 14×14 feature grid and averaged across heads and directions. Figure 6 shows that the executed step forms structured, content-dependent patterns across spatial locations. Darker regions represent relatively larger values of $\widetilde{\eta}_t$ and therefore a stronger tendency to modify the coupled memories at the corresponding locations. These regions frequently align with prominent objects and distinctive visual structures, suggesting that VisionHOPE learns semantically meaningful spatial variation in its within-image learning dynamics. The maps thus provide a direct and intuitive view of this adaptive process.

D STRENGTHS & WEAKNESSES

Strengths. VisionHOPE unifies within-image adaptation of stored content, update representations, and learning dynamics through five coupled memories. Its stability-matched step-size control provides non-expansion guarantees for both token-wise and chunk-wise memory recurrences. The operator supports hierarchical and plain backbones with computation and activation memory that scale linearly with visual token count. Results on classification and dense prediction benchmarks demonstrate its effectiveness across tasks, while ablations further support the main design choices.

Weaknesses and future work. Our current evaluation focuses on general-purpose visual recognition. We have not yet evaluated VisionHOPE as a visual encoder in Multimodal Large Language Models (MLLMs). Its effectiveness on specialized downstream tasks in medical imaging and remote sensing also remains to be empirically validated. We plan to extend VisionHOPE to these settings and assess its performance in both multimodal understanding and domain-specific visual tasks.