The Hidden Costs of Efficient Visual Representation Learning

by

Sahar DASTANI OGHANI

MANUSCRIPT-BASED THESIS PRESENTED TO ÉCOLE DE
TECHNOLOGIE SUPÉRIEURE
IN PARTIAL FULFILLMENT
FOR THE DEGREE OF DOCTOR OF PHILOSOPHY
Ph.D.

MONTREAL, SEPTEMBER 10, 2026

ÉCOLE DE TECHNOLOGIE SUPÉRIEURE
UNIVERSITÉ DU QUÉBEC

BOARD OF EXAMINERS

THIS THESIS HAS BEEN EVALUATED

BY THE FOLLOWING BOARD OF EXAMINERS

Mr. Christian Desrosiers, Thesis supervisor
Department of Software and IT Engineering, École de technologie supérieure

Mr. Hervé Lombaert, Thesis Co-Supervisor
Department of Computer Engineering and Software Engineering, Polytechnique Montréal

Mr. Mohammadhadi Shateri, Chair, Board of Examiners
Department of System Engineering, École de technologie supérieure

Mr. Ismail Ben Ayed, Member of the Jury
Department of System Engineering, École de technologie supérieure

Mr. Mahdi S. Hosseini, External Examiner
Department of Computer Science and Software Engineering, Concordia University

THIS THESIS WAS PRESENTED AND DEFENDED

IN THE PRESENCE OF A BOARD OF EXAMINERS AND THE PUBLIC

ON AUGUST 06, 2026

AT ÉCOLE DE TECHNOLOGIE SUPÉRIEURE

ACKNOWLEDGEMENTS

My PhD journey came with many challenges, but it also brought people into my life who had a profound impact on me. I feel especially fortunate that this journey led me to Christian. From our very first meeting, he welcomed me with kindness, care, and an open heart. His mentorship, generosity, encouragement, and belief in me helped me regain my confidence and move forward in my PhD. The supportive and collaborative environment he created allowed me to grow, work freely with others, and contribute to meaningful research. He also introduced me to industry opportunities and supported me wholeheartedly throughout my internships.

I am also deeply grateful to Hervé, who was a constant source of encouragement throughout my journey. He had a special ability to see the best in people and their work, to highlight their strengths, and to help them believe in themselves. His kindness, optimism, and support meant a great deal to me.

A very special thanks goes to my beloved family. To my mother, who spent many sleepless nights worrying about me from far away, yet made me feel that she was always beside me. To my father, whose wise words, strength, and support helped me through many difficult moments, especially when I felt close to giving up. To my twin sister, whose presence here made my life brighter and this journey much easier; her kindness, love, and support made this path feel possible. And to my brother, who has always been my steady anchor, bringing reassurance whenever I needed it most.

Finally, I would like to thank my dear close friends Saba, Zahra, Yasaman, Morti, Kiarash, and Masood, who became my family away from home. Their love, care, laughter, and constant presence carried me through many moments of this journey and made my life warmer and more beautiful. I would also like to thank my friends in Iran, my lab friends, teammates, collaborators, and my internship colleagues. Although I cannot name everyone here, I am deeply grateful for their friendship, support, conversations, and shared moments, which made my personal life and research journey more meaningful and enjoyable.

Les coûts cachés de l'apprentissage efficace des représentations visuelles

Sahar DASTANI OGHANI

RÉSUMÉ

L'apprentissage efficace de représentations visuelles constitue un défi central en vision par ordinateur moderne. À mesure que les modèles visuels sont de plus en plus utilisés pour des images à haute résolution et des tâches de prédiction dense, ils doivent capturer des dépendances à longue portée sans devenir prohibitifs sur le plan computationnel. Les Vision Transformers répondent à ce besoin en utilisant l'auto-attention pour modéliser les interactions globales entre les patchs d'image, mais la complexité quadratique de l'auto-attention limite leur passage à l'échelle. Cela a motivé le développement d'alternatives plus efficaces, telles que les modèles à espace d'états (State Space Models, SSMs) et l'attention linéaire, qui réduisent le coût des interactions entre tokens. Cependant, l'efficacité computationnelle seule ne suffit pas pour assurer un apprentissage fiable des représentations visuelles. Ces modèles doivent également être robustes aux changements des conditions d'entrée et suffisamment expressifs pour capturer des relations complexes entre les régions de l'image. Cette thèse étudie les coûts cachés introduits par les mécanismes efficaces d'interaction entre tokens dans l'apprentissage de représentations visuelles.

Premièrement, nous examinons la robustesse des SSMs face aux transformations géométriques. Les SSMs traitent les patchs d'image selon des ordres de balayage séquentiels fixes, ce qui permet une modélisation globale avec une complexité linéaire. Cependant, cette conception impose un ordre unidimensionnel fixe à des données d'image intrinsèquement bidimensionnelles. Par conséquent, les représentations apprises peuvent devenir sensibles au chemin de parcours choisi et instables face à des transformations géométriques telles que la rotation. Pour remédier à cette limitation, nous introduisons Spectral VMamba, un SSM visuel qui utilise la décomposition spectrale du laplacien de graphe des patchs d'image afin de définir le parcours des patchs selon la structure intrinsèque de l'image, plutôt que selon des directions spatiales fixes. Cela permet un traitement des patchs invariant à la rotation tout en préservant les avantages d'efficacité des SSMs.

Deuxièmement, nous examinons la robustesse des SSMs en présence de décalage de distribution, lorsque les images de test diffèrent des données d'entraînement en raison de corruptions, de changements de domaine ou de variations liées à l'acquisition. Dans les SSMs, les patchs sont traités séquentiellement, et l'état caché à chaque étape transporte le contexte accumulé à partir de tous les patchs précédemment traités. En présence de décalage de distribution, les embeddings de patchs corrompus sont injectés dans l'état caché et se propagent à travers toutes les mises à jour suivantes. De plus, comme les ordres de balayage fixes injectent toujours ces artefacts à la même position dans la séquence de parcours, leur influence est systématiquement amplifiée plutôt que diluée. Pour répondre à ce problème, nous proposons TRUST, une méthode d'adaptation au moment du test conçue pour les modèles visuels basés sur Mamba. TRUST génère des vues causales complémentaires de l'entrée à l'aide de permutations diverses du parcours, filtre les

prédictions selon l'incertitude afin de ne conserver que les permutations fiables, et met à jour les paramètres spécifiques à Mamba au moment de l'inférence, sans nécessiter de données sources ni de réentraînement. Cela permet au modèle d'adapter ses paramètres vers des représentations plus robustes sous des distributions de test décalées.

Troisièmement, nous abordons les limitations d'expressivité de l'attention linéaire. Bien que l'attention linéaire améliore le passage à l'échelle en évitant le coût quadratique de l'auto-attention, les formulations kernelisées standards reposent principalement sur des interactions directes et de premier ordre entre les tokens. Par conséquent, elles peuvent avoir des difficultés à capturer des relations d'ordre supérieur ou transitives entre les régions de l'image. Pour surmonter cette limitation, nous introduisons LiSA, un module d'attention stochastique linéaire qui reformule l'attention comme une marche aléatoire tronquée sur un graphe de patchs d'image. Grâce à la propagation d'information multi-sauts, LiSA capture des relations plus complexes entre tokens tout en préservant une complexité linéaire et un entraînement stable.

À travers la classification d'images, la segmentation sémantique et la détection d'objets, les méthodes proposées améliorent la robustesse et la qualité des représentations tout en maintenant l'efficacité computationnelle. Dans l'ensemble, cette thèse montre que la conception de backbones visuels efficaces ne doit pas être évaluée uniquement en fonction du coût computationnel. La scalabilité doit plutôt être considérée conjointement avec la robustesse et l'expressivité des interactions entre tokens, en particulier lorsque les modèles sont appelés à fonctionner de manière fiable face aux variations visuelles du monde réel.

Mots-clés: apprentissage de représentations visuelles, modèles de vision efficaces, modèles d'espace d'états, Mamba, attention linéaire, robustesse, décalage de distribution, adaptation au test, invariance à la rotation

The Hidden Costs of Efficient Visual Representation Learning

Sahar DASTANI OGHANI

ABSTRACT

Efficient visual representation learning is a key challenge in modern computer vision. As visual models are increasingly used for high-resolution images and dense prediction tasks, they must capture long-range dependencies without becoming computationally prohibitive. Vision Transformers address this by using self-attention to model global interactions between image patches, but the quadratic complexity of self-attention limits their scalability. This has motivated more efficient alternatives, such as State Space Models (SSMs) and linear attention, which reduce the cost of token interactions. However, computational efficiency alone is not enough for reliable visual representation learning. These models must also be robust to changes in input conditions and expressive enough to capture complex relationships across image regions. This thesis investigates the hidden costs introduced by efficient token interaction mechanisms in visual representation learning.

First, we examine the robustness of SSMs under geometric transformations. SSMs process image patches using fixed sequential scan orders, enabling global modeling with linear complexity. However, this design imposes a fixed one-dimensional order on inherently two-dimensional image data. Consequently, the learned representations can become sensitive to the chosen traversal path and unstable under geometric transformations such as rotation. To address this limitation, we introduce Spectral VMamba, a vision SSM that uses spectral decomposition of the graph Laplacian of image patches to define patch traversal according to the intrinsic structure of the image rather than fixed spatial directions. This enables rotation-invariant patch processing while preserving the efficiency advantages of SSMs.

Second, we examine the robustness of SSMs under distribution shift, where test images differ from training data due to corruptions, domain changes, or acquisition variability. In SSMs, patches are processed sequentially, and the hidden state at each step carries accumulated context from all previously processed patches. Under distribution shift, corrupted patch embeddings are injected into the hidden state and propagate through all subsequent updates, and because fixed scan orders always inject these artifacts at the same position in the traversal sequence, their influence is consistently amplified rather than diluted. To address this, we propose TRUST, a test-time adaptation method tailored for Mamba-based vision models. TRUST generates complementary causal views of the input via diverse traversal permutations, filters predictions by uncertainty to retain only reliable permutations, and updates Mamba-specific parameters at inference time, requiring neither source data nor retraining. This allows the model to adapt its parameters toward more robust representations under shifted test distributions.

Third, we address the expressiveness limitations of linear attention. Although linear attention improves scalability by avoiding the quadratic cost of self-attention, standard kernelized formulations mainly rely on direct, first-order interactions between tokens. As a result, they may struggle to capture higher-order or transitive relationships between image regions. To overcome

this limitation, we introduce LiSA, a Linear Stochastic Attention module that reformulates attention as a truncated random walk over a graph of image patches. Through multi-hop information propagation, LiSA captures richer token relationships while preserving linear complexity and stable training.

Across image classification, semantic segmentation, and object detection, the proposed methods improve robustness and representation quality while maintaining computational efficiency. Overall, this thesis shows that efficient visual backbone design should not be evaluated by computational cost alone. Instead, scalability must be considered together with the robustness and expressiveness of token interactions, especially when models are expected to operate reliably under real-world visual variations.

Keywords: Visual representation learning, efficient vision models, State Space Models, Mamba, linear attention, robustness, distribution shift, test-time adaptation, rotation invariance

TABLE OF CONTENTS

Page

LIST OF TABLES

LIST OF FIGURES

Page

LIST OF ABBREVIATIONS

PTQ	Post-training Quantization
QAT	Quantization-Aware Training
ToMe	Token Merging
DPLR	Diagonal Plus Low-Rank
FFT	Fast Fourier Transform
MFLOPs	Mega Floating-Point Operations
GFLOPs	Giga Floating-Point Operations
ms	Milliseconds
s	Seconds
CNN	Convolutional Neural Network
ViT	Vision Transformer
NLP	Natural Language Processing
SSM	State Space Model
SS2D	2D Selective Scan
DeiT	Data-efficient Image Transformer
EMA	Exponential Moving Average
KNN	K-Nearest Neighbors
RFN	Rotational Feature Normalizer
STS	Spectral Traversal Scan

STM	Spectral Traversal Merge
ERF	Effective Receptive Field
VLM	Vision-Language Model
OOD	Out-of-Distribution
TTA	Test-Time Adaptation
i.i.d.	Independent and Identically Distributed
BN	Batch Normalization
TRUST	Test-Time Refinement using Uncertainty-Guided SSM Traverses
RRM	Robust Risk Minimization
ZOH	Zero-Order Hold
LiSA	Linear Stochastic Attention
SA	Self-Attention
LA	Linear Attention
RW	Random Walk
ReLU	Rectified Linear Unit
ELU	Exponential Linear Unit
GPU	Graphics Processing Unit

LIST OF SYMBOLS

A	State transition matrix (SSM)
A^{local}	Local affinity matrix (LiSA)
B	Input projection matrix (SSM)
C	Output projection matrix (SSM)
D	Degree matrix (Spectral VMamba); row-normalization matrix (LiSA)
d	Token feature embedding dimension
d_h	Attention head dimension
$\mathcal{G}$	Graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$
$\mathcal{V}$	Set of graph nodes
$\mathcal{E}$	Set of graph edges
H	Image height
$\mathbf{h}(t)$	Hidden state vector at time step t (SSM)
$\mathbf{I}$	Identity matrix
K	Keys matrix (attention)
$\overline{K}$	Convolution kernel (SSM)
k	Number of nearest neighbors (KNN)
L	Sequence/convolution-kernel length (SSM)
$\mathbf{L}_{\text{sym}}$	Symmetric normalized Laplacian matrix
m	Number of eigenvectors (Spectral VMamba)

N	State dimension (SSM)
n	Number of tokens
p	Patch size
$P^{(T)}$	Truncated random-walk probability matrix with truncation length T
Q	Query matrix
$\tilde{Q}$	Row-normalized query matrix
S	Row-stochastic transition matrix
T	Number of image patches (TRUST); truncation length (LiSA)
V	Value matrix (attention)
W	Image width / Weighted adjacency matrix
W_Q, W_K, W_V	Learnable projection matrices for queries, keys, and values
$x(t)$	Input sequence at time t
$y(t)$	Output sequence at time t
α	Learnable global-local mixing parameter (LiSA)
β	Random-walk continuation probability (LiSA)
β_k	Learnable normalized diffusion-step weight, predicted per token and head (LiSA)
Δ	Discretization step size (SSM)
ε	Domain-specific corruption artifact
λ	Laplacian eigenvalue
π	Traversal permutation (TRUST)

σ	Scaling parameter for adjacency weights
θ	Model / SSM parameters
$\phi(\cdot)$	Positive feature map (linear attention)

INTRODUCTION

Deep learning has become a dominant approach in computer vision, enabling major progress in tasks such as image classification, object detection, and semantic segmentation. Early successes in visual recognition were largely driven by Convolutional Neural Networks (CNNs), which use local convolutional filters to extract hierarchical visual features. By progressively combining low-level patterns such as edges and textures into higher-level semantic concepts, CNNs have provided a strong foundation for many vision systems. Despite their success, CNNs mainly rely on local operations, which makes modeling global relationships less direct.

More recently, Vision Transformers (ViTs) have introduced a different paradigm for visual representation learning. Instead of relying primarily on local convolutional operations, ViTs divide an image into a sequence of patches and use self-attention to model relationships among them. This allows the model to capture long-range dependencies and global context more directly than traditional convolutional architectures.

However, this flexibility comes at a significant computational cost, since self-attention compares every token with every other token and its complexity grows quadratically with the number of image patches. This creates a major bottleneck for high-resolution images, dense prediction tasks, and resource-constrained deployment settings, where the cost of self-attention limits not only speed, but also memory usage and scalability. This has motivated the development of efficient alternatives that aim to retain the benefits of global token interactions while reducing computational complexity.

Among these alternatives, State Space Models (SSMs) and linear attention have recently gained traction. SSMs process visual tokens sequentially through a latent state, which acts as a compact memory that carries information from previously processed patches. Since images do not have a natural sequential order, structured scan operations are used to define the order in which image patches are processed. Linear attention, on the other hand, reformulates self-attention

by applying kernel-based transformations to queries and keys, allowing the computation to be reordered without explicitly constructing the full pairwise attention matrix. This makes both SSMs and linear attention substantially more efficient than standard self-attention.

However, efficiency alone is not sufficient for reliable visual representation learning. In practice, a visual backbone must not only process tokens at low computational cost, but also produce representations that are both robust and informative across diverse visual conditions. On one hand, it must generalize to inputs that may differ from the training distribution due to changes in lighting, viewpoint, geometric transformations, corruption, or acquisition conditions. On the other hand, it must capture rich relationships between image regions, including long-range dependencies and higher-order interactions that are essential for understanding complex scenes.

This motivates two further properties. *Robustness* refers to the stability of these interactions under changes in input conditions, such as geometric transformations, corruptions, or distribution shifts between training and deployment. *Expressiveness* refers to the ability of a token interaction mechanism to capture rich relationships between image regions, including long-range dependencies, fine-grained spatial structure, and higher-order interactions that are essential for understanding complex scenes. Together with efficiency, these properties define a fundamental efficiency–robustness–expressiveness trade-off in visual backbone design. The **central problem** addressed in this thesis is therefore how to preserve the computational advantages of efficient token-interaction mechanisms while overcoming the robustness and expressiveness limitations introduced by their structural constraints.

Evaluated from this perspective, current efficient alternatives to self-attention reveal characteristic limitations. SSMs process tokens through fixed sequential scans over image patches, imposing a causal ordering on inherently two-dimensional visual data. Unlike language, where causal left-to-right ordering is natural, images do not have a sequential direction. As a result, SSM representations can become highly dependent on the chosen traversal order and become less

stable when test inputs deviate from the training distribution. Geometric transformations, such as rotations, cause the same visual content to be processed in a different sequence, disrupting the order-dependent representations learned during training. Similarly, corruptions and domain shifts reveal the rigidity of a fixed scan order, limiting the model's ability to adapt to unseen input variations. This poses a robustness challenge in real-world settings, where such variations are common.

Linear attention, on the other hand, achieves efficiency by replacing the full softmax attention matrix with kernel-based approximations that allow the computation to be reordered. While this reduces complexity, it also weakens the selective, content-dependent interactions provided by softmax attention. Consequently, linear attention mixes each query with values exactly once through direct first-order affinities, and cannot capture transitive relationships between tokens that are semantically related but weakly similar in the feature space, leading to an expressiveness limitation that can constrain the quality of learned representations.

Existing work has proposed several strategies to mitigate these limitations. For vision SSMs, methods such as VMamba (Liu *et al.*, 2024b), LocalMamba (Huang *et al.*, 2024), and PlainMamba (Yang *et al.*, 2024) redesign the spatial scanning mechanism to better accommodate the two-dimensional structure of images, while hybrid architectures such as MambaVision (Hatamizadeh & Kautz, 2024) combine state-space modeling with self-attention. However, these approaches primarily focus on architectural design and predictive performance, while the dependence of SSM representations on traversal order under geometric transformations remains insufficiently addressed. Similarly, although test-time adaptation methods such as Tent (Wang, Shelhamer, Liu, Olshausen & Darrell, 2021a), EATA (Niu *et al.*, 2022), SAR (Niu *et al.*, 2023), and CoTTA (Wang, Fink, Van Gool & Dai, 2022a) improve robustness under distribution shift, they are largely architecture-agnostic and do not explicitly exploit the directional traversal structure of Mamba-based models.

For linear attention, prior work has mainly sought to close the performance gap with softmax attention through improved kernel approximations (Choromanski *et al.*, 2021; Qin *et al.*, 2022c), stronger local modeling (Han *et al.*, 2023a; Cai, Gan & Han, 2022), improved normalization and gating mechanisms (Qin *et al.*, 2022b; Han *et al.*, 2024b), or combinations with softmax-based interactions (Han *et al.*, 2024c; You *et al.*, 2023). Despite these advances, most linear-attention formulations still compute token representations through a single direct aggregation of values, limiting their ability to model transitive and higher-order relationships between visual regions. Taken together, these observations reveal a common gap: existing efficient visual architectures have primarily focused on reducing computational cost, while the consequences of their interaction structures for robustness and expressiveness remain comparatively underexplored.

Research statement

In this thesis, we investigate the hidden costs of efficient visual representation learning. While recent visual backbones improve computational scalability by replacing quadratic self-attention with more efficient token interaction mechanisms, these gains often come with overlooked limitations in robustness and expressiveness. The main objective of this thesis is to **improve the reliability of efficient visual representation learning by developing token interaction mechanisms that balance computational efficiency, robustness, and expressiveness**. This objective is addressed through three key challenges: maintaining reliable representations under visual transformations, adapting to deployment-time distribution shifts, and capturing long-range dependencies required to represent complex visual context.

The central hypothesis of this thesis is that improving computational efficiency does not necessarily require sacrificing robustness or expressiveness. Instead, these limitations depend largely on how efficient models organize and propagate information between visual tokens. By redesigning these interaction mechanisms while preserving their computational advantages,

efficient visual backbones can become more robust and expressive without returning to the quadratic cost of standard self-attention.

This main objective is decomposed into three specific objectives, each corresponding to one of the limitations identified above. **O1: Reducing the sensitivity of SSM-based vision models to geometric transformations.** We investigate how fixed sequential scan operations impose a spatial ordering on image patches that changes under rotation, producing inconsistent representations of the same visual content across different orientations. **O2: Strengthening the robustness of SSMs under deployment-time distribution shifts.** We examine how fixed traversal mechanisms propagate and accumulate domain-specific artifacts within hidden states, degrading generalization to unseen conditions. **O3: Enhancing the expressiveness of linear attention.** We overcome the limitations of single-pass aggregation, which captures mainly first-order token interactions and restricts the modeling of transitive and higher-order relationships between image regions. These three objectives define the structure of the thesis and motivate the three method chapters, each developing a solution to one of the identified limitations.

To address these objectives and investigate the central hypothesis, the thesis examines efficient token interaction from three complementary perspectives. First, at the architectural level, Spectral VMamba redesigns the spatial traversal used by visual SSMs to improve consistency under geometric transformations. Second, at deployment time, TRUST exploits traversal diversity to adapt SSM representations to distribution shifts without source-data retraining. Third, at the token-interaction level, LiSA replaces single-pass linear aggregation with stochastic higher-order propagation to increase expressive capacity while maintaining linear complexity. Together, these approaches provide a unified methodological framework for improving the robustness and expressiveness of efficient visual models while preserving their computational advantages.

Contributions

The contributions of this thesis are organized around these three directions:

- **Chapter 2**: we introduce **Spectral VMamba**, a novel State Space Model (SSM) designed to reduce the sensitivity of vision-based SSMs to geometric transformations such as rotation. Spectral VMamba organizes input patches based on spectral information derived from the graph Laplacian of image patches, leveraging eigenvectors from spectral decomposition to define a traversal order that is independent of image orientation. We also introduce the Rotational Feature Normalizer (RFN) module, which aggregates features from multiple canonical orientations of the input image to ensure rotation-consistent patch representations. Together, these components achieve patch traversal rotation invariance while outperforming leading SSM-based vision models, including VMamba and MSVMamba, on image classification; additional semantic segmentation results are reported in Appendix I.

  Related publication:

  - Spectral State Space Model for Rotation-Invariant Visual Representation Learning, published in the *IEEE / CVF Computer Vision and Pattern Recognition Conference (CVPR)*, 2025. (Dastani *et al.*, 2025b)

- **Chapter 3**: we propose **TRUST** (Test-Time Refinement using Uncertainty-Guided SSM Traverses), the first test-time adaptation (TTA) method specifically designed for Mamba-based vision models. TRUST addresses the degraded generalization of SSMs under distribution shifts by exploiting VMamba's intrinsic directional traversal mechanism. The method generates diverse traversal permutations of the four canonical scan directions, exposing the model to complementary causal perspectives of the input. An entropy-based selection strategy identifies the most confident permutations, and the Mamba-specific state-space parameters adapted under each permutation are aggregated via parameter averaging, yielding a consolidated model that combines information learned across traversal permutations. TRUST requires no source data or offline retraining, while adapting selected model parameters at test

time, and consistently outperforms existing TTA methods across seven standard benchmarks spanning corruption robustness and domain generalization.

Related publication:

- TRUST: Test-Time Refinement using Uncertainty-Guided SSM Traverses, published in the *Conference on Neural Information Processing Systems (NeurIPS)*, 2025. (Dastani *et al.*, 2025a)

- **Chapter 4**: we introduce **LiSA** (Linear Stochastic Attention), a plug-in linear-attention module that overcomes a fundamental expressiveness limitation of existing linear-attention methods: their single-pass, first-order aggregation. Standard kernelized linear-attention mixes each query with values exactly once via direct query-key affinities, failing to capture transitive dependencies between semantically related but weakly similar tokens. LiSA reformulates attention as a truncated random walk on a graph of image patches, where edge weights encode transition probabilities derived from query-key similarities. This stochastic diffusion process enables efficient higher-order token interactions while preserving linear-time complexity and numerical stability. An adaptive local attention branch grounds representations in local structure, complementing global diffusion with fine-grained context. Extensive experiments on image classification, semantic segmentation, and object detection demonstrate that LiSA consistently outperforms both softmax-attention and prior linear-attention baselines across diverse Vision Transformer backbones.

  Related publication:

  - LiSA: Linear Stochastic Attention for Long-Range Visual Modeling, submitted to the *The IEEE/CVF Winter Conference on Applications of Computer Vision (WACV)*, 2027.

External Collaborations

During my doctoral studies, I also benefited from two external research experiences:

- **Research Chair ÉTS–Zebra Technologies (February 2024–December 2025)**, where I contributed to applied research projects at the intersection of machine learning and industrial AI, with a focus on large language models and their practical deployment.
- **Borealis AI (January 2026–May 2026)**, where I contributed to research on efficient large language model inference. This work extended the broader theme of efficiency studied in this thesis beyond visual representation learning and led to a submission to NeurIPS 2026.

Thesis Organization

The remainder of this thesis is organized as follows. Chapter 2 investigates the robustness of visual state-space models to geometric transformations and introduces Spectral VMamba. Chapter 3 addresses robustness under deployment-time distribution shifts and presents TRUST, a test-time adaptation framework designed for Mamba-based vision models. Chapter 4 addresses the expressiveness limitations of linear attention and introduces LiSA for efficient higher-order visual interactions. The final chapter summarizes the main findings, discusses the broader implications of the efficiency–robustness–expressiveness perspective developed throughout the thesis, and outlines directions for future research.

CHAPTER 1

LITERATURE REVIEW

1.1 Overview of Efficient Vision Models

The development of efficient visual backbones is motivated by the growing computational demands of modern vision models, particularly as they are applied to high-resolution inputs and dense prediction tasks. In vision Transformers, this challenge is closely tied to the self-attention mechanism, which provides global pairwise interactions among visual tokens but becomes increasingly expensive as the sequence length grows. This has motivated a broad body of work on efficient alternatives to standard self-attention and Transformer-based visual backbones.

These methods can be broadly organized into three directions. The first direction reduces the cost of token interactions while preserving the full sequence, as in state space models and linear attention. The second direction reduces the sequence length itself through hierarchical downsampling, token pruning, or token merging. The third direction targets model size through post-hoc compression techniques such as quantization, pruning, knowledge distillation, or low-rank factorization. While these approaches improve scalability, they can also change how visual information is represented and propagated across the network. In this thesis, we focus on the first direction and study how efficient interaction mechanisms affect not only computational cost, but also the robustness and expressiveness of visual representations. We also briefly survey sequence reduction and model compression techniques for completeness.

1.2 Efficient Interaction Modeling

Efficient interaction modeling addresses the quadratic cost of self-attention by modifying the interaction mechanism itself rather than reducing the number of tokens or the size of the model. The methods surveyed here replace or approximate the self attention mechanism while retaining the full token sequence. In this section, we discuss two approaches: state space models, which replace attention entirely with a recurrence-based sequence map; and linear attention, which

approximates the full attention matrix in linear time. For each approach, we examine how it improves computational efficiency, how it changes information propagation across visual tokens, and what limitations it introduces.

1.2.1 State Space Models

Background. State space models (SSMs) originate from classical control theory and describe a linear time-invariant system that maps an input sequence $x(t)$ to an output sequence $y(t)$ through a latent state $h(t)$:

$$\dot{h}(t) = Ah(t) + Bx(t), \quad y(t) = Ch(t) + Dx(t), \tag{1.1}$$

where A, B, C, and D are system matrices that are parameterized and learned in deep SSMs. For sequence modeling, this continuous-time system is discretized using a step size Δ, yielding a recurrent update that can also be expressed as a convolution kernel, enabling efficient parallel training. Prior deep SSMs showed promise for modeling long-range dependencies when the state matrix A was chosen appropriately, but they were limited by computational and memory costs as well as the difficulty of parameterizing A in a stable and effective way.

S4, S5, and H3. The Structured State Space Sequence (S4) model (Gu, Goel & Ré, 2021a) was among the first to demonstrate that SSMs can match or exceed Transformers on long-range sequence benchmarks, by restricting A to a Diagonal Plus Low-Rank (DPLR) form initialized with the HiPPO matrix (Gu, Dao, Ermon, Rudra & Ré, 2020). This parameterization stabilizes training and admits Fast Fourier Transform (FFT)-based convolution, reducing complexity from $O(N^2)$ to $O(N \log N)$. Despite its effectiveness, S4's DPLR parameterization is complex to implement and difficult to parallelize efficiently on modern hardware. S5 (Smith, Warrington & Linderman, 2022) addresses this by replacing DPLR with a fully diagonal A and introducing a parallel scan (Blelloch, 1990), achieving competitive performance with significantly lower implementation complexity. However, both S4 and S5 treat all tokens uniformly, limiting their ability to selectively recall or compare tokens across the sequence. H3 (Fu *et al.*, 2022) fills this gap by interleaving SSM layers with multiplicative interactions

inspired by linear attention, enabling explicit token recall and comparison at near-Transformer quality while maintaining sub-quadratic cost.

Mamba. While H3 partially addresses content-dependent processing through multiplicative interactions, a more fundamental limitation persists across S4 and its successors: the transition matrices A, B, C remain input-independent, applying the same recurrence dynamics to every token regardless of its content. This prevents the model from selectively retaining or discarding information based on the input, a capability critical for many sequence modeling tasks.

Mamba (Gu & Dao, 2023) addresses this by introducing a *selective scan* (S6), where B, C, and the discretization step Δ are computed as functions of the input. As illustrated in Figure 1.1, this allows the model to dynamically control how much information is retained or discarded at each step, effectively giving the recurrence content-dependent filtering behavior. Because a content-dependent Δ breaks the time-invariance required for the convolutional view, Mamba cannot be trained as a global convolution. Instead, it relies on a hardware-aware parallel scan that achieves high GPU utilization while maintaining $O(N)$ memory. On language modeling benchmarks, Mamba scales to millions of tokens with linear time and memory while matching Transformer quality on standard benchmarks.

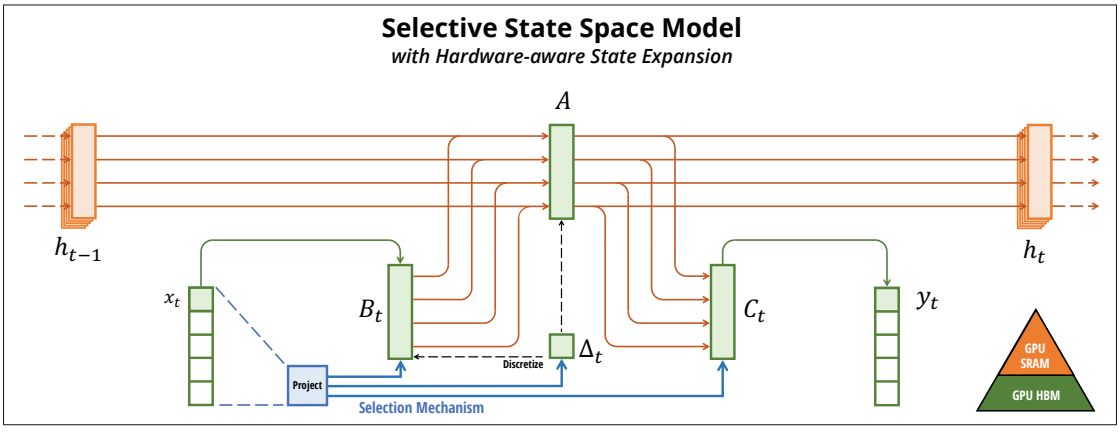


Figure 1.1 Overview of the Mamba architecture
Taken from (Gu & Dao, 2023)

SSMs in vision. Adapting Mamba to vision is non-trivial because images are two-dimensional and non-causal, whereas Mamba's selective scan is inherently defined over ordered 1D sequences. A naive row-major flattening introduces directional bias, breaking the spatial symmetry expected of a visual backbone.

VMamba (Liu *et al.*, 2024b) addresses this with the *Cross-Scan* module, 2D Selective Scan (SS2D), which traverses each image patch four times along predefined directions (Figure 1.2). The four resulting sequences are processed independently and their outputs are aggregated back into the spatial grid. This cross-scan design removes the directional preference of a single scan and gives every token a global receptive field along multiple orientations without introducing self-attention. VMamba adopts a hierarchical backbone structure analogous to the Swin Transformer (Liu *et al.*, 2021), applying progressive downsampling and patch merging to produce multi-scale feature maps suitable for dense prediction tasks.

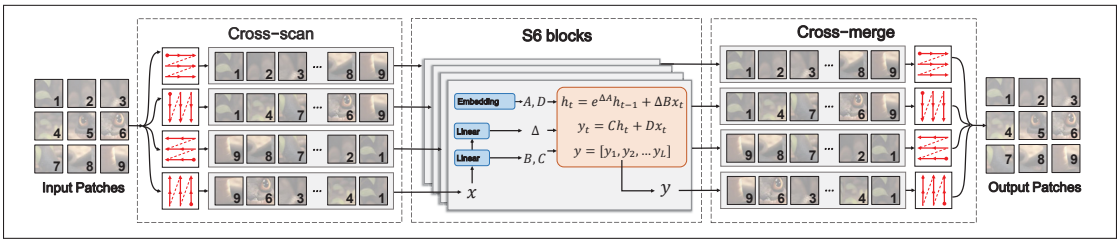


Figure 1.2 Overview of the VMamba architecture
Taken from (Liu *et al.*, 2024b)

While effective, VMamba's global scans force tokens from distant spatial regions to interact before neighboring tokens have been fully processed, which can degrade local feature quality. As illustrated in Figure 1.3, LocalMamba (Huang *et al.*, 2024) addresses this by partitioning each feature map into non-overlapping windows and running the selective scan within each window before aggregating information across windows in a secondary scan. This local-to-global processing order preserves neighborhood structure, improves performance on dense prediction tasks, and reduces the effective sequence length per scan.

Rather than refining the scan design, PlainMamba (Yang *et al.*, 2024) revisits the architectural assumptions of VMamba entirely. It replaces the hierarchical design with a non-hierarchical,

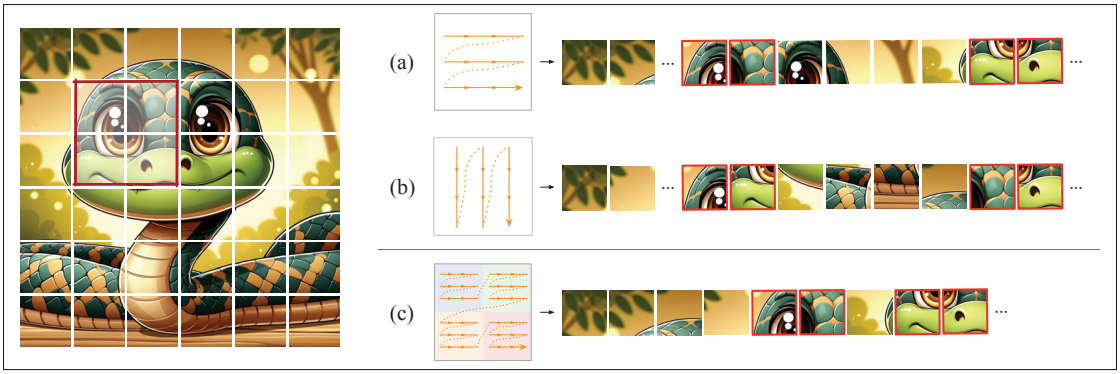


Figure 1.3 Overview of the LocalMamba
Taken from (Huang *et al.*, 2024)

isotropic backbone that keeps a single feature scale throughout the network. Instead of using a fixed traversal order, it adopts a continuous 2D scanning strategy, where scan directions are chosen based on spatial proximity. This simplification reduces design complexity while remaining competitive on standard benchmarks.

MambaVision (Hatamizadeh & Kautz, 2024) takes yet another direction by questioning whether SSMs alone are sufficient. It combines Mamba blocks with standard self-attention layers in a single backbone, using Mamba in the early and mid-level stages for efficient global context modeling, and self-attention in the final stages for fine-grained pairwise interactions (Figure 1.4). This hybrid design outperforms both pure-Mamba and pure-Transformer baselines, suggesting the two mechanisms are complementary rather than competing.

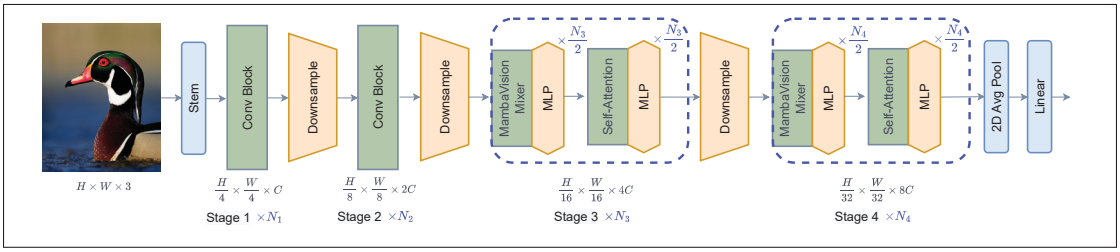


Figure 1.4 Overview of the MambaVision architecture
Taken from (Hatamizadeh & Kautz, 2024)

Beyond general vision backbones, SSMs have also been extended to specialized domains. U-Mamba (Ma *et al.*, 2024a) (Figure 1.5) incorporates Mamba blocks within a U-Net encoder-decoder architecture for biomedical image segmentation, where capturing long-range dependencies across volumetric slices is critical. By combining convolutions for local feature extraction with SSMs for long-range context, U-Mamba demonstrates strong performance on 3D medical segmentation tasks, highlighting the versatility of selective scan mechanisms beyond standard image recognition.

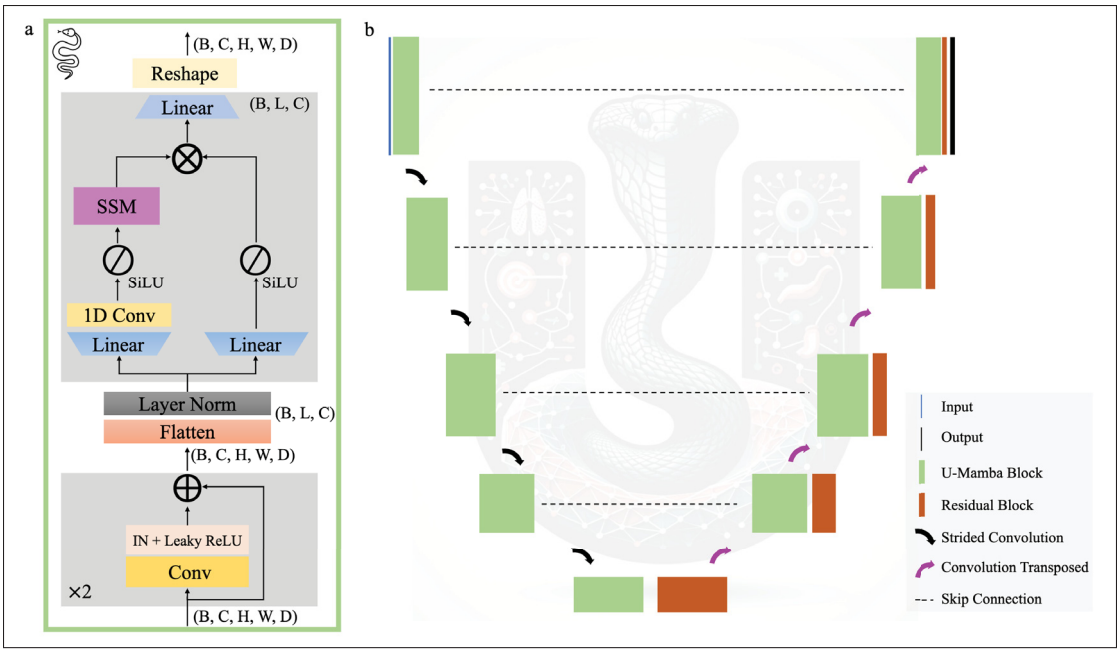


Figure 1.5 Overview of the U-Mamba architecture
Taken from (Ma *et al.*, 2024a)

Geometric transformation robustness. A limitation shared by previous scan-based vision SSMs is sensitivity to patch traversal order: rotating an input image permutes the order in which patches enter the selective scan, producing different hidden state trajectories and thus different output representations. This is a specific instance of a broader and long-standing problem in visual recognition, achieving invariance or equivariance to geometric transformations such as rotation, reflection, and scaling. A model is *equivariant* to a transformation group G

if transforming the input produces a predictable, corresponding transformation in the output representation; it is *invariant* if the output is unchanged by the transformation.

Prior work has addressed this problem through several distinct strategies. Group-equivariant convolutional networks (Cohen & Welling, 2016) achieve equivariance by replacing standard convolutions with filters defined over a transformation group, ensuring that rotating the input rotates the feature maps in a corresponding, structured way. Steerable CNNs (Cohen & Welling, 2017; Weiler, Hamprecht & Storath, 2018; Weiler & Cesa, 2019) generalize this by constructing filter banks from irreducible representations of the symmetry group, enabling a wide range of equivariance constraints to be imposed at design time. These architectures provide exact equivariance by construction but are limited to the symmetry groups they encode and require significant architectural redesign.

A more flexible alternative is learned canonicalization. Rather than encoding symmetry directly into the architecture, the model learns to map inputs to a canonical pose before processing. Spatial Transformer Networks (Jaderberg, Simonyan, Zisserman & Kavukcuoglu, 2015) pioneered this idea by predicting a spatial transformation that aligns the input. More recent work on equivariant canonicalization (Kaba, Mondal, Zhang, Bengio & Ravanbakhsh, 2023; Mondal, Panigrahi, Kaba, Mudumba & Ravanbakhsh, 2023) provides theoretical guarantees by constructing canonicalization maps that are equivariant by design. Symmetrization strategies (Dieleman, De Fauw & Kavukcuoglu, 2016; Laptev, Savinov, Buhmann & Pollefeys, 2016; Zhou, Ye, Qiu & Jiao, 2017b; Puny *et al.*, 2021) follow a simpler route: they aggregate predictions over multiple rotated versions of the input at inference time. This achieves invariance through pooling rather than architectural symmetry, but increases inference cost in proportion to the number of orientations.

Despite extensive prior work, existing methods do not address geometric transformation sensitivity in SSM-based vision models. This issue is structurally distinct from image-level pose invariance because it arises from the permutation of patch tokens entering the selective scan, rather than from the spatial arrangement of pixels. Our work in Chapter 2 is the first to

address this gap, applying the canonicalization perspective directly to the SSM context: rather than modifying the scan architecture or augmenting the input, we use spectral decomposition to canonicalize the patch traversal order. This makes the selective scan invariant to patch-level rotations without architectural changes or additional inference cost.

Distribution shift robustness. Beyond geometric robustness, SSM-based backbones face another deployment challenge: sensitivity to distribution shift. In selective scan models, patches are processed sequentially, so the hidden state at each step depends on the accumulated information from previous patches. When the target data distribution differs from the training distribution due to changes in lighting, weather, sensor properties, or image style, the patch embeddings may shift accordingly. These shifts can then propagate through the scan trajectory and amplify across the sequence, leading to degraded performance. This challenge motivates a broad line of work on domain adaptation, domain generalization, and test-time adaptation, with TTA being particularly practical because it adapts a pretrained model at inference time using only unlabeled target data, without source data or retraining.

Tent (Wang *et al.*, 2021a), the foundational TTA method, adapts the affine parameters of batch normalization layers by minimizing the entropy of the model's softmax predictions on each incoming test batch. This is motivated by the observation that distribution shift reduces prediction confidence, and that restoring high confidence, if achieved on correctly classified samples, corresponds to good adaptation. Subsequent work refines this idea along several axes: EATA (Niu *et al.*, 2022) and OSTTA (Lee, Das, Choo & Choi, 2023) select only high-confidence samples before computing the entropy gradient, reducing noise from unreliable predictions; SAR (Niu *et al.*, 2023) adds a sharpness-aware minimization objective to find flat entropy minima that are more robust to sample variability; RoTTA (Yuan, Xie & Li, 2023) maintains class-balanced memory queues to ensure that the adaptation gradient is not dominated by frequently occurring classes in the target stream; and CoTTA (Wang *et al.*, 2022a) uses augmentation-averaged predictions as reliable soft pseudo-labels, combined with stochastic weight restoration to prevent catastrophic forgetting.

Architecture-specific TTA. While most TTA methods are architecture-agnostic, recent work has explored strategies tailored to specific model families. For ViT-based models, FOA (Niu, Miao, Chen, Wu & Zhao, 2024) adapts token embeddings as additional prompt parameters using a gradient-free optimizer, avoiding backpropagation through the full model. For vision-language models (Shu *et al.*, 2022; Karmanov, Guan, Lu, El Saddik & Xing, 2024; Osowiechi *et al.*, 2024; Vargas Hakim *et al.*, 2025), adaptation strategies exploit the alignment between image and text spaces: TPT (Shu *et al.*, 2022) minimizes entropy over augmented views of the test image while jointly optimizing a prompt appended to the text encoder; TDA (Karmanov *et al.*, 2024) builds positive and negative caches of test samples using text-based pseudo-labels; and WATT (Osowiechi *et al.*, 2024) averages weights computed over multiple text template variants to improve robustness. For SSM-based models, STAD (Schirmer, Zhang & Nalisnick, 2024) uses Markov processes to learn time-varying prototypes for classifying examples during inference, adapting only the classification head rather than the SSM parameters themselves.

Despite this extensive prior work, existing methods do not exploit the structural properties of SSM backbones for adaptation. Most approaches either treat the backbone as a black box or focus on architectural features specific to ViTs and vision-language models, leaving the scan-order diversity inherent to Mamba models unused. In Chapter 3, we address this gap by adapting the Mamba-specific state-space parameters directly. We further treat multiple scan traversals as a source of model variation and average the resulting adapted weights to obtain a flatter and more robust minimum. This requires no external augmentation, memory queues, or auxiliary supervision.

1.2.2 Linear Attention

Background. Standard self-attention computes pairwise interactions between all N tokens, yielding $O(N^2)$ time and memory complexity. Linear attention (Katharopoulos, Vyas, Pappas & Fleuret, 2020) addresses this by replacing the softmax kernel with a positive

feature map $\phi(\cdot)$ applied separately to queries and keys:

$$y_i = \frac{\sum_{j=1}^{N} \phi(q_i)\phi(k_j)^\top v_j}{\sum_{j=1}^{N} \phi(q_i)\phi(k_j)^\top} = \frac{\phi(q_i)\left(\sum_{j=1}^{N} \phi(k_j)^\top v_j\right)}{\phi(q_i)\left(\sum_{j=1}^{N} \phi(k_j)^\top\right)}. \tag{1.2}$$

This factorization allows the computation to be reordered: rather than forming the $N \times N$ attention matrix, one first computes $K^\top V \in \mathbb{R}^{d\times d}$ and then applies Q, reducing complexity from $O(N^2 d)$ to $O(Nd^2)$. The full token sequence is retained and a global receptive field is preserved, at linear cost in sequence length. However, simple feature map choices such as $\phi(x) = \mathrm{ELU}(x) + 1$ lead to substantial performance degradation relative to Softmax attention, a persistent gap that has driven the field for several years.

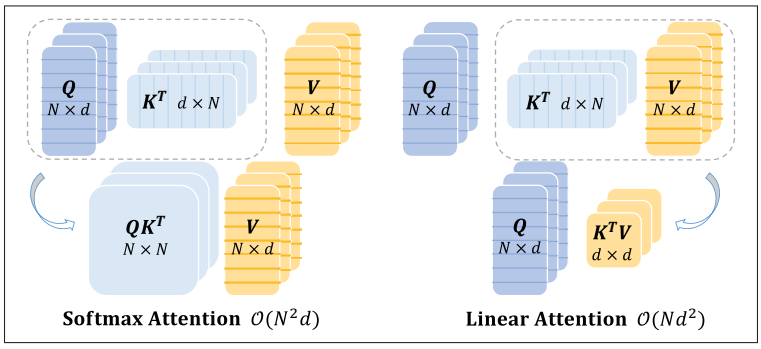


Figure 1.6 Overview of the linear attention formulation
Taken from (Han *et al.*, 2023a)

Kernel approximations. The first wave of work responded to this gap by designing better feature maps to more faithfully approximate the Softmax kernel. Performer (Choromanski *et al.*, 2021) uses orthogonal random features to construct an unbiased estimator of $\exp(QK^\top/\sqrt{d})$, providing formal approximation guarantees. Efficient Attention (Shen *et al.*, 2021) takes a simpler route by applying Softmax normalization independently along the query and key dimensions, ensuring attention scores sum to one without forming the full matrix. The overview of Efficient Attention is shown in Figure 1.7.

CosFormer (Qin *et al.*, 2022c) attributes Softmax's advantage to its data-dependent re-weighting mechanism and proposes a cosine-based substitute to replicate this property. SOFT (Lu *et al.*,

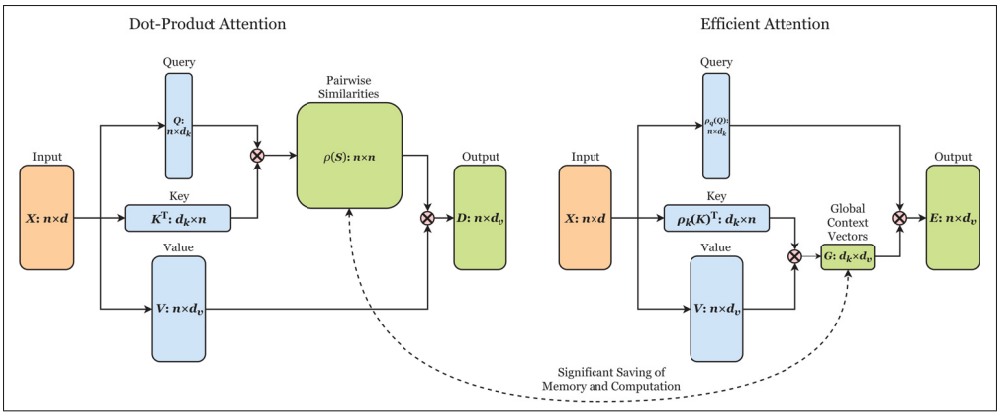


Figure 1.7 Overview of the Efficient Attention
Taken from (Shen *et al.*, 2021)

2021a) and Nyströmformer (Xiong *et al.*, 2021) resort to low-rank matrix decompositions to approximate the full attention matrix. The overview of Nyströmformer is shown in Figure 1.8.

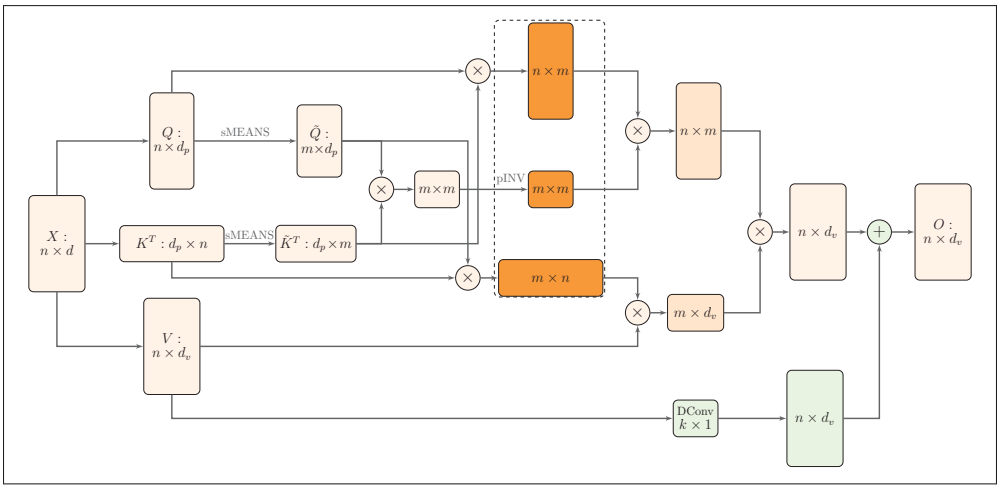


Figure 1.8 Overview of the Nyströmformer architecture
Taken from (Xiong *et al.*, 2021)

Hydra Attention (Bolya *et al.*, 2022b) replaces Softmax with cosine similarity and introduces a factorization that further reduces complexity to $O(Nd)$. The overview of Hydra Attention formulation is shown in Figure 1.9. Despite the diversity of these designs, a consistent finding across all of them is that no kernel substitution alone recovers the expressiveness of Softmax

attention, indicating that the performance gap has deeper structural roots than the choice of kernel.

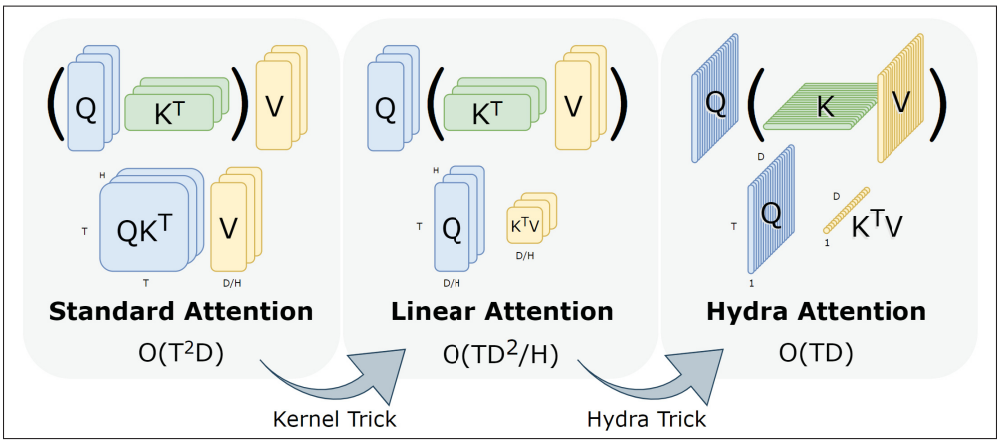


Figure 1.9 Overview of the Hydra formulation
Taken from (Bolya *et al.*, 2022b)

Diagnosing the expressiveness gap. A second line of work moves beyond designing better kernels to identify which properties of Softmax attention are missing from linear attention. FLatten Transformer (Han *et al.*, 2023a) highlights two key issues: poor focus ability, caused by the absence of Softmax's exponential sharpening, and low-rank attention maps, whose rank is bounded by the head dimension $d \ll N$. It addresses the first with a focused function f_p that sharpens queries and keys through element-wise powers and ℓ_2 normalization, and the second with a lightweight depthwise convolution (DWC) branch that restores local feature diversity. EfficientViT (Cai *et al.*, 2022) reaches a similar conclusion, using DWC to compensate for the local modeling capacity lost through linearization. TransNormer (Qin *et al.*, 2022b) instead focuses on optimization and dilution effects, showing that linear attention can suffer from gradient instability and vanishing token contributions when the normalization denominator becomes too large. MLLA (Han *et al.*, 2024b) introduces input-dependent gating, inspired by selective state space models (Gu & Dao, 2023), to recover content-dependent filtering. Together, these works show that linear attention is not simply an approximation of Softmax attention, but lacks several structural properties that must be recovered explicitly.

Integration with Softmax attention. In parallel, a different school of thought questioned whether linear attention needs to be fixed at all, and instead proposed to integrate it with Softmax attention directly. Agent Attention (Han *et al.*, 2024c) introduces a small set of learnable agent tokens $A \in \mathbb{R}^{n \times d}$ ($n \ll N$) that serve as intermediaries: agent tokens first aggregate global context from all key-value pairs via Softmax attention, and query tokens then retrieve from these aggregated representations, again via Softmax. The authors show that this two-stage process is formally equivalent to a generalized linear attention with data-dependent mapping functions $\phi_q(Q) = \sigma(QA^\top)$ and $\phi_k(K) = \sigma(AK^\top)^\top$, inheriting Softmax's expressiveness through its nonlinearity while achieving $O(Nn)$ complexity through the bottleneck of n agent tokens.

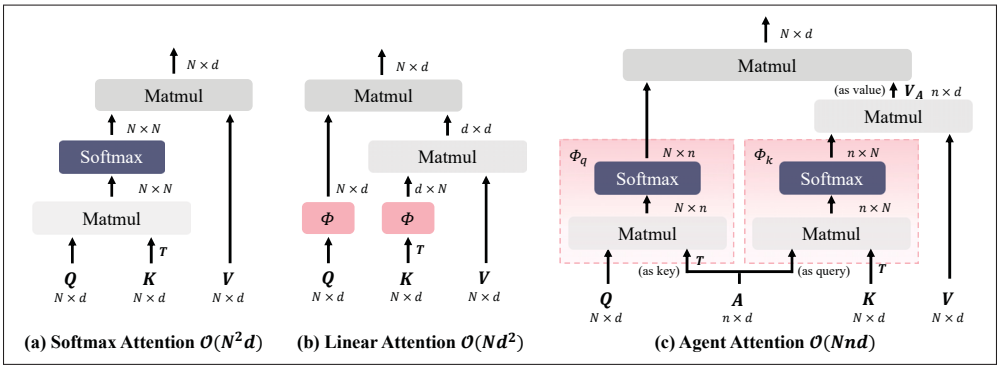


Figure 1.10 Overview of the Agent Attention formulation
Taken from (Han *et al.*, 2024c)

Castling-ViT (You *et al.*, 2023) takes a training-time perspective: it uses full Softmax attention as an auxiliary teacher during training and switches entirely to a linear angular kernel at inference, effectively distilling the rich attention patterns of Softmax into a linear module without the quadratic inference cost.

While these methods substantially reduce the performance gap, they share a fundamental structural constraint with all prior linear attention approaches: the output at each token is computed as a single weighted average of value vectors, $y_i = \phi(q_i)(K^\top V)$, which encodes only *direct*, first-order token affinities. Semantic relationships that span multiple hops, where two tokens are related through a chain of intermediate tokens, require higher-order terms that no single bilinear pass can express. This first-order bottleneck limits the model's ability to

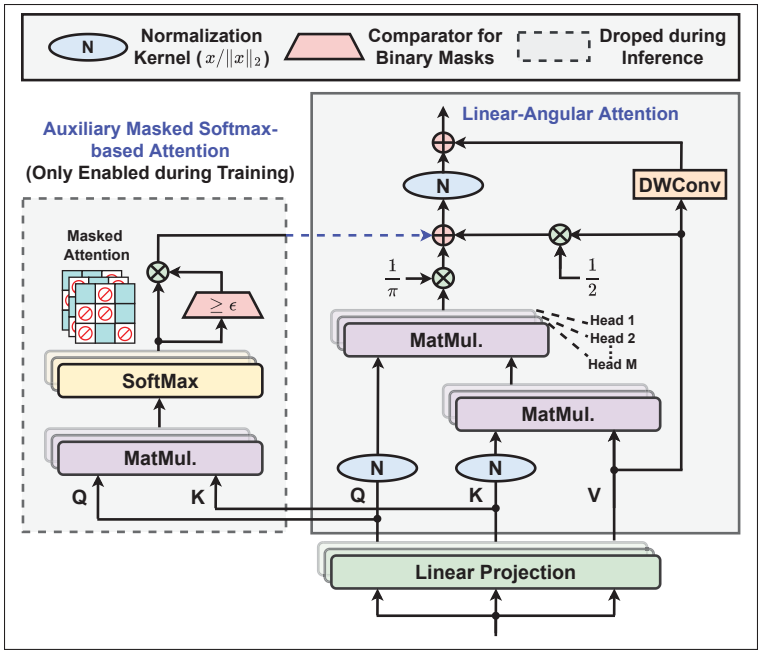


Figure 1.11 Overview of the Castling-ViT architecture
Taken from (You *et al.*, 2023)

propagate information across the token graph and capture transitive structure in the scene. In Chapter 4, we address this limitation directly by reformulating linear attention as a stochastic diffusion process over the token graph, enabling multi-hop information propagation at linear cost.

1.3 Sequence Reduction

Sequence reduction methods address the quadratic cost of self-attention by decreasing the number of tokens that the attention mechanism must process, rather than modifying the mechanism itself. In this section, we discuss three strategies for sequence reduction: hierarchical downsampling, token pruning, and token merging.

Hierarchical downsampling progressively reduces the spatial resolution of feature maps across the stages of the network, following the pyramid structure established by convolutional architectures. At each stage transition, spatial tokens are pooled or strided into a smaller set, so that deeper

layers operate on shorter sequences and therefore incur lower attention costs. Pyramid Vision Transformer (PVT) (Wang *et al.*, 2021b) introduced this design to the Transformer family by applying spatial reduction attention, where keys and values are downsampled before the attention operation. Swin Transformer (Liu *et al.*, 2021) similarly organizes tokens into a hierarchical structure through local window attention and patch merging layers that halve the spatial resolution at each stage. Because downsampling is a fixed architectural decision applied uniformly to all inputs, it reduces computational cost predictably but without accounting for the varying informativeness of different spatial regions.

Token pruning addresses this limitation by dynamically discarding tokens that are deemed uninformative during inference. Relevance scores are typically derived from attention weights or similarity to the classification token, and tokens falling below a threshold are dropped and never processed by subsequent layers. DynamicViT (Rao *et al.*, 2021) learns a lightweight prediction module that produces token-level keep or discard decisions in a differentiable manner, allowing the pruning policy to be trained end-to-end with the main classification objective. EViT (Liang *et al.*, 2022) similarly identifies inattentive tokens using the attention scores of the class token and fuses them into a single summary token rather than discarding them entirely. The fundamental limitation of pruning is that once a token is removed, its information is permanently lost and cannot be recovered by later layers, which can harm performance on dense prediction tasks where spatial completeness is essential.

Token merging offers a softer alternative by combining similar tokens into a single representative token rather than discarding them. Token Merging (ToMe) (Bolya *et al.*, 2022a) identifies pairs of similar tokens using bipartite matching on their key vectors and averages each matched pair into one, progressively reducing sequence length across layers without any learned parameters or training overhead. Because information is aggregated rather than discarded, token merging is generally more information-preserving than pruning, though the merged token remains an approximation of the originals and introduces a form of lossy compression that accumulates across layers.

Collectively, these methods reduce the effective sequence length seen by the attention mechanism, but leave its quadratic complexity unchanged. The speedup is therefore indirect and bounded by how aggressively tokens can be removed without sacrificing accuracy.

1.4 Model Compression

Model compression includes post-hoc and training-time techniques that reduce the cost of an existing model without fundamentally changing its architecture. Here, we focus on four common strategies: quantization, pruning, knowledge distillation, and low-rank factorization.

Quantization and pruning reduce model cost by making the existing network cheaper to store or execute. Quantization lowers the numerical precision of weights and activations, often from 32-bit floating point to 8-bit integers or lower, reducing memory usage and enabling faster computation on suitable hardware. Post-Training Quantization (PTQ) applies this after training, while Quantization-Aware Training (QAT) simulates low-precision operations during training to reduce accuracy degradation (Krishnamoorthi, 2018; Jacob *et al.*, 2018). In vision Transformers, quantization is especially challenging in self-attention, where operations such as softmax can be sensitive to numerical perturbations, motivating attention-specific quantization schemes (Li *et al.*, 2022e).

Pruning instead removes redundant model components. While unstructured pruning zeros individual weights and can achieve high compression, its irregular sparsity is difficult to accelerate on standard hardware. Structured pruning is often more practical because it removes regular components such as attention heads, neurons, or layers, producing smaller dense models that better translate to hardware speedups (Michel, Levy & Neubig, 2019). Prior work has shown that many attention heads can be removed with limited performance loss, suggesting redundancy in multi-head attention (Michel *et al.*, 2019).

Knowledge distillation and low-rank factorization improve efficiency by training smaller or more parameter-efficient models. Knowledge distillation transfers knowledge from a large teacher model to a smaller student by using the teacher's soft predictions or intermediate features as

supervision (Hinton, Vinyals & Dean, 2015). These signals provide richer information than hard labels alone, since they capture the teacher's uncertainty and inter-class relationships. In vision Transformers, DeiT showed that distillation from a convolutional teacher can make ViT training more data-efficient (Touvron *et al.*, 2021), while TinyViT further used distillation from large pretrained models to obtain compact Transformers with strong accuracy-efficiency trade-offs (Wu *et al.*, 2022).

Low-rank methods exploit the fact that weight updates can often be represented in a low-dimensional subspace. For example, LoRA freezes the pretrained weight matrix $\mathbf{W} \in \mathbb{R}^{d_{\text{out}} \times d_{\text{in}}}$ and learns a low-rank update $\Delta\mathbf{W} = \mathbf{AB}$, where $\mathbf{A} \in \mathbb{R}^{d_{\text{out}} \times r}$ and $\mathbf{B} \in \mathbb{R}^{r \times d_{\text{in}}}$ with $r \ll \min(d_{\text{out}}, d_{\text{in}})$ (Hu *et al.*, 2022). This reduces the number of trainable parameters from $d_{\text{out}} d_{\text{in}}$ to $r(d_{\text{out}} + d_{\text{in}})$ and is commonly applied to attention projection layers.

Although these methods can substantially reduce model size, memory usage, and latency, they generally operate on a fixed architecture. Therefore, they do not directly address the main computational bottleneck of vision Transformers: the quadratic cost of self-attention with respect to sequence length.

CHAPTER 2

SPECTRAL STATE SPACE MODEL FOR ROTATION-INVARIANT VISUAL REPRESENTATION LEARNING

Sahar Dastani[a,c] , Ali Bahri[a] , Moslem Yazdanpanah[a] , Mehrdad Noori[a] , David Osowiechi[a] , Gustavo Adolfo Vargas Hakim[a,b] , Farzad Beizaee[a] , Milad Cheraghalikhani[a] , Arnab Kumar Mondal[c,d]† , Herve Lombaert[c,f] , Christian Desrosiers[a]

[a] Department of Information Technologies Engineering, École de Technologie Supérieure
[b] Department of Systems Engineering, École de Technologie Supérieure
1100 Notre-Dame West, Montreal, Quebec, Canada H3C 1K3
[c] Mila – Quebec AI Institute
6666 Saint-Urbain Street, Suite 200, Montreal, QC H2S 3H1, Canada
[d] McGill University
845 Sherbrooke Street West, Montreal, QC H3A 0G4, Canada
[f] Polytechnique Montréal
2500 chemin de Polytechnique, Montreal, QC H3T 1J4, Canada

Paper published in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2025

Candidate contribution. The candidate contributed to the problem formulation, theory, implementation, experiments, analysis, and manuscript preparation. Co-authors contributed through technical discussions, feedback, and revisions.

This chapter addresses Objective O1, which aims to reduce the sensitivity of State Space Models (SSMs) to geometric transformations. To achieve this objective we introduce Spectral VMamba, a rotation-robust token interaction mechanism that preserves the computational efficiency of SSMs while improving robustness to changes in image orientation.

Abstract

State Space Models (SSMs) have recently emerged as an alternative to Vision Transformers (ViTs) due to their unique ability of modeling global relationships with linear complexity. SSMs are specifically designed to capture spatially proximate relationships of image patches. However, they fail to identify relationships between conceptually related yet not adjacent patches. This limitation arises from the non-causal nature of image data, which lacks inherent directional

relationships. Additionally, current vision-based SSMs are highly sensitive to transformations such as rotation. Their predefined scanning directions depend on the original image orientation, which can cause the model to produce inconsistent patch-processing sequences after rotation. To address these limitations, we introduce Spectral VMamba, a novel approach that effectively captures the global structure within an image by leveraging spectral information derived from the graph Laplacian of image patches. Through spectral decomposition, our approach encodes patch relationships in a manner robust to image orientation, while our Rotational Feature Normalizer (RFN) promotes rotation-consistent patch traversal. Our experiments on classification tasks show that Spectral VMamba outperforms the leading SSM models in vision, such as VMamba, while demonstrating strong robustness across rotations. The implementation is available at: https://github.com/Sahardastani/spectral_vmamba.git.

2.1 Introduction

Visual representation learning is fundamental to computer vision, where precise feature extraction is essential for tasks such as image classification, detection, and segmentation. The ability to extract meaningful patterns directly impacts performance and unlocks deeper insights into visual tasks. Significant advances in representation learning have been driven by the development of Convolutional Neural Networks (CNNs) (Simonyan & Zisserman, 2014; He, Zhang, Ren & Sun, 2016; Huang, Liu, Van Der Maaten & Weinberger, 2017; Tan & Le, 2019; Liu *et al.*, 2022b) and Vision Transformers (ViTs) (Dosovitskiy, Beyer, Kolesnikov *et al.*, 2021; Liu *et al.*, 2021; Zhang *et al.*, 2023; Touvron *et al.*, 2021). Although CNNs are effective at capturing local patterns with linear scaling, they struggle to efficiently model global relationships. ViTs address this limitation by introducing self-attention mechanisms, which capture global dependencies by analyzing relationships between all input elements (Vaswani, Shazeer, Parmar *et al.*, 2017; Dosovitskiy *et al.*, 2021). However, the quadratic complexity of ViTs poses challenges for large-scale data processing, as computational costs increase with input size.

State Space Models (SSMs) (Gu *et al.*, 2021a; Fu *et al.*, 2022; Smith *et al.*, 2022) have recently gained attention as a compelling alternative to traditional architectures, offering global receptive

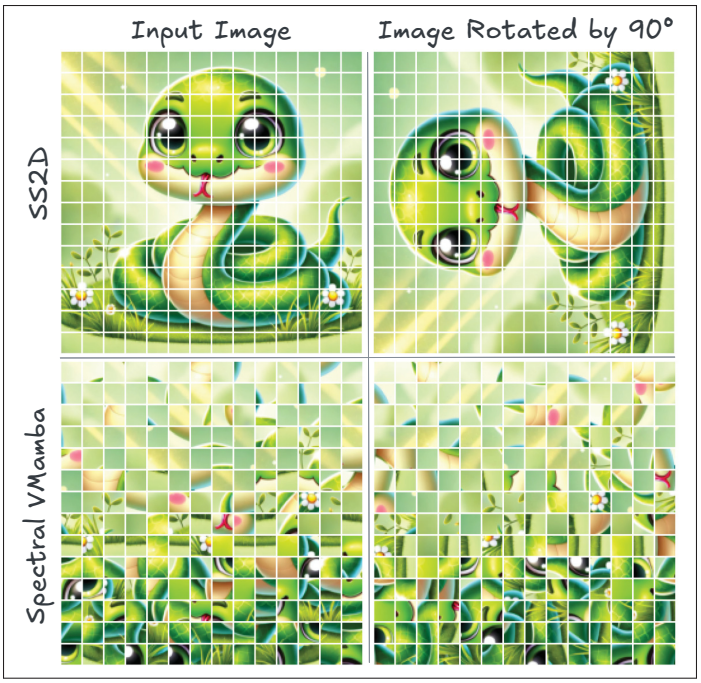


Figure 2.1 Effect of image rotation on patch processing in VMamba and Spectral VMamba networks. In VMamba networks (first row), patches are traversed using predefined horizontal and vertical scanning routes. As a result, rotating the image by 90° significantly changes the sequence in which patches are processed. In contrast, Spectral VMamba (second row) organizes patches using spectral information derived from a graph Laplacian of the image patches. As shown in the second row, Spectral VMamba produces a consistent traversal pattern, processing background patches before snake patches in both the original and rotated images. The snake image in this analysis is AI-generated

fields with linear scalability. Inspired by the achievements of Mamba (Gu & Dao, 2023) in language modeling, recent studies (Liu *et al.*, 2024b; Zhu *et al.*, 2024; Shi, Dong & Xu, 2024; Hatamizadeh & Kautz, 2024) have explored the application of linearly complex global receptive fields to the visual domain. However, unlike textual data, image pixels do not have a sequential dependency. Building on the parallelized selective scan operation in Mamba, known as S6 (Gu & Dao, 2023), which processes sequential one-dimensional data, VMamba (Liu *et al.*, 2024b) introduced 2D Selective Scan (SS2D), a novel four-way scanning mechanism tailored for spatial domain traversal. Vision Mamba (Zhu *et al.*, 2024) further enhanced sequence modeling by employing a bidirectional processing strategy that sums both forward and reverse passes. This process enables the model to capture comprehensive dependencies across an image.

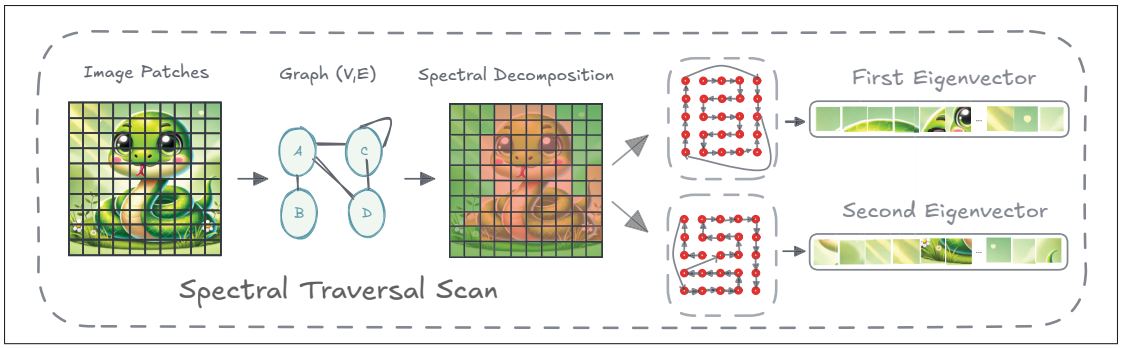


Figure 2.2 The Spectral Traversal Scan (STS) architecture begins by representing image patches as a graph. The Laplacian spectrum of this graph is then generated, and its eigenvectors are computed. Traversal paths for the image patches are obtained by ordering patches according to selected Laplacian eigenvectors. The leading non-trivial eigenvectors capture coarse structural partitions of the image, while subsequent eigenvectors progressively capture finer variations in patch relationships

Expanding upon these approaches, MSVMamba (Shi *et al.*, 2024) incorporates a multiscale 2D scanning technique, which processes both original and down-sampled feature maps. This strategy improves capturing long-range dependency while minimizing computational costs. Finally, MambaVision (Hatamizadeh & Kautz, 2024) introduces a hybrid approach. Early stages employ CNNs for fast feature extraction and later stages use Mamba and Transformer blocks to capture both local and global visual features.

Vision-based SSMs are primarily designed to capture relationships between spatially adjacent patches, focusing on spatial proximity or neighborhood-based interactions. However, SSMs are not inherently equipped to identify relationships between patches that are conceptually related but not spatially adjacent. This limitation arises due to the non-causal nature of image data, where pixels lack inherent temporal order or directional relationships.

In addition, current vision-based SSMs tend to exhibit high sensitivity to spatial information. For instance, VMamba uses predefined traversal routes that cause the network to indiscriminately switch between foreground and background regions within an image. This approach makes the network vulnerable to perturbations and leads to confusion in assessing feature importance. By treating all regions equally, VMamba fails to prioritize the features most relevant to the

task, particularly when the foreground contains critical information. This can lead to poor performance when analyzing images with complex compositions.

Another notable challenge emerges when SSMs attempt to process images that undergo transformations, such as rotation. Vision-based SSMs typically process images by (1) unfolding image patches into sequences, and then (2) scanning these sequences in specific directions, such as horizontally and vertically. The sequence of image patches depends on the image orientation, meaning that rotating the image alters the order of the sequence. Similarly, the predefined scanning directions, which are fixed to the image original orientation, result in different scan paths when the image is rotated. Consequently, SS2D struggle to accommodate transformations such as rotation. Consider Figure 2.1 (first row): when the image is rotated by 90 degrees, the predefined and fixed traversal mechanism of VMamba significantly alters the sequence in which patches are processed.

To address these limitations, we propose **Spectral VMamba**, which organizes input patches based on the spectral information derived from a graph Laplacian of image patches. By leveraging the eigenvectors from spectral decomposition of an affinity matrix, our approach captures the global structure and relationships within the image. This eigenvector-driven traversal clusters similar patches together, even if they are not spatially adjacent, thereby preventing indiscriminate transitions between foreground and background regions. Moreover, the Laplacian spectrum preserves the underlying patch relationships under rotations, while our RFN module leverages the associated spectral representation to provide a rotation-consistent basis for patch traversal. Hence, our approach defines relationships between patches by their content and mutual associations rather than by their spatial arrangement. Figure 2.1 (second row) illustrates that Spectral VMamba preserves a consistent patch-processing order under the shown rotation. As we can see, the patches representing the background are processed first, followed by those representing the foreground (the snake in Fig. 2.1).

Our main contributions are summarized as follows:

- We propose a novel technique named Spectral VMamba based on spectral graph analysis for traversing image patches. By leveraging the eigenstructure of graphs that model the relationships between patches, our approach ensures that spatially-coherent structures in image data are maintained. Our method also improves the performance of SSMs in capturing and interpreting complex data patterns.
- We introduce the Rotational Feature Normalizer (RFN) module, designed to enhance the model's robustness and consistency under image rotations.
- Our method demonstrates a notable improvement on the image classification task, outperforming the current state-of-the-art SSM-based model in computer vision.

2.2 Related Work

State Space Models (SSMs). Transformers have significantly improved the ability to manage long-range dependencies. However, their self-attention mechanism is computationally expensive, especially for high-resolution images, due to the quadratic scaling with input size. To address these challenges, SSMs have emerged as an effective tool for long sequence modeling (Gu *et al.*, 2021b; Gu *et al.*, 2021a; Smith *et al.*, 2022; Mehta, Gupta, Cutkosky & Neyshabur, 2022). In particular, the structured state-space sequence (S4) (Gu *et al.*, 2021a) model stands out for its efficient handling of long-range dependencies through a diagonal parameterization, which alleviates computational issues seen in earlier models.

Recent advancements in sequence modeling have been driven by innovations in parallel scanning techniques, inspired by models such as S4, S5 (Smith *et al.*, 2022) and the H3 model (Fu *et al.*, 2022). Mamba (Gu & Dao, 2023) improved the S4 model by integrating a data-dependent selection mechanism. This mechanism allows Mamba to achieve linear scalability, ensuring robust performance while accommodating longer sequence lengths without the exponential increase in resource demands. Such improvements are essential for applications requiring real-time or large-scale sequence processing, including Natural Language Processing (NLP) and time-series forecasting.

SSMs in vision. The success of Mamba models in the NLP domain has stimulated significant progress in computer vision, leading to various adaptations tailored for visual data processing. Several studies have extended Mamba architectures to address specific challenges in vision tasks. For instance, VMamba (Liu *et al.*, 2024b) introduced an SS2D module to tackle direction sensitivity between non-causal 2D images and ordered 1D sequences. LocalMamba (Huang *et al.*, 2024) focuses on localized feature extraction, utilizing state-space layers to enhance neighborhood interactions and reduce dependence on self-attention, leading to efficient high-resolution image analysis. QuadMamba (Xie, Zhang, Wang & Ma, 2024) uses a quadrilateral traversal mechanism to manage directional dependencies in 2D images, integrating features from different image segments.

Hybrid architectures. Recent work has also focused on combining SSMs with traditional CNNs or ViTs, leveraging the respective strengths of these models. For example, U-Mamba (Ma *et al.*, 2024a) merges CNNs and state space models to address long-range dependencies in biomedical image segmentation. MambaVision (Hatamizadeh & Kautz, 2024) integrates Mamba architectures with convolutional neural networks to improve performance across various vision tasks by leveraging both sequential modeling and spatial feature extraction. Finally, the TranS4mer model (Smith *et al.*, 2022) integrates S4 with self-attention mechanisms to achieve state-of-the-art performance in movie scene detection. These advancements collectively demonstrate the versatility and scalability of structured state space models across various applications in the vision domain.

Rotation-invariance in vision. Numerous approaches have been developed to embed equivariance and invariance directly within neural network architectures. Group-equivariant convolutional networks explicitly encode rotation symmetry into their design, providing intrinsic equivariance to rotations through steerable or group-convolutional filters (Cohen & Welling, 2016, 2017; Weiler *et al.*, 2018; Weiler & Cesa, 2019). Learned canonicalization methods, such as Spatial Transformer Networks, address invariance by predicting transformations to consistently align input images into a canonical orientation, effectively standardizing pose variability (Jaderberg *et al.*, 2015; Kaba *et al.*, 2023; Mondal *et al.*, 2023). Alternatively, symmetrization strategies

explicitly aggregate feature responses across multiple rotated inputs or filters, achieving invariance by pooling orientation-specific activations (Dieleman *et al.*, 2016; Laptev *et al.*, 2016; Zhou *et al.*, 2017b; Puny *et al.*, 2021). Additionally, capsule networks explicitly represent the pose of the object, leveraging pose-aware activations to enhance robustness against rotations (Sabour, Frosst & Hinton, 2017; Hinton, Krizhevsky & Wang, 2018). Within this landscape, our current work aligns with learned canonicalization approaches by leveraging spectral decomposition to canonicalize patch ordering and achieve patch traversal rotation invariance.

2.3 Method

This section begins with an overview of key concepts in SSMs and spectral graph analysis in Section 2.3.1, forming the foundation for our approach. Next, in Section 2.3.2, we introduce our proposed architecture, Spectral VMamba, along with the RFN module designed to promote consistent feature representations under rotation (Section 2.3.3). Finally, we discuss the robustness of our method to rotational variations in Section 2.3.6.

2.3.1 Preliminaries

SSM formulation. SSM-based models (Gu *et al.*, 2021a; Gu & Dao, 2023) map a one-dimensional function or an input sequence $x(t) \in \mathbb{R}$ to an output sequence $y(t) \in \mathbb{R}$ through a learnable hidden state $\mathbf{h}(t) \in \mathbb{R}^N$, where N denotes the hidden-state dimension. This mapping follows a continuous dynamical system formulated as

$$\frac{d}{dt}\mathbf{h}(t) = \mathbf{A}\mathbf{h}(t) + \mathbf{B}x(t), \quad y(t) = \mathbf{C}\mathbf{h}(t) \tag{2.1}$$

where $\mathbf{A} \in \mathbb{R}^{N\times N}$, $\mathbf{B} \in \mathbb{R}^{N\times 1}$ and $\mathbf{C} \in \mathbb{R}^{1\times N}$ are parameters governing the evolution of system.

To adapt continuous SSMs in deep learning frameworks, discretization is required. This can be achieved using the zero-order hold rule with a time scale parameter $\Delta \in \mathbb{R}$. Based on this rule,

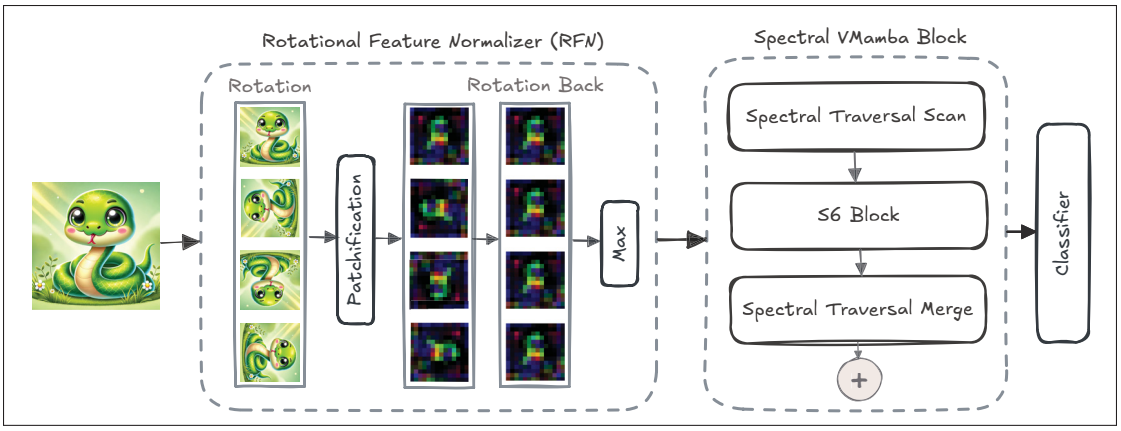


Figure 2.3 An overview of the proposed method. Our network consists of two primary components: the RFN module, which normalizes feature orientation to ensure consistency across different perspectives, and a series of Spectral VMamba blocks that process data in three stages: (1) Spectral Traversal Scan (STS), (2) S6 block (selective scan), and (3) Spectral Traversal Merge (STM). In STS, patches are reordered based on spectral coherence derived from the Laplacian spectrum. Each sequence is then processed in parallel by dedicated S6 blocks, capturing localized patterns, and finally merged in STM to reconstruct the spatial layout, producing the final feature map

the discretized versions of $\mathbf{A}$ and $\mathbf{B}$ are defined as

$$\begin{aligned} \overline{\mathbf{A}} &= \exp(\Delta\mathbf{A}), \\ \overline{\mathbf{B}} &= (\Delta\mathbf{A})^{-1}\left(\exp(\Delta\mathbf{A}) - \mathbf{I}\right)\Delta\mathbf{B} \approx \Delta\mathbf{B} \end{aligned} \tag{2.2}$$

and the system becomes

$$\mathbf{h}(t) = \overline{\mathbf{A}}\mathbf{h}(t-1) + \overline{\mathbf{B}}x(t), \quad y(t) = \mathbf{C}\mathbf{h}(t). \tag{2.3}$$

As $\overline{\mathbf{A}}$ and $\overline{\mathbf{B}}$ are linear operators, the auto-regressive formulation of Eq. (2.3), computing the output at each time step t in a recurrent manner, can be unrolled and expressed as single convolution with kernel $\overline{\mathbf{K}} \in \mathbb{R}^L$, where L denotes the sequence length:

$$\mathbf{y} = \mathbf{x} \odot \overline{\mathbf{K}}, \quad \overline{\mathbf{K}} = \left(\mathbf{C}\overline{\mathbf{B}}, \mathbf{C}\overline{\mathbf{A}\mathbf{B}}, \ldots, \mathbf{C}\overline{\mathbf{A}}^{L-1}\overline{\mathbf{B}}\right). \tag{2.4}$$

During training, this convolutional can be employed to compute all output values in parallel.

Spectral Graph Analysis studies the properties of a graph by analyzing the eigen-spectrum of its Laplacian matrix, which can be seen as a discrete version of the continuous Laplace-Beltrami operator obtained by a finite difference method (Reuter, Wolter & Peinecke, 2006). Given a real-valued function $f \in C^2$ on a Riemannian manifold $\mathcal{M}$, the Laplace-Beltrami operator Δ of f is defined as $\Delta f = div(grad\ f)$ where with $grad\ f$ is the gradient of f and div denotes the divergence.

The Laplacian Eigenvalue Problem corresponds to solving the Helmholtz equation $\Delta f = -\lambda f$, whose solutions (eigenfunctions) are directly related to the vibration of a physical membrane (Bayliss, Goldstein & Turkel, 1983). Following Courant's Nodal Line Theorem (Courant & Hilbert, 1937), the Laplacian eigen-spectrum is composed of non-negative eigenvalues $0 \leq \lambda_1 \leq \lambda_2 \leq \ldots$, each one repeating according to its multiplicity. Moreover, the eigenfunction corresponding to λ_n has at most n poles of vibration, hence smaller eigenvalues and their associated eigenvector encode lower frequency structural information which captures relationships at a more global scale.

The Laplacian matrix of a graph $G = (V, E)$ with nodes V and edges E is defined as $\mathbf{L} = \mathbf{D} - \mathbf{W}$, where $\mathbf{D}$ is the diagonal degree matrix with entries $D_{ii} = \sum_j W_{ij}$. A normalized version of the Laplacian is typically employed to address differences in the weight scales W_{ij} and variations in the distribution of node degrees D_{ii}. In this work, we utilize the symmetric normalized Laplacian, defined as

$$\mathbf{L}_{\text{sym}} = \mathbf{I} - \mathbf{D}^{-\frac{1}{2}}\mathbf{W}\mathbf{D}^{-\frac{1}{2}}, \tag{2.5}$$

which is positive semi-definite and possesses $|\mathcal{V}|$ non-negative real-valued eigenvalues $0 = \lambda_1 \leq \ldots \leq \lambda_{|\mathcal{V}|}$.

A key property of the graph Laplacian is that its spectral structure is preserved under permutations of corresponding image patches. In our work, we use this property to design an SSM that enhances rotational robustness, where the order in which patches are traversed is determined based on the eigenvectors corresponding to the smallest eigenvalues of the Laplacian matrix. Sign

and repeated-eigenvalue ambiguities are discussed in Section 2.3.5, and the rotation-consistency property is formalized in Section 2.3.6.

2.3.2 Spectral VMamba

Existing vision-based Mamba methods, such as VMamba (Liu *et al.*, 2024b), are based on SS2D, a four-way scanning mechanism designed for spatial domain traversal. SS2D organizes input patches along four predefined scanning routes. However, two key challenges emerge: (1) connections between semantically related but non-neighboring patches are overlooked; and (2) scanning is sensitive to isometric transformations, such as rotation. To address these limitations, we introduce the Spectral VMamba model whose architecture is summarized in Figure 2.3. This architecture consists of two main components: a RFN module ensuring consistent feature representation across orientations and a sequence of Spectral VMamba blocks forming the network body.

Data forwarding in our Spectral VMamba block comprises three key steps: Spectral Traversal Scan (STS), S6 block (selective scan), and Spectral Traversal Merge (STM). In the STS step, input patches are reordered based on the Laplacian spectrum of their underlying graph structure, ensuring traversal aligns with spectral coherence (see Section 2.3.4). The selective scan step then processes each reordered patch sequence in parallel using dedicated S6 blocks, effectively capturing localized patterns. Finally, in the STM step, the reordered traversal paths are restored to their original spatial format, merging the resulting sequences to generate the final output map. The detailed architecture of STS is depicted in Figure 2.2. The sequence of patch processing is determined by the order of eigenvectors derived from the spectral decomposition of the graph Laplacian. It is important to note that STS is only performed once for the initial layer. In subsequent layers, we apply a downsampling strategy for generating the traversal path. For a detailed explanation of the downsampling process, please refer to the supplementary material.

The graph Laplacian eigenspectrum is preserved under image rotations, providing a consistent spectral basis for patch traversal. However, the values in the Laplacian matrix are computed

using patch features which are themselves sensitive to rotation of the patch. A simple solution to this problem is to use patches containing a single pixel, however this would be impractical for two reasons. First, individual pixels encode little information (red-green-blue (RGB) color) and thus using their similarity to construct the Laplacian matrix would not be useful. More importantly, this would severely increase the computational cost of downstream operations such as the spectral decomposition of the Laplacian. To address this issue, we compute patch features using our RFN module (Section 2.3.3) designed to normalize the feature extraction process by aggregating features from multiple orientations of the input image. This approach allows for effectively handling of feature changes during rotations, preserving the desired order and mitigating orientation-dependent discrepancies.

2.3.3 Rotational Feature Normalizer (RFN)

An overview of the module is illustrated in Figure 2.3. This module rotates the input image by a pre-determined set of angles $\{\theta_i, \ldots, \theta_R\}$ and processes them with a *stem module* which serves as a feature extraction backbone for patchification. This backbone partitions the input image $\mathbf{I} \in \mathbb{R}^{H\times W\times 3}$ into patches $\mathbf{x} \in \mathbb{R}^{\frac{H}{p}\times\frac{W}{p}\times C}$, where (H, W) denote the dimensions of the input image, C is the number of feature channels, and p represents the patch size. To consolidate features from all rotated versions, each feature map is rotated back to its original orientation. This unrotation ensures that all extracted features align with the original image coordinate system. Finally, an element-wise max operation is applied across the unrotated feature maps to generate a single aggregated feature. The final aggregated feature map can be expressed as:

$$\mathbf{F}_{i,j} = \max_{r\in\{1,\ldots,R\}} \left[\mathcal{R}_{-\theta_r}\left(\text{Patchify}\left(\mathcal{R}_{\theta_r}(\mathbf{I})\right)\right)\right]_{i,j} \tag{2.6}$$

where θ_r is the angle of the r-th rotation and $\mathcal{R}_{\theta_r}$ is the transformation applying this rotation.

The choice of rotation angles for the RFN module represents a trade-off between rotational coverage and computational overhead. Using a larger number of angles increases the range of explicitly normalized orientations, potentially improving rotational robustness at

the cost of additional computation. In our implementation, we use four *canonical* angles, $\{0°, 90°, 180°, 270°\}$, corresponding to the quarter-turn rotation group. These rotations preserve pixel-grid alignment and therefore avoid interpolation during rotation and back-rotation. While this construction explicitly normalizes the four canonical orientations, our experiments further evaluate and demonstrate robustness to intermediate rotation angles.

2.3.4 Spectral Traversal Scan

In this section, we describe how the graph Laplacian eigen-spectrum is leveraged to define the traversal order of patches.

We represent the image as a graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$, where each node $v_i \in V$ corresponds to an image patch $\mathbf{x}_i$. We construct a k-nearest-neighbor (KNN) graph in feature space, where patches i and j are connected if either patch belongs to the other's k nearest neighbors according to the Euclidean distance between their feature representations $\mathbf{f}_i$ and $\mathbf{f}_j$. Following methods for spectral clustering (Ng, Jordan & Weiss, 2001) and normalized cuts (Shi & Malik, 2000), we consider a weighted adjacency matrix $\mathbf{W}$ of the graph, which is constructed using Euclidean distances between patches. Specifically, the weight W_{ij} between the nodes v_i and v_j is given by

$$W_{ij} = \exp\left(-\frac{1}{2\sigma^2}\|\mathbf{f}_i - \mathbf{f}_j\|^2\right) \tag{2.7}$$

where $\mathbf{f}_i$ is the feature vector of patch $\mathbf{x}_i$ and σ is a scaling parameter determined as the average of the Euclidean distances between patches. We set $W_{ij} = 0$ for all pairs not connected by the feature-space KNN rule above, resulting in a sparse, symmetric adjacency matrix.

We compute the eigenpairs of the symmetric normalized Laplacian $\mathbf{L}_{\text{sym}}$ and order them according to increasing eigenvalue. The first eigenvector associated with the smallest eigenvalue is excluded, and we retain the next m eigenvectors, denoted by $\mathbf{u}^{(j)} \in \mathbb{R}^{N_p}$, $j = 1, \ldots, m$, where N_p is the number of image patches. Each eigenvector assigns a scalar value $\mathbf{u}_i^{(j)}$ to every image patch i. For each selected eigenvector, we construct two traversals: one by sorting the patches

in increasing order of $\mathbf{u}_i^{(j)}$ and the other in decreasing order. The resulting $2m$ traversals are processed by the corresponding S6 branches and subsequently merged.

Patches $\mathbf{x}_i$ are ordered based on these eigenvectors, generating multiple traversal sequences:

$$\mathbf{X}^{(j,\text{rev})} = \left\{\mathbf{x}_{i_1}, \mathbf{x}_{i_2}, \ldots, \mathbf{x}_{i_{N_p}}\right\}, \tag{2.8}$$

$$i_k^{(j,\text{rev})} = \arg\min_{i\in\{1,\ldots,N_p\}} \left\{(-1)^{\text{rev}} u_i^{(j)} \mid i \notin \{i_1, \ldots, i_{k-1}\}\right\}, \qquad k = 1, \ldots, N_p.$$

where $\text{rev} \in \{0, 1\}$ indicates if the traversal is reversed (rev = 1) or not (rev = 0). This concatenated sequence $\mathbf{X}$ is then fed into the Mamba network for further processing, allowing the model to incorporate rich spectral relationships while preserving global coherence.

2.3.5 Spectral Ambiguities

Laplacian eigenvectors are defined up to sign: if $\mathbf{u}$ is an eigenvector, then $-\mathbf{u}$ is equally valid. Since STS considers both increasing and decreasing traversal orders, a sign change simply swaps these two traversals. For repeated eigenvalues, the corresponding eigenvectors are not unique; therefore, the traversal is uniquely defined only for simple eigenvalues. In practice, we sort the eigenvectors by increasing eigenvalue, discard the first one, and use the basis returned by the eigensolver for equal or nearly equal eigenvalues.

2.3.6 Rotation Invariance

Spectral decomposition provides a natural basis for constructing rotation-consistent patch traversals. Under a grid-preserving rotation, corresponding image patches are permuted while the graph structure is preserved. This property is formalized in the following theorem.

Theorem 2.3.1. *Under the assumption that patch features are rotation-consistent, i.e.,* $\mathbf{F}(\mathcal{R}_\theta(\mathbf{I})) = \mathbf{PF}(\mathbf{I})$*, the STS traversal is preserved under rotation, up to patch permutation and reversal of the traversal direction.*

Proof. Under the assumption that the patch features transform consistently under rotation, the corresponding graph weights satisfy

$$\mathbf{W}' = \mathbf{P}\mathbf{W}\mathbf{P}^T, \qquad \mathbf{L}' = \mathbf{P}\mathbf{L}\mathbf{P}^T. \tag{2.9}$$

Thus, $\mathbf{L}$ and $\mathbf{L}'$ have the same eigenvalues. If $\mathbf{L}\mathbf{u}_j = \lambda_j \mathbf{u}_j$, then

$$\mathbf{L}'(\mathbf{P}\mathbf{u}_j) = \lambda_j \mathbf{P}\mathbf{u}_j. \tag{2.10}$$

For a simple eigenvalue, the corresponding normalized eigenvector satisfies $\mathbf{u}'_j = \pm\mathbf{P}\mathbf{u}_j$. The permutation preserves the ordering of corresponding patches, while the sign change reverses the traversal direction. Since STS uses both increasing and decreasing orders, the traversal pair is preserved. This statement assumes distinct eigenvector entries; repeated eigenvalues are discussed in Section 2.3.5.

2.4 Experiments

We start by comparing our method against recently-proposed ViTs and SSMs on an image classification task. We then present an ablation study to thoroughly investigate the impact of various components on the performance of our method.

2.4.1 Image Classification

We conduct classification experiments on the *mini*ImageNet (Vinyals, Blundell, Lillicrap, Wierstra *et al.*, 2016) dataset. This dataset includes $50,000$ training images and $10,000$ validation images across 100 categories. Following previous works (Liu *et al.*, 2021, 2024b; Zhu *et al.*, 2024), we train our models for 300 epochs with a batch size of 128 per graphics processing unit (GPU). The models are optimized using the AdamW optimizer with a momentum of 0.9. A cosine decay scheduler manages the learning rate, which starts at 5×10^{-4}, alongside a weight decay of 0.05. Additionally, we apply an Exponential Moving Average (EMA) to

Table 2.1 Classification performance comparison on *mini*ImageNet. All images are of size 224×224. T, S, and B denote the tiny, small, and base scales, respectively

Model	GFLOPs	Accuracy	
		Top-1	Top-5
Transformer-Based			
DeiT-S (Touvron *et al.*, 2021)	4.6	70.83	89.74
DeiT-B (Touvron *et al.*, 2021)	17.5	72.43	90.14
Swin-T (Liu *et al.*, 2021)	4.5	83.25	95.54
Swin-S (Liu *et al.*, 2021)	8.7	84.10	95.53
Swin-B (Liu *et al.*, 2021)	15.4	82.77	95.33
XCiT-S24 (Ali *et al.*, 2021)	9.2	85.79	96.31
XCiT-M24 (Ali *et al.*, 2021)	16.2	86.80	96.38
SSM-Based			
Vim-T (Zhu *et al.*, 2024)	1.5	67.30	87.97
Vim-S (Zhu *et al.*, 2024)	5.1	79.70	93.20
LocalVim-T (Huang *et al.*, 2024)	1.5	82.12	94.60
LocalVim-S (Huang *et al.*, 2024)	4.8	81.68	93.63
MSVMamba-N (Shi *et al.*, 2024)	0.9	82.16	95.10
MSVMamba-M (Shi *et al.*, 2024)	1.5	83.72	95.81
MSVMamba-T (Shi *et al.*, 2024)	4.6	86.48	96.43
VMamba-T (Liu *et al.*, 2024b)	4.9	86.25	96.64
VMamba-S (Liu *et al.*, 2024b)	8.7	86.48	96.79
VMamba-B (Liu *et al.*, 2024b)	8.7	87.17	96.96
Ours-T	3.9	87.86 (↑1.61)	97.25 (↑0.61)
Ours-S	6.3	88.09 (↑1.61)	97.08 (↑0.29)
Ours-B	6.3	88.17 (↑1.00)	97.27 (↑0.31)

stabilize training. For input images of size 224×224, our data augmentation techniques include color jittering, AutoAugment, random erasing, mixup, and cutmix. We use the standard AutoAugment policy during training, which may include rotation transformations; no additional rotation-specific augmentation is applied.

Table 2.1 compares our Spectral VMamba method against ViTs and SSM-based models on the *mini*ImageNet dataset. As can be seen, our method outperforms all other approaches in both Top-1 and Top-5 EMA accuracy. Compared to VMamba, its closest competitor, our method

improves Top-1 accuracy by 1.61% for the tiny and small models (T and S) and by 1% for the base model (B), while requiring less FLOPs (see 2.4.2 for explanation). Moreover, our Spectral VMamba (B) model yields a 1.37% higher Top-1 accuracy than the strongest ViT-based model, XCiT-M24, and an improvement of 5.4% over Swin-B, one of the most popular architectures based on ViT.

2.4.2 Ablation Study

Rotational invariance. To evaluate the robustness to rotation of our method, we evaluated its performance on test images rotated at angles from 0° to 360° with 15° increments. As a reminder, our RFN module extracts and aggregates features for four canonical rotations: $\{0°, 90°, 180°, 270°\}$. We therefore expect the greatest consistency at these canonical angles, which are explicitly normalized by the RFN construction.

As shown in Figure 2.4, our model with the RFN (blue line) maintains a high accuracy near 87% at those canonical angles, demonstrating the intended robustness and consistency at the canonical rotations. In contrast, VMamba (orange line) suffers from a significant drop in performance for images rotated at 90° (~30% drop), 180° (~20% drop) and 270° (~30% drop). Although RFN explicitly normalizes only the canonical angles, we observe remarkably consistent performance across all rotation angles, we observe a remarkably consistent performance across all rotation angles, stabilizing at around 78% in Top-1 accuracy. On the other hand, the performance of VMamba fluctuates drastically, reaching accuracy values near 50% at certain angles (e.g., 105° or 255°).

Impact of RFN. To assess the usefulness of our RFN module, we also tested an ablation variant of our method without this module (Ours w/o RFN in Figure 2.4). As discussed above, the Laplacian eigenvalues are preserved under permutation of corresponding patches, while the associated simple eigenvectors transform equivariantly up to sign. However, the Laplacian matrix is constructed from patch features that are themselves sensitive to rotation. As we expect, removing the RFN causes the performance to drop for all angles, this drop growing as we

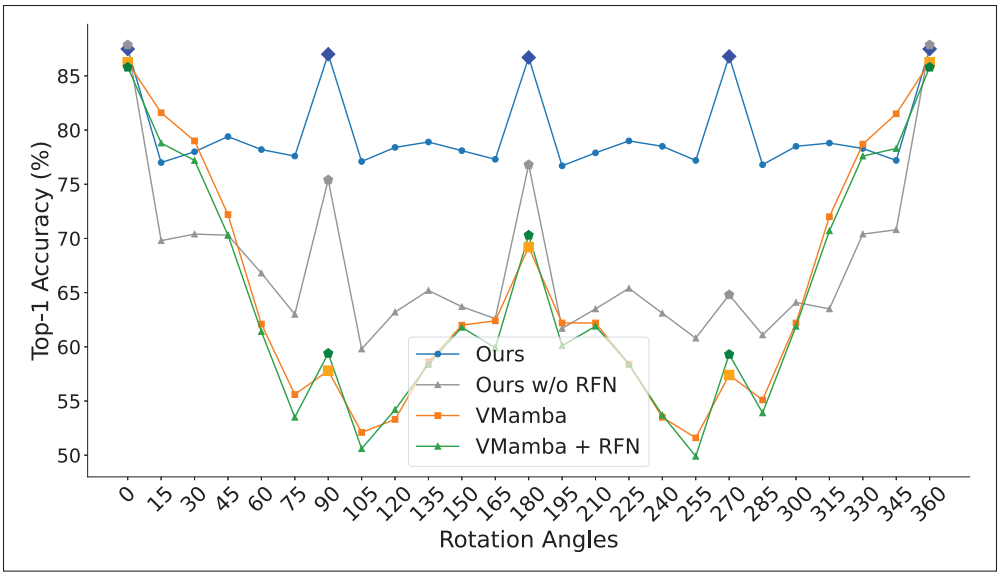


Figure 2.4 Model performance comparison across various rotation angles. Our method demonstrates consistent accuracy across all rotation angles, while VMamba exhibits significant fluctuations

increase the rotation (away from 0° or 360°). Interestingly, our method without the RFN module still achieves a higher accuracy than VMamba for angles between 60° and 300°, which represent severe rotations. This could be explained by the small size of patches (16×16) compared to the image, minimizing the impact of rotations (full invariance would be achieved for 1×1 patches).

Impact of STS. Since the STS is a critical part of our Spectral VMamba method, we cannot remove it. To evaluate its impact, we instead add the RFN to VMamba, the resulting approach now only differing from our method by not having a STS. As shown in Figure 2.4, the VMamba + RFN model (green line) offers a low robustness to rotation, its accuracy similar to that of VMamba. These results demonstrate that improving rotational invariance at the patch level is not sufficient to achieve a high performance.

Number of eigenvectors. In Figure 2.5, we assess the effect on performance of varying the number of eigenvectors used in the STS ($m \in \{1, 2, 3, 4\}$). As can be seen, accuracy improves consistently with more eigenvectors, peaking at 87.48% for $m = 4$. This suggests that eigenvectors offer complementary information and that their combination enhances the

recognition of complex global structures. In the same plot, we also report the performance of the standard VMamba model and a variant using a random traversal strategy. As can be observed, the improvement provided by our STS over VMamba, for any number of eigenvectors, is greater than the one offered by the raster scan of VMamba over a random traversal.

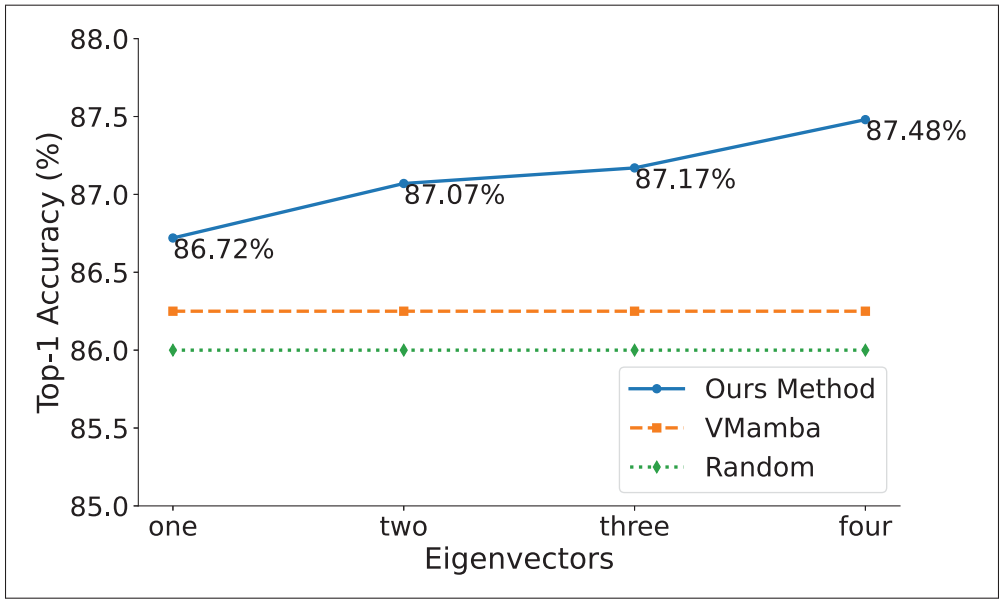


Figure 2.5 Model performance for different number of eigenvectors. Four eigenvectors exhibit the best performance

Number of nearest neighbors. The K-Nearest Neighbors (KNN) algorithm constructs a sparse adjacency matrix by selecting the top k nearest neighbors for each image patch. In Table 2.2, we report our model's performance across k values from 5 to 25. As can be seen, our model is not very sensitive to this parameter and achieves its highest accuracy of 87.48% for $k = 5$ neighbors. This suggests that only a few neighbors are necessary to capture the global structure of an image. This also has a positive impact on the efficiency of our method, since a smaller k results in a sparser Laplacian matrix and reduces the computational burden of the spectral decomposition.

Memory Usage and Runtime. Next, we analyze the computational cost and time efficiency of the STS. As mentioned before, the proposed strategy minimizes computations by leveraging

Table 2.2 Model performance for different values of k (number of nearest neighbors). The best performance is achieved at $k = 5$

k	**5**	**10**	**15**	**20**	**25**
Top-1 Accuracy (%)	**87.48**	86.98	87.14	86.91	87.15

the sparsity of the Laplacian matrix and limiting the number of computed eigenvectors. In the following experiments, we use $m = 4$ eigenvectors and batch size = 1.

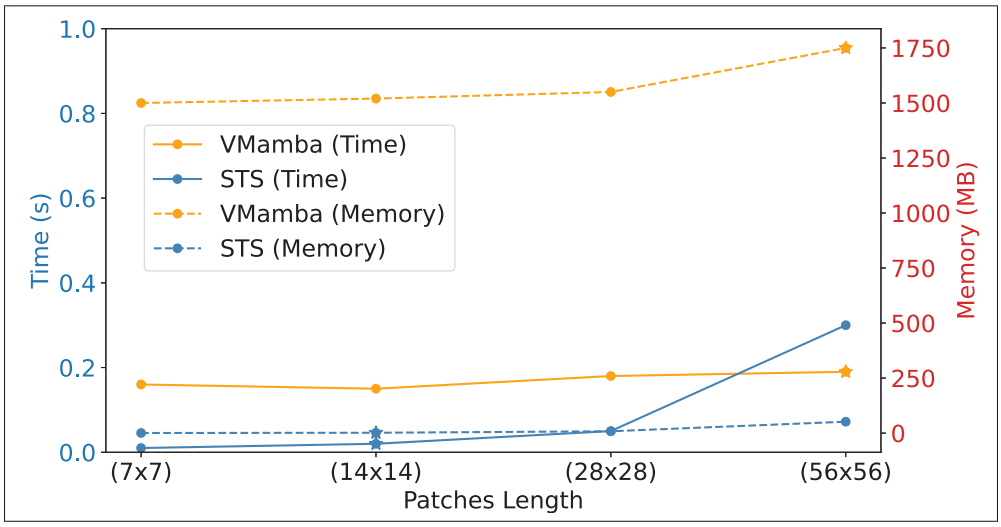


Figure 2.6 Runtime and memory usage for different patch lengths

Memory Usage: As depicted in 2.6, our STS approach (blue line) is highly efficient in terms of memory usage compared to the VMamba (orange line). Even when the number of patches increases, the STS based on a sparse eigen-solver, requires only a modest increase in memory compared to the VMamba. The stars along each line indicate the token lengths used in our method, highlighting the specific points where processing occurs across different sequence lengths.

Runtime: Our STS strategy, which can run on a central processing unit (CPU), also shows a favorable runtime scaling with increased patch length (2.6). Specifically, for a patch length of 196 (marked by a star), which is the setting used in our main experiments, the increase in

runtime is minimal when compared to the VMamba backbone, highlighting the efficiency of our approach.

FLOPs: The computational cost of our traversal strategy includes constructing the adjacency matrix, computing the Laplacian, and obtaining its leading eigenvectors. These operations require approximately 2 MFLOPs, indicating that the STS module itself introduces only a small computational overhead. The VMamba backbone requires approximately 3.9 GFLOPs. As reported in Table 2.1, the total FLOPs of Spectral VMamba are lower than those of VMamba at the same model scale. However, this reduction should not be attributed solely to STS, since Spectral VMamba operates on a 14!×!14 token representation, whereas VMamba uses 56!×!56 tokens.

Latency/Memory breakdown. We profile five components, RFN's four rotated forward passes, patch feature extraction, KNN graph construction, spectral eigensolving, and the S6 blocks, using CUDA-event timing averaged over 30 forward passes (10 warm-up iterations discarded); peak memory is measured via `torch.cuda.max_memory_allocated()`. Table 2.3 reports results at two resolutions (224^2, 384^2) and three batch sizes (1, 8, 32).

Eigensolving dominates at larger batch sizes, rising from 11.6% of end-to-end latency at batch size 1 to 62.2% at batch size 32 (224^2), while its latency grows only ~1.9× from 224^2 to 384^2 despite a ~2.9× increase in patch count, indicating favorable batched-GPU scaling. KNN graph construction stays below 3% of end-to-end latency throughout, and RFN rotation and patch feature extraction scale predictably with resolution and batch size (2.0–2.8 ms and 1.5–58.4 ms, respectively) without becoming a bottleneck. Peak memory grows from 0.99 GB to 3.49 GB across settings, driven by graph and eigenvector matrices whose size scales with patch count. Overall, the spectral components add manageable overhead at these resolutions, with eigensolving the main target for further scalability, e.g. via sparse or approximate decomposition.

Table 2.3 End-to-end latency (ms) and peak GPU memory (GB) breakdown by pipeline stage, across batch size and input resolution

Batch	Res.	Latency (ms)					End-to-end (ms)	Peak Mem. (GB)
		RFN Rot.	Patch Feat.	KNN	Eigensolve	S6 Blocks		
1	224^2	2.0	1.5	0.6	3.3	16.8	28.8	0.99
8	224^2	2.2	5.4	0.6	24.7	18.2	55.7	1.08
32	224^2	2.2	20.0	0.8	99.4	32.8	159.9	1.75
1	384^2	2.0	2.0	0.7	5.8	16.9	31.9	1.00
8	384^2	2.1	14.8	1.4	45.8	22.6	90.2	1.51
32	384^2	2.8	58.4	4.9	185.8	64.2	327.0	3.49

2.5 Conclusion

In this paper, we introduce Spectral VMamba, a novel approach to visual state space models that promotes rotation-consistent patch traversal. Our method leverages spectral decomposition from the graph Laplacian of image patches to define traversal paths for patch processing. In addition, we develop the RFN module to promote consistent feature representations across different orientations. This module serves as a foundation for our STS module, enabling patch features that are robust to rotational variations. By incorporating the RFN and STS modules, Spectral VMamba achieves a robust feature extraction pipeline that maintains high accuracy across a variety of image orientations. Our experiments demonstrate that Spectral VMamba consistently outperforms popular models including Swin Transformer, VMamba, MSVMamba, and LocalVim in image classification tasks. Our method does not yet address generalization to the medical domain. In future work, we aim to handle disconnected graph components, common in medical images with distinct anatomical structures or pathologies. Adapting spectral traversal for these regions will enhance pattern capture and robustness across imaging modalities.

CHAPTER 3

TRUST: TEST-TIME REFINEMENT USING UNCERTAINTY-GUIDED SSM TRAVERSES

Sahar Dastani[a,c*] , Ali Bahri[a*] , Gustavo Adolf Vargas Hakim[b] , Moslem Yazdanpanah[a] , Mehrdad Noori[a] , David Osowiechi[a] , Samuel Barbeau[a] , Ismail Ben Ayed[b] , Herve Lombaert[c,d] , Christian Desrosiers[a]

[a] Department of Information Technologies Engineering, École de Technologie Supérieure
[b] Department of Systems Engineering, École de Technologie Supérieure
1100 Notre-Dame West, Montreal, Quebec, Canada H3C 1K3
[c] Mila – Quebec AI Institute
6666 Saint-Urbain Street, Suite 200, Montreal, QC H2S 3H1, Canada
[d] Polytechnique Montréal
2500 chemin de Polytechnique, Montreal, QC H3T 1J4, Canada

[*]Equal Contribution
Paper published in *Advances in Neural Information Processing Systems (NeurIPS)*, December 2025

Candidate contribution. The candidate and Co-first author contributed equally overall. The candidate led the implementation, experiments, and analysis, and contributed to the problem formulation, theory, and manuscript preparation. Other co-authors contributed through discussions, feedback, and revisions.

This chapter addresses Objective O2, which aims to strengthen the robustness of State Space Models (SSMs) under deployment-time distribution shifts. To achieve this objective, we introduce TRUST, a test-time adaptation method that improves the robustness of Mamba-based vision models without requiring source data or offline retraining.

Abstract

State Space Models (SSMs) have emerged as efficient alternatives to Vision Transformers (ViTs), with VMamba standing out as a pioneering architecture designed for vision tasks. However, their generalization performance degrades significantly under distribution shifts. To address this limitation, we propose TRUST (Test-Time Refinement using Uncertainty-Guided

SSM Traverses), a novel test-time adaptation (TTA) method that leverages diverse traversal permutations to generate multiple causal perspectives of the input image. Model predictions serve as pseudo-labels to guide updates of the Mamba-specific parameters, and the adapted weights are averaged to integrate the learned information across traversal scans. Altogether, TRUST is the first approach that explicitly leverages the unique architectural properties of SSMs for adaptation. Experiments on seven benchmarks show that TRUST consistently improves robustness and outperforms the evaluated TTA baselines. The code is available at: https://github.com/Sahardastani/trust.

3.1 Introduction

The field of visual representation learning has advanced rapidly due to the ability of deep neural networks to extract rich and generalizable features. Convolutional Neural Networks (CNNs) (Simonyan & Zisserman, 2014; He *et al.*, 2016; Huang *et al.*, 2017; Tan & Le, 2019; Liu *et al.*, 2022b) excel in modeling local patterns through strong inductive biases but struggle with global context. Vision Transformers (ViTs) (Dosovitskiy *et al.*, 2021; Liu *et al.*, 2021; Zhang *et al.*, 2023; Touvron *et al.*, 2021) address this challenge using a self-attention mechanism, although at higher computational cost. More recently, State Space Models (SSMs) (Gu *et al.*, 2021a; Fu *et al.*, 2022; Smith *et al.*, 2022) emerged as a scalable alternative, providing global receptive fields with linear complexity. Despite their efficiency, the performance of SSMs degrades under distribution shift. This is mainly due to violations of the Independently and Identically Distributed (i.i.d.) assumption, which is often disrupted in real-world settings. While generalization strategies are well developed for CNNs and ViTs, vision-specific strategies for SSMs are still lacking.

This work focuses on *VMamba* (Liu *et al.*, 2024b), a vision-adapted variant of the Mamba architecture (Gu & Dao, 2023), designed for sequential visual processing. VMamba introduces 2D Selective Scan (SS2D), a four-way traversal mechanism that scans image patches along predefined spatial directions. However, this directional processing introduces a strong inductive bias by aligning internal representations with fixed traversal paths (Dastani *et al.*, 2025b), which may hinder generalization under distribution shifts. Additionally, the hidden states of VMamba

store historical context over the traversal sequence. When exposed to unseen domains, this context may accumulate domain-specific artifacts, leading to amplified bias during propagation and ultimately degrading generalization (Long *et al.*, 2024).

To address these challenges, we propose **T**est-Time **R**efinement using **U**ncertainty-Guided **S**SM **T**raverses (TRUST), a novel Test-Time Adaptation (TTA) strategy specifically designed for SSMs that leverages VMamba's internal traversal mechanism. As illustrated in the offline phase of Figure 3.1, different traversal permutations result in varying prediction entropy levels, revealing the sensitivity of the model to causal ordering. TRUST systematically generates multiple traversal permutations by reordering the four directional scans, exposing the model to diverse causal perspectives of the same input. At test time, we compute the entropy of predictions from each permutation and select the ones with the lowest entropy, which are associated with more stable and domain-robust hidden states.

Inspired by previous work on flat minima and weight averaging (Keskar, Nocedal, Tang, Mudigere & Smelyanskiy, 2017; Cha *et al.*, 2021), our method aggregates the outputs of different traversal permutations to implicitly explore a broader and flatter region of the loss landscape. Each permutation activates a distinct computational pathway through the model, effectively sampling different local minima. Selecting the top-k traversal permutations with the lowest predictive entropy and computing a average of their model parameters mitigates the impact of noisy or uncertain predictions. This enhances generalization without requiring source data or offline retraining.

We outline the main contributions of our work as follows:

- We propose TRUST, the first TTA approach specifically designed for Mamba-based vision models. Unlike prior methods that depend on data augmentation or auxiliary models, our approach takes advantage of the internal traversal dynamics of VMamba to improve generalization under distribution shift.
- TRUST introduces a parameter-averaging strategy to promote robustness, which permutes traversal directions and uniformly averages model weights from the most confident paths.

- We validate our method on **seven standard benchmarks**, where it consistently outperforms the evaluated TTA baselines.

3.2 Related Work

TTA is a paradigm of Domain Adaptation (Liang, Hu & Feng, 2020) that intends to enhance model generalization during deployment. Specifically, a pre-trained model is adapted to incoming data batches without requiring label supervision or having access to source data. As a pioneering approach, Tent (Wang *et al.*, 2021a) minimizes prediction entropy as an adaptation objective, based on the principle that distribution shifts in the target domain reduce the confidence of a model in its predictions. Examples of Tent-based approaches include using confidence thresholds for sample selection before adaptation (Niu *et al.*, 2022; Lee *et al.*, 2023), minimizing the sharpness of the entropy loss (Niu *et al.*, 2023), using class-balanced memory queues for better adaptation sampling (Yuan *et al.*, 2023), and meta-learning the entropy loss (Goyal, Sun, Raghunanthan & Kolter, 2022), among other alternatives (Gong, Kim, Lee, Chottananurak & Lee, 2024; Yu, Sheng, He & Liang, 2024; Gao, Zhang & Liu, 2024). The literature also includes several other adaptation strategies that do not minimize entropy, such as contrastive learning (Wang *et al.*, 2022a; Chen, Wang, Darrell & Ebrahimi, 2022b), Laplacian optimization (Boudiaf, Mueller, Ben Ayed & Bertinetto, 2022; Sun *et al.*, 2024), and prototype-based pseudo-labeling (Jang, Chung & Chung, 2024b). While flexible, these techniques do not capitalize on the unique properties of specific architectures such as VMamba.

Architecture-specific TTA. Recent advances in TTA have introduced a variety of model-specific strategies. Among those, FOA (Niu *et al.*, 2024) loosely adheres to ViTs by incorporating gradient-free learning of additional prompts in the form of token embeddings. For Vision-language Models (VLMs) (Shu *et al.*, 2022; Karmanov *et al.*, 2024; Osowiechi *et al.*, 2024; Vargas Hakim *et al.*, 2025), recent strategies incorporate text information as an auxiliary tool for adaptation. Examples include prompt learning via entropy minimization (Shu *et al.*, 2022), positive and negative caching with text-based pseudo-labels (Karmanov *et al.*, 2024), transductive adaptation using conformal pseudo-captions (Vargas Hakim *et al.*, 2025), and weight averaging

across text templates (Osowiechi *et al.*, 2024). SSM-oriented TTA has also been explored to a smaller extent. In a recent work, STAD (Schirmer *et al.*, 2024) uses Markov processes to learn time-varying prototypes for classifying examples during inference and adapts the classification head of the model.

Weight Averaging for Robust Adaptation. Initially introduced in the context of Domain Generalization (Li, Yang, Song & Hospedales, 2018; Zhou, Yang, Qiao & Xiang, 2021c; Noori *et al.*, 2025a), weight averaging has gained popularity as an effective technique for enhancing model robustness to domain shifts (Izmailov, Podoprikhin, Garipov, Vetrov & Wilson, 2018; Cha *et al.*, 2021). Notably, methods like SWAD (Cha *et al.*, 2021) aim to identify flat minima in the loss landscape by minimizing loss fluctuations across diverse inputs. This is often accomplished by averaging the weights of models trained on different image augmentations. While similar ideas have been adopted in the TTA literature (Niu *et al.*, 2023; Osowiechi *et al.*, 2024; Bahri *et al.*, 2025b; Yazdanpanah *et al.*, 2025; Bahri *et al.*, 2025a), they typically rely on explicit input diversification, which introduces additional computational overhead. In contrast, our approach leverages the intrinsic diversity of SSM-based traversals to induce variation directly in the weights of our model, without requiring external data augmentations.

3.3 Method

Section 3.3.1 introduces key concepts in TTA and SSM, laying the groundwork for our approach. In Section 3.3.2, we propose traversal permutations to generate complementary causal perspectives of the input, followed by a weight averaging strategy in Section 3.3.3 for robust and efficient inference. Figure 3.1 shows an overview of our method.

3.3.1 Preliminaries

TTA in Image Classification. Consider a classification model composed of a feature processor f_θ, parameterized by θ, and a classifier h_ϕ parameterized by ϕ. The model is pre-trained on a source dataset $\mathcal{D}_s = \{(\mathbf{X}_j, y_j)\}_{j=1}^{N_s}$ consisting of images $\mathbf{X}_j \in \mathbb{R}^{W\times H\times 3}$ and corresponding

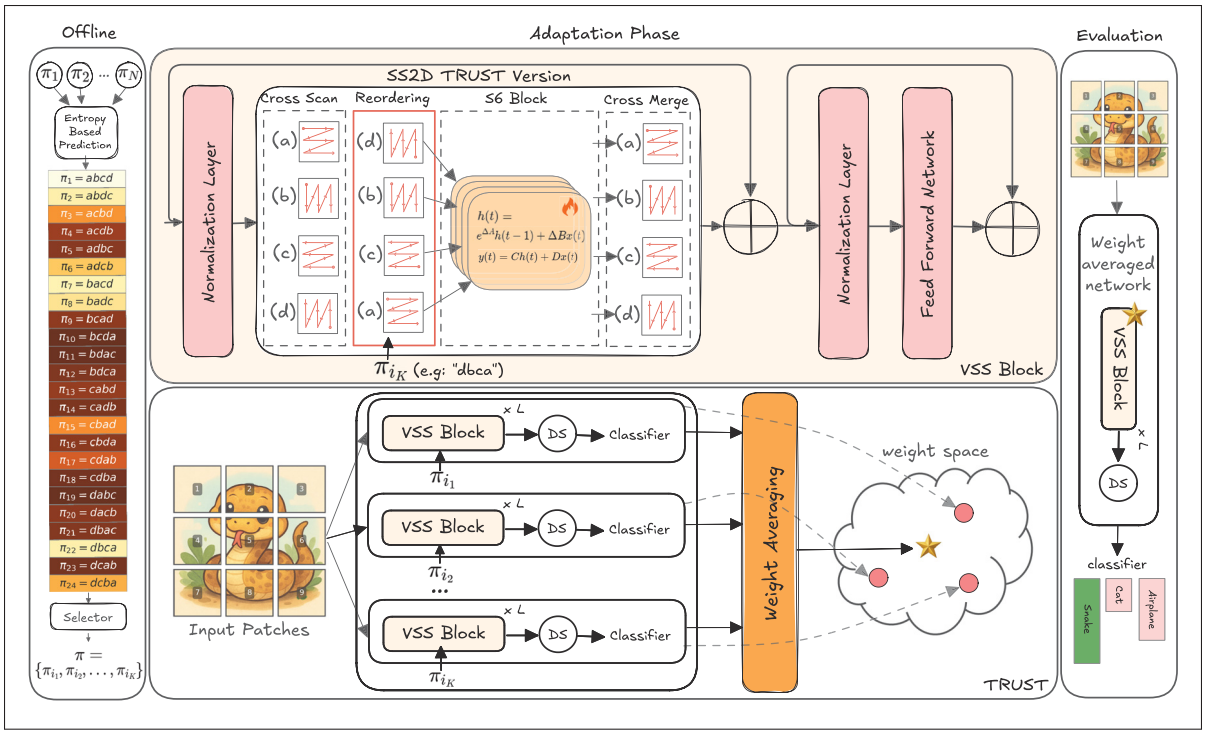


Figure 3.1 An overview of the proposed method. Our network consists of three stages: offline, adaptation, and evaluation. In the offline stage, multiple traversal permutations are generated and ranked by entropy. The top-K most confident permutations are selected. During adaptation, the selected permutations are applied sequentially, with each step updating the Mamba-specific state-space parameters from the previous step. The intermediate parameter states are stored and then merged via parameter-space averaging. The final averaged model is used for inference in the evaluation stage

labels $y_j \in \{1, \ldots, C\}$, where C is the number of classes. Formally, the model learns a map $F_s : \mathcal{X}_s \to \mathcal{Y}_s$ between the input distribution $\mathcal{X}_s$ and output distribution $\mathcal{Y}_s$ from the source. Training on $\mathcal{D}_s$ is performed using a supervised loss $\mathcal{L}_{\text{train}}$ (typically cross-entropy). At test-time, the model is exposed to a target dataset $\mathcal{D}_t = \{\mathbf{X}_j\}_{j=N_s+1}^{N_s+N_t}$, which lacks labels and follows a different distribution, giving rise to the map $F_t : \mathcal{X}_t \to \mathcal{Y}_t$. Although the source and target datasets share the same label space ($\mathcal{Y}_s = \mathcal{Y}_t$), their joint distribution differ, *i.e.*, $P(\mathcal{X}_s, \mathcal{Y}_s) \neq P(\mathcal{X}_t, \mathcal{Y}_t)$. Such a difference is referred to as *covariate domain shift* (Boudiaf *et al.*, 2022).

SSM Formulation. SSM-based models (Gu *et al.*, 2021a; Gu & Dao, 2023) map a 1D input sequence $\mathbf{x}(t) \in \mathbb{R}$ to an output $\mathbf{y}(t) \in \mathbb{R}$ through a learnable hidden state $\mathbf{h}(t) \in \mathbb{R}^N$, governed by a linear dynamical system:

$$\mathbf{h}'(t) = \mathbf{A}\mathbf{h}(t) + \mathbf{B}\mathbf{x}(t), \quad \mathbf{y}(t) = \mathbf{C}\mathbf{h}(t), \tag{3.1}$$

where $\mathbf{A} \in \mathbb{R}^{N\times N}$, $\mathbf{B} \in \mathbb{R}^{N\times 1}$, and $\mathbf{C} \in \mathbb{R}^{1\times N}$ are learnable parameters. To integrate this formulation into deep learning, it is discretized using the zero-order hold (ZOH) method with step size $\Delta \in \mathbb{R}$, resulting in the discrete recurrence:

$$\mathbf{h}(t) = \overline{\mathbf{A}}\mathbf{h}(t-1) + \overline{\mathbf{B}}\mathbf{x}(t), \quad \mathbf{y}(t) = \mathbf{C}\mathbf{h}(t), \quad \texttt{where} \quad \overline{\mathbf{A}} = \exp(\Delta\mathbf{A}), \quad \overline{\mathbf{B}} \approx \Delta\mathbf{B}. \tag{3.2}$$

Unrolling this recurrence over time yields a 1D convolution with kernel $\overline{\mathbf{K}} \in \mathbb{R}^L$, where $L \in \mathbb{N}$ denotes the kernel length. This form enables efficient parallel computation during training.

$$\mathbf{y} = \mathbf{x} \odot \overline{\mathbf{K}}, \quad \overline{\mathbf{K}} = \left(\mathbf{C}\overline{\mathbf{B}}, \mathbf{C}\overline{\mathbf{A}\mathbf{B}}, \ldots, \mathbf{C}\overline{\mathbf{A}}^{L-1}\overline{\mathbf{B}}\right). \tag{3.3}$$

SS2D for 2D Visual Processing. To extend 1D state-space models to visual data, VMamba introduces the SS2D module, which reshapes an input image $\mathbf{X} \in \mathbb{R}^{H\times W\times C}$ into a sequence of T non-overlapping patches $\{\mathbf{x}_t\}_{t=1}^T$. As shown in Fig. 3.1, these patches are traversed in a predefined causal order determined by one of four canonical directions using the Cross-Scan module: left-to-right (a), top-to-bottom (b), right-to-left (c), and bottom-to-top (d). For each direction, the patch sequence is processed recurrently by a shared, discretized 1D SSM, ensuring causality and directional consistency. The outputs from all four scans are then aggregated using the Cross-Merge module.

3.3.2 Traversal Permutation

To address the limitations of VMamba under distribution shifts, we propose a TTA strategy that leverages VMamba's intrinsic directional traversal mechanism. We define a set of traversal

permutations

$$\mathcal{P} = \{\pi_1, \pi_2 \ldots, \pi_N\}, \tag{3.4}$$

where each π_i represents a unique ordering of the four canonical scan directions, with $\pi_1 = \{a, b, c, d\}$ corresponding to the original ordering used in VMamba (see Fig. 3.1 for the definition of directions a, b, c and d). Permuting the scan directions changes their association with direction-specific SS2D parameter slices, producing different hidden states before the directional outputs are aligned and combined by Cross-Merge. We assess the predictive confidence of the model under each traversal permutation $\pi_i \in \mathcal{P}$ by computing the *Shannon entropy* of its output distribution. Permutations are then ranked in ascending order of entropy, and the top-K permutations with the lowest entropy are selected, where K denotes the number of selected traversal permutations (the offline mode of Figure 3.1):

$$\mathcal{P}_{\texttt{selected}} = \{\pi_{i_1}, \pi_{i_2}, \ldots, \pi_{i_K}\}. \tag{3.5}$$

During the adaptation phase, for each selected permutation π_{i_k}, $k = 1, \ldots, K$, the input is processed according to the corresponding traversal order. Denoting $p(\mathbf{X}; \pi_{i_k}) \in [0, 1]^C$ as the output of the model for an image $\mathbf{X}$ when using traversal permutation π_{i_k}, we compute a pseudo-label $\hat{y}_k$ by taking the class with maximum predicted probability:

$$\hat{y}_k = \underset{c \in \{1,\ldots,C\}}{\arg\max} \left[p(\mathbf{X}; \pi_{i_k})\right]_c. \tag{3.6}$$

The Mamba-specific parameters are then updated to minimize the average cross-entropy (*log loss*) between the model prediction and the corresponding pseudo-label over each target sample in batch $\mathcal{B}$:

$$\boldsymbol{\theta}_k = \underset{\boldsymbol{\theta}}{\arg\min} \; -\frac{1}{|\mathcal{B}|} \sum_{\mathbf{X} \in \mathcal{B}} \log \left[p(\mathbf{X}; \pi_{i_k})\right]_{\hat{y}_k}. \tag{3.7}$$

This is achieved using back-propagation as in standard methods. After optimization, the adapted parameter set $\boldsymbol{\theta}_k$ is cached, allowing a subsequent ensemble-based aggregation at inference.

Our method processes the same image through multiple directional permutations, enabling VMamba to exploit complementary causal views of the input. These distinct trajectories expose the model to both global consistency (identical token set) and local variation (different hidden-state evolutions), which helps find a flatter minima offering a stronger out-of-distribution (OOD) generalization (Cha *et al.*, 2021). Crucially, re-ordering prevents domain-specific artifacts from always entering the recurrence at the same time step. If an artifact $\boldsymbol{\varepsilon}$ is embedded in a corrupted patch $\mathbf{x}_{t_\varepsilon}$, then under the default traversal π_1, this patch always appears at time step $t = t_{\boldsymbol{\varepsilon}}$ in the processing sequence. The hidden state update at that step becomes:

$$\mathbf{h}^{(1)}(t_{\boldsymbol{\varepsilon}}) = f\left(\mathbf{h}^{(1)}(t_{\boldsymbol{\varepsilon}} - 1),\ \mathbf{x}_{t_\varepsilon} + \boldsymbol{\varepsilon}\right). \tag{3.8}$$

Since the hidden state $\mathbf{h}(t)$ is recurrent and accumulates over time, injecting $\boldsymbol{\varepsilon}$ early in the sequence allows its influence to propagate through many subsequent updates. Therefore, the impact of the artifact depends heavily on the position of $t_{\boldsymbol{\varepsilon}}$ within the sequence. In our method, we apply a traversal permutation π_{i_k}, which changes the order in which patches are processed. Under π_{i_k}, the corrupted patch $\mathbf{x}_{t_\varepsilon}$ appears at time step $t = \pi_{i_k}(t_{\boldsymbol{\varepsilon}})$, leading to the updated hidden state:

$$\mathbf{h}^{(i)}\left(\pi_{i_k}(t_{\boldsymbol{\varepsilon}})\right) = f\left(\mathbf{h}^{(i)}\left(\pi_{i_k}(t_{\boldsymbol{\varepsilon}}) - 1\right),\ \mathbf{x}_{t_\varepsilon} + \boldsymbol{\varepsilon}\right). \tag{3.9}$$

Across different permutations, the corrupted patch enters the sequence at varying time steps. This shifts the effect of artifact to different hidden states, breaking its consistent influence pattern. During test-time adaptation, these variations allow the model to learn diverse responses, and the subsequent weight averaging of adapted models helps suppress artifact-induced bias.

3.3.3 Leveraging Traversal Permutations Through Weight Averaging

To improve both stability and generalization under distribution shifts, we aggregate the adapted models obtained from the top-K traversal permutations. During the evaluation phase, we utilize

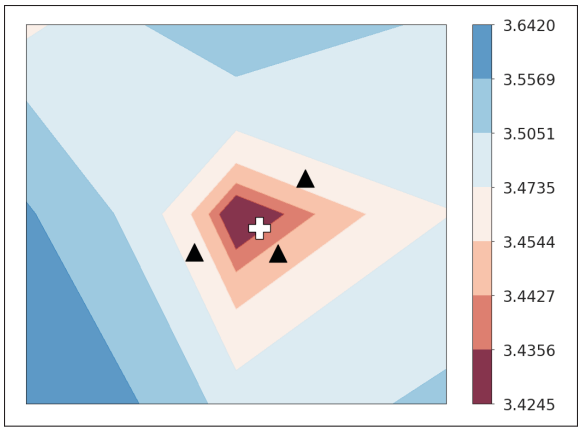


Figure 3.2 Loss surface of model parameters

the averaged weights

$$\bar{\theta} = \frac{1}{K} \sum_{k=1}^{K} \theta_k \tag{3.10}$$

and evaluate the model using the default traversal path π_1. In this setting, each θ_k represents an adapted model under a distinct traversal permutation. This results in a single, consolidated model that captures the benefits of multi-directional adaptation.

This simple yet effective averaging strategy draws inspiration from the notion of *flat minima* in the loss landscape (Izmailov *et al.*, 2018; Cha *et al.*, 2021), where parameter configurations that lie in flat regions tend to exhibit greater generalization and robustness to perturbations. Weight averaging encourages convergence to a region in parameter space with low curvature, thereby reducing sensitivity to input variations and improving performance under distribution shifts. In our approach, the integration of traversal permutations with weight averaging is inspired by Robust Risk Minimization (RRM) (Cha *et al.*, 2021). RRM aims to minimize the worst-case empirical loss within a neighborhood around the current parameters, formulated as:

$$\hat{\mathcal{E}}^{\gamma}_{\mathcal{D}}(\theta) = \max_{\|\Delta\| \leq \gamma} \hat{\mathcal{E}}_{\mathcal{D}}(\theta + \Delta), \tag{3.11}$$

where $\gamma > 0$ defines the neighborhood around the model parameters θ, and Δ shows small perturbations. To further illustrate this point, Figure 3.2 represents the test loss surface over

model parameters under Gaussian noise corruption from the ImageNet-C dataset. Each triangle marks a model adapted via a different traversal permutation, while the central cross denotes the weight-averaged model. The axes depict linear interpolations between parameter sets, offering a 2D view of the high-dimensional landscape. This visualization suggests that different traversal permutations lead to diverse parameter configurations and that their averaged model lies in a relatively low-loss region.

Unlike traditional TTA approaches that rely on data augmentation, external source data, or auxiliary objectives to generate diverse model variants for weight averaging, our approach leverages the internal architecture of VMamba, specifically, its canonical traversal mechanism. By permuting traversal scans, we induce structural variability that yields diverse perspectives of the same input, enabling effective adaptation without external augmentation.

3.4 Experiments

In this section, we present a comprehensive evaluation of our proposed method across seven benchmark datasets. We begin by describing the datasets used for evaluation, followed by implementation details and the baselines employed. We then report our main results and conclude with a set of ablation studies to analyze key components of our approach.

3.4.1 Datasets

We evaluate the performance and generalization of our method across a wide range of TTA benchmarks, covering corruption robustness and domain generalization. For corruption-based robustness, we use CIFAR10-C, CIFAR100-C, and ImageNet-C (Hendrycks & Dietterich, 2019), which apply 15 corruption types (e.g., noise, blur, weather, digital artifacts) at five severity levels. These datasets scale from 10 (CIFAR10-C) to 100 (CIFAR100-C) to 1000 (ImageNet-C) classes, enabling systematic analysis of model degradation.

For domain generalization, we assess on PACS-Photo (Li, Yang, Song & Hospedales, 2017), ImageNet-S (Wang, Ge, Lipton & Xing, 2019), ImageNet-V2 (Recht, Roelofs, Schmidt & Shankar,

2019), and ImageNet-R (Hendrycks *et al.*, 2021). PACS contains four visual domains (photo, art painting, cartoon, and sketch) with seven shared classes. In our PACS-Photo setting, the photo domain is held out for evaluation, while the remaining three domains are used for training. ImageNet-S features sketch-style abstractions; ImageNet-V2 provides a cleaner, independently collected test set; and ImageNet-R introduces style shifts across 200 classes with diverse artistic renditions. Together, these benchmarks test adaptability to unseen distributions, styles, and abstractions.

3.4.2 Implementation Details

We perform adaptation exclusively on the Mamba-specific state space parameters within the SS2D module of each VMamba block, while keeping the rest of the model frozen. The base VMamba model is trained with batch normalization layers. The adaptation is guided using a pseudo-labeling strategy, where confident model predictions are treated as supervision targets in a cross-entropy loss. This enables self-supervised adaptation without requiring access to ground-truth labels. The adaptation proceeds in an online manner, with model updates applied sequentially (i.e., without resetting the weights to their pre-adaptation values). Optimization is performed using the Adam optimizer with a learning rate of 10^{-4} and a batch size of 128, ensuring consistent dynamics and fair comparison across benchmarks. All experiments were conducted using a single NVIDIA *A*6000 GPU. Further details regarding the online test stream, reset policy, and optimizer-state handling are provided in the Supplementary Material.

3.4.3 Baselines

We compare our method against several state-of-the-art TTA approaches. Source Only refers to the performance of VMamba without any adaptation. ETA (Niu *et al.*, 2022) minimizes an entropy loss modulated by a sample-adaptive weighting mechanism, updating only the affine parameters of normalization layers. LAME (Boudiaf *et al.*, 2022) enforces smooth decision boundaries by introducing Laplacian regularization over the model predictions. SAR (Niu *et al.*, 2023) promotes robustness by explicitly optimizing for flat minima using sharpness-aware

Table 3.1 Top-1 classification accuracy (%) under various corruption types on CIFAR10-C, CIFAR100-C, and ImageNet-C datasets. Increases/decreases in mean accuracy compared to Source only are indicated by upward/downward arrows and additionally highlighted in green/red

	Method	gaussian noise	shot noise	impulse noise	defocus blur	glass blur	motion blur	zoom blur	frost	snow	fog	brightness	contrast	elastic	pixelate	JPEG compression	Mean
CIFAR10-C	Source only	46.8	48.4	45.0	73.5	52.6	73.0	78.7	71.8	75.8	77.3	85.7	69.6	63.7	67.9	59.0	65.9
	ETA (Niu *et al.*, 2022)	46.7	48.3	44.8	73.5	52.6	73.0	78.6	75.7	71.4	77.2	85.7	69.6	63.7	67.9	59.0	65.8 (↓0.1)
	LAME (Boudiaf *et al.*, 2022)	46.7	48.3	44.8	73.5	52.6	73.0	78.6	71.8	75.8	77.2	85.7	69.6	63.7	67.9	59.0	65.9
	SAR (Niu *et al.*, 2023)	47.7	49.5	46.2	74.3	53.4	73.8	79.1	72.5	76.5	78.0	86.1	70.8	64.5	68.9	60.0	66.8 (↑0.9)
	SHOT (Liang *et al.*, 2020)	47.8	49.7	46.3	74.3	53.7	74.0	79.3	72.6	76.6	78.1	86.3	70.7	64.5	68.9	59.9	66.8 (↑0.9)
	Tent (Wang *et al.*, 2021a)	47.3	49.2	45.8	74.2	53.1	73.7	79.1	72.2	76.3	77.9	86.1	70.4	64.3	68.7	59.6	66.5 (↑0.6)
	TRUST naive	58.9	61.8	62.0	79.8	60.9	79.1	82.6	80.5	81.8	83.6	88.8	81.8	70.3	75.1	66.0	74.2 (↑8.3)
	TRUST	**63.1**	**67.8**	**70.3**	**81.0**	**64.5**	**81.4**	**85.0**	**83.2**	**85.4**	**85.8**	**90.1**	**85.7**	**72.1**	**79.1**	**68.6**	**77.5** (↑**11.6**)
CIFAR100-C	Source only	21.0	22.1	18.3	50.6	27.7	51.0	56.2	45.3	50.6	52.4	65.3	43.2	39.0	41.7	33.4	41.2
	ETA (Niu *et al.*, 2022)	21.2	22.3	18.7	50.8	27.8	51.2	56.3	45.4	50.8	52.7	65.5	43.5	39.2	42.0	33.7	41.4 (↑0.2)
	LAME (Boudiaf *et al.*, 2022)	21.0	22.1	18.3	50.6	27.7	51.0	56.2	45.3	50.6	52.5	65.4	43.2	39.0	41.7	33.4	41.2
	SAR (Niu *et al.*, 2023)	21.9	22.8	19.3	51.1	28.2	51.5	56.7	46.4	51.4	53.1	65.8	44.2	39.9	42.8	34.2	41.9 (↑0.7)
	SHOT (Liang *et al.*, 2020)	21.9	22.9	19.1	51.4	28.3	51.8	56.8	46.3	51.4	53.3	66.0	44.2	39.8	42.7	34.3	42.0 (↑0.8)
	Tent (Wang *et al.*, 2021a)	21.6	22.6	18.9	51.1	28.2	51.5	56.7	46.2	51.1	53.1	65.8	44.0	39.7	42.5	34.0	41.8 (↑0.6)
	TRUST naive	32.1	32.8	34.1	56.9	35.4	57.2	61.6	54.6	57.8	60.1	69.6	55.6	46.6	50.8	41.0	49.8 (↑8.6)
	TRUST	**37.8**	**38.9**	**42.3**	**60.9**	**36.6**	**60.8**	**65.4**	**59.0**	**62.2**	**64.5**	**71.7**	**63.1**	**50.3**	**56.6**	**44.9**	**54.3** (↑**13.1**)
ImageNet-C	Source only	24.3	26.1	25.1	22.2	23.2	35.4	43.2	49.3	48.4	56.9	70.0	26.8	45.1	43.7	41.4	38.7
	ETA (Niu *et al.*, 2022)	26.4	28.4	27.2	23.5	24.6	37.2	45.1	50.8	51.0	58.8	70.6	29.1	47.7	46.9	45.0	40.8 (↑2.1)
	LAME (Boudiaf *et al.*, 2022)	24.3	26.1	25.1	22.2	23.2	35.4	43.2	49.3	48.4	56.9	70.0	26.8	45.1	43.7	41.4	38.8 (↑0.1)
	SAR (Niu *et al.*, 2023)	26.5	29.2	28.0	24.5	25.3	37.4	45.1	51.0	51.7	59.1	70.5	31.5	48.2	48.6	46.3	41.5 (↑2.8)
	SHOT (Liang *et al.*, 2020)	28.0	30.1	28.8	25.0	26.0	38.0	45.7	51.0	51.5	59.1	70.6	30.2	48.4	47.8	45.8	41.7 (↑3.0)
	Tent (Wang *et al.*, 2021a)	27.8	30.0	28.8	24.9	25.9	38.0	45.5	51.0	51.3	59.1	70.6	30.0	48.2	47.8	45.7	41.7 (↑3.0)
	TRUST naive	43.4	45.6	44.9	38.3	36.6	53.0	54.9	57.1	60.2	66.0	72.2	50.2	59.0	61.1	58.5	53.4 (↑14.7)
	TRUST	**46.8**	**49.4**	**48.5**	**42.8**	**40.8**	**57.1**	**57.9**	**57.3**	**61.7**	**66.8**	**71.9**	**54.9**	**61.4**	**63.6**	**60.2**	**56.1** (↑**17.4**)

minimization. SHOT (Liang *et al.*, 2020) aligns the classifier by minimizing entropy while encouraging confident and diverse predictions, using a fixed feature extractor. Tent (Wang *et al.*, 2021a) adapts the model by updating only the affine parameters of normalization layers via entropy minimization.

We evaluate two variants of our approach: TRUST naive, which adapts only the Mamba-specific SS2D parameters, and TRUST, which further enhances robustness by averaging over multiple traversal permutations during adaptation.

3.4.4 Main Results

Table 3.1 presents the Top-1 accuracy under the highest corruption severity (level 5) for CIFAR10-C, CIFAR100-C, and ImageNet-C datasets. TRUST consistently outperforms all baselines across the three datasets, demonstrating strong generalization under severe distribution shifts. It also surpasses its naive variant, underscoring the effectiveness of permutation-based strategy in enhancing robustness.

CIFAR10-C. TRUST achieves a mean accuracy of 77.5%, outperforming both Tent and SHOT by margins of 11.0% and 10.7%, respectively. On challenging corruptions, TRUST yields the largest improvements over SHOT, such as elastic (7.5%), glass blur (10.8%), and motion blur (7.4%). Compared to its naive variant, TRUST provides a further 3.3% boost. These results highlight the benefits of traversal diversity in enhancing corruption robustness.

CIFAR100-C. On a more fine-grained benchmark, TRUST attains a mean accuracy of 54.3%, exceeding SHOT and SAR by 12.3% and 12.4%, respectively. The largest absolute improvements over SHOT are observed under challenging corruptions such as impulse noise (23.2%), contrast (18.9%), and shot noise (16.1%), demonstrating the robustness of our method under severe distribution shifts. It also improves upon TRUST naive by 4.5%.

ImageNet-C. TRUST achieves a strong mean accuracy of 56.1% on the large-scale ImageNet-C benchmark, outperforming both Tent and SHOT by 14.4%, and TRUST naive by 2.7%. On challenging corruptions, TRUST yields the largest improvements over SHOT, such as glass blur (14.8%), elastic (19.3%), and JPEG compression (14.4%), reflecting the advantage of our method in mitigating both spatial distortions and digital artifacts.

In addition to corruption-specific robustness, our method demonstrates strong generalization to broader distribution shifts, as summarized in Table 3.2. On **ImageNet-S**, our method achieves 41.5% accuracy, outperforming SHOT by a substantial margin of 8.9%, and providing a modest 0.4% improvement over the naive variant. For **ImageNet-V2**, TRUST attains 64.0%, exceeding SHOT by 1.6% and providing a 0.6% gain over its naive variant. The most significant boost

Table 3.2 Top-1 classification accuracy (%) across datasets. For CIFAR10-C, CIFAR100-C, and ImageNet-C, values are averaged over all corruptions; for ImageNet-S, V2, R, and PACS-Photo, they reflect test set accuracy. Increases/decreases in mean accuracy compared to Source only are indicated by upward/downward arrows and additionally highlighted in green/red

Method	CIFAR10-C	CIFAR100-C	ImageNet-C	ImageNet-S	ImageNet-V2	ImageNet-R	PACS-Photo
Source only	65.9	41.2	38.7	31.4	62.2	31.3	66.7
ETA (Niu *et al.*, 2022)	65.8 (↓0.1)	41.4 (↑0.2)	40.8 (↑2.1)	31.4	62.2	31.3	66.7
LAME (Boudiaf *et al.*, 2022)	65.9	41.2	38.8 (↑0.1)	31.4	62.2	31.3	66.7
SAR (Niu *et al.*, 2023)	66.8 (↑0.9)	41.9 (↑0.7)	41.5 (↑2.8)	32.6 (↑1.2)	62.4 (↑0.2)	32.0 (↑0.7)	67.3 (↑0.6)
SHOT (Liang *et al.*, 2020)	66.8 (↑0.9)	42.0 (↑0.8)	41.7 (↑3.0)	32.6 (↑1.2)	62.4 (↑0.2)	31.9 (↑0.6)	67.4 (↑0.7)
Tent (Wang *et al.*, 2021a)	66.5 (↑0.6)	41.8 (↑0.6)	41.7 (↑3.0)	32.5 (↑1.1)	62.3 (↑0.1)	31.9 (↑0.6)	67.4 (↑0.7)
TRUST naive	74.2 (↑8.3)	49.8 (↑8.6)	53.4 (↑14.7)	41.1 (↑9.7)	63.4 (↑1.2)	39.7 (↑8.4)	67.1 (↑0.4)
TRUST	**77.5** (↑**11.6**)	**54.3** (↑**13.1**)	**56.1** (↑**17.4**)	**41.5** (↑**10.1**)	**64.0** (↑**1.8**)	**44.3** (↑**13.0**)	**69.9** (↑**3.2**)

is observed on the **ImageNet-R** benchmark, which features real-world renditions of ImageNet categories. Here, our method achieves 44.3%, surpassing both Tent and SHOT by 12.4%, and outperforming TRUST naive by 4.6%. On the **PACS-Photo** dataset, our model reaches 69.9% accuracy, offering consistent improvements over SHOT and Tent by 2.5%, and over the naive variant by 2.8%. These consistent gains across varied benchmarks highlight the adaptability of our permutation-based strategy, which not only improves corruption robustness but also scales effectively to domain generalization tasks. For completeness, we also report results under the standard reset protocol in the Appendix, where the model is reset to its source state after each test batch.

3.4.5 Ablation Study

In this section, we present a comprehensive ablation study on the CIFAR10-C dataset to assess the impact of key factors on the performance of our model. Specifically, we analyze the effects of batch size, augmentation types, number of traversal permutations, number of adaptation iterations, aggregation strategies, effect of traversing type in evaluation and computational overhead.

Batch Size. Figure 3.3 demonstrates that TRUST maintains a consistent advantage over Tent across different batch sizes. Even at smaller sizes such as 16 and 32, it delivers an accuracy

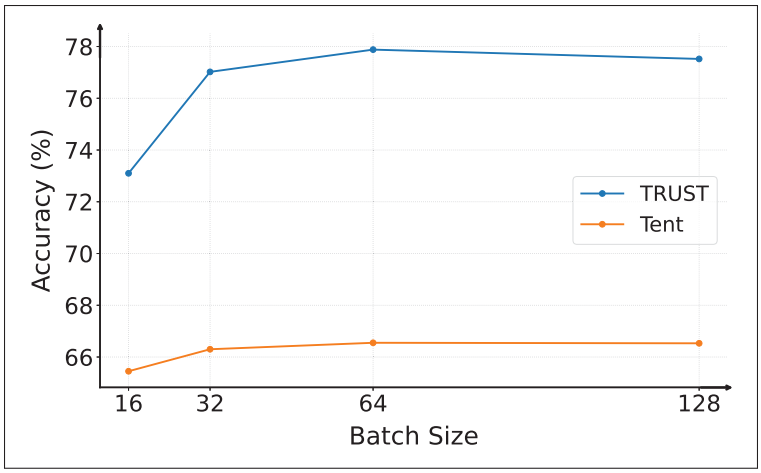


Figure 3.3 Accuracy comparison between TRUST and Tent across varying batch sizes on the CIFAR10-C dataset

improvement of around 8%. This indicates that our approach remains reliable and effective, even when operating with limited data per batch.

Augmentation Impact. To evaluate whether standard data augmentations can serve as effective alternatives to our traversal permutation strategy, we compare TRUST against common augmentations including rotation, random cropping, and color jitter. As shown in Figure 3.4, these traditional augmentations yield only modest improvements over the baseline source-only model on CIFAR10-C, with accuracies ranging from 65.9% to 68.3%. In contrast, our method achieves a substantial performance boost, reaching 77.5%, which is significantly higher than all augmentation baselines. This indicates that simple augmentations fail to sufficiently address distribution shifts.

Number of Traversal Permutations. Figure 3.5 shows how accuracy varies with the number of traversal permutations across CIFAR10-C, CIFAR100-C, and ImageNet-C. Overall, increasing the number of permutations consistently improves performance on all datasets. This is because by applying multiple traversal permutations, we expose the model to diverse causal perspectives of the same input. This variation encourages the model to learn different adaptation patterns at test time. Increasing the number of traversal permutations introduces more variation and, as

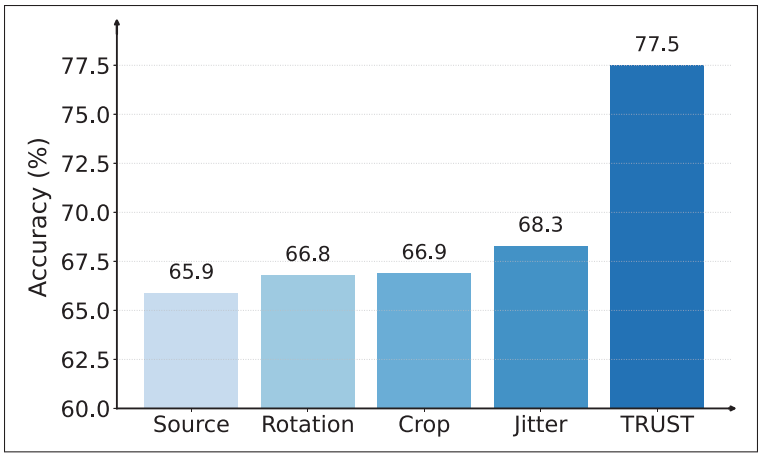


Figure 3.4 Performance comparison between standard augmentations and TRUST on the CIFAR10-C dataset

shown in works such as Model-Soups (Wortsman *et al.*, 2022), a greater number of diverse, correctly-aligned representations (our top-K permutations) can enhance the effectiveness of weight averaging.

More specifically on CIFAR10-C, accuracy rises from 75.6% (2 permutations) to 77.6% (8 permutations). For CIFAR100-C, the gain is from 51.7% to 54.7%, and for ImageNet-C, from 54.4% to a peak of 56.1% at six permutations before slightly dropping to 55.5% at eight. These results confirm that incorporating multiple traversal permutations enhances robustness under distribution shifts. Although accuracy improves with more permutations, gains diminish after six, suggesting that six permutations strike a good balance between performance and computational efficiency. For a breakdown of memory overhead, see Figure 3.9.

Number of Adaptation Iterations. Figure 3.6 illustrates that accuracy improves progressively with an increasing number of iterations. Despite the potential for further improvement with more iterations, we chose to use only a single iteration in subsequent experiments to enable faster adaptation without compromising on competitive performance.

Aggregation Strategy. We evaluate two baseline strategies for aggregation across traversal permutations: (1) an ensemble approach that averages model predictions from independently

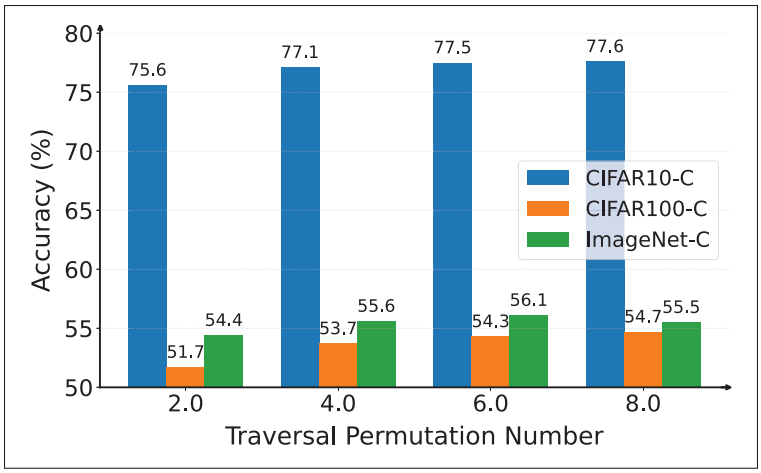


Figure 3.5 Effect of traversal permutation count on accuracy across three datasets

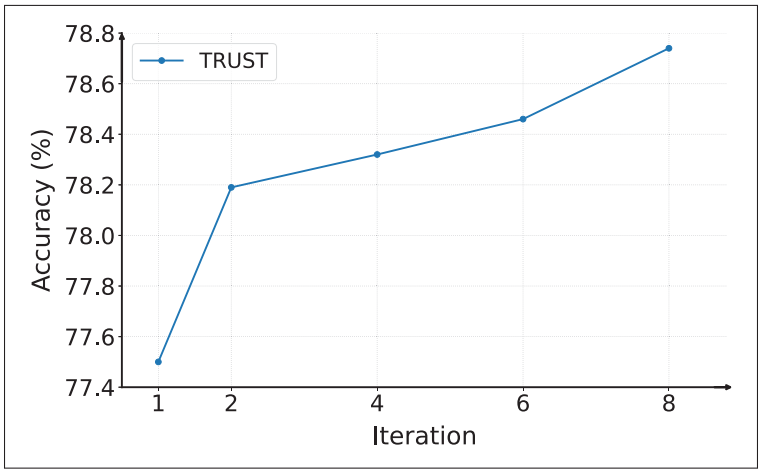


Figure 3.6 Model performance across adaptation iterations on the CIFAR10-C dataset

processed traversal orders, and (2) a repetition baseline, where the same traversal permutation is applied k times, followed by weight averaging. As shown in Figure 3.7, both baselines yield lower performance on CIFAR10-C, with ensemble achieving 68.1% and repetition 69.6%, compared to 75.6% with our method.

The ensemble variant offers traversal diversity, but because its individual models are never weight-averaged, it lacks parameter alignment. The repetition baseline, in contrast, repeats the same traversal and updates the model sequentially; the weights, therefore, remain aligned, yet no

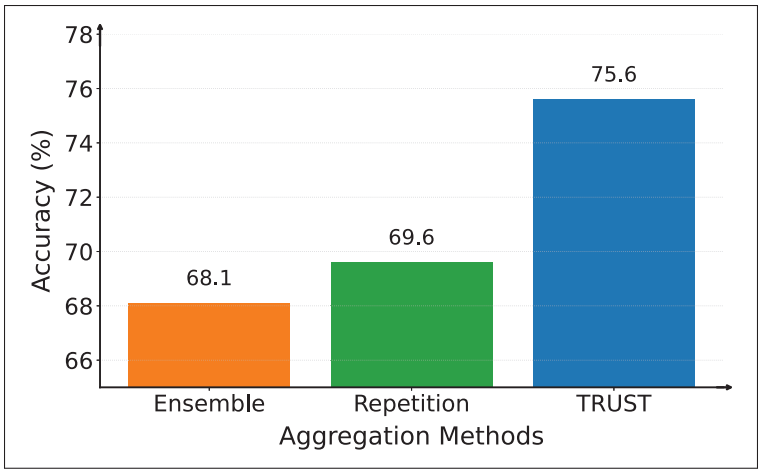


Figure 3.7 Accuracy comparison of different aggregation strategies on the CIFAR10-C dataset

traversal diversity is introduced. Both baselines perform worse than our method, demonstrating that neither unaligned traversal diversity (ensemble) nor aligned repetitions without diversity (repetition) is sufficient on its own. By combining these two ingredients, traversal diversity and parameter alignment, our approach achieves better generalization under distribution shift.

Effect of Traversal Permutation in Evaluation. Figure 3.8 illustrates how accuracy varies with different traversal permutations during evaluation. We observe that the choice of permutation impacts the performance. The default permutation used in the original VMamba architecture, "$abcd$", achieves the highest accuracy at 77.5%, while others, such as "$badc$", yield lower performance (e.g., 71.6%). Given this sensitivity, we adopt the "$abcd$" traversal permutation, on which the model was trained, for all evaluations to ensure consistency and leverage the strongest baseline.

Computational Overhead. We evaluate GPU memory usage as a function of the number of traversal permutations used during parallel adaptation. Since only the SS2D blocks are updated, we instantiate one SS2D block per traversal while sharing the rest of the network. Traversals are batched and routed to their corresponding SS2D blocks, and their outputs are then concatenated. This design minimizes computational overhead. As shown in Figure 3.9, memory

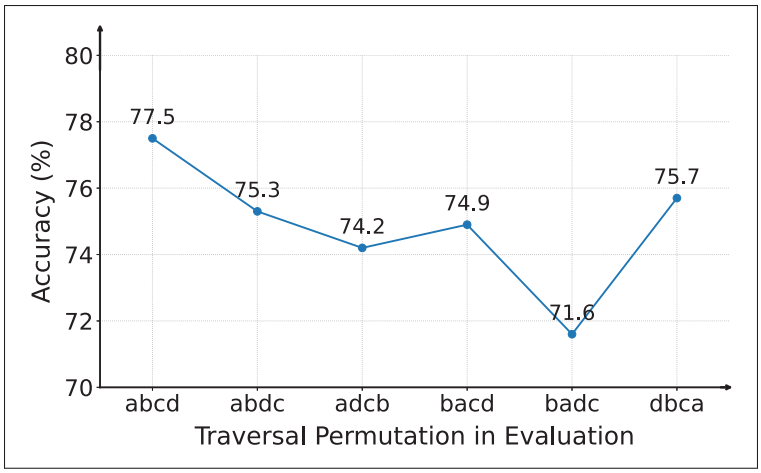


Figure 3.8 Impact of traversal permutation during evaluation on the CIFAR10-C dataset

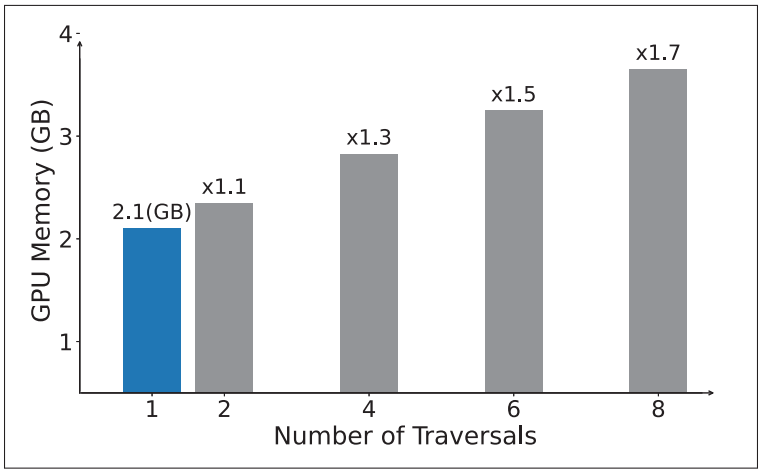


Figure 3.9 GPU memory usage across traversals

usage increases moderately, 1.1×, 1.3×, 1.5×, and 1.7× for 2, 4, 6, and 8 traversals, respectively, relative to the 2.1 GB baseline (measured with batch size 1). In sequential mode, forward and backward passes are performed separately for each traversal. We measure a per-traversal time of approximately 135 ms (with the batch size of 1), leading to a total cost of $K \times 135$ ms for K traversals. As expected, memory usage remains nearly constant, while adaptation time grows linearly. Conversely, in parallel mode, adaptation time remains nearly constant, with moderate memory overhead, offering a practical trade-off given the performance gains. For more details, refer to the Supplementary Material.

3.5 Conclusion

In this paper, we introduced TRUST, a test-time adaptation strategy specifically designed for vision-oriented state-space models. Our method addresses two key challenges in VMamba under distribution shift: (1) the strong inductive bias introduced by fixed traversal scans, and (2) the accumulation of domain-specific artifacts in hidden states during sequential processing. To overcome these issues, TRUST exploits the directional traversal mechanism of VMamba to generate complementary causal views of the input, performing adaptation based on each. The resulting models are aggregated via parameter averaging, promoting convergence toward flatter, more robust regions in the loss landscape. Experiments on seven standard benchmarks show that TRUST consistently outperform popular TTA models including Tent, SHOT, and SAR, in image classification tasks. Our method does not yet generalize to the medical domain.

CHAPTER 4

LISA: LINEAR STOCHASTIC ATTENTION FOR LONG-RANGE VISUAL MODELING

Sahar Dastani[a,c] , Fereshteh Shakeri[b] , Samuel Barbeau[a] , Kunal Mahatha[a] , Ismail Ben Ayed[b] , Herve Lombaert[c,d] , Christian Desrosiers[a]

[a] Department of Information Technologies Engineering, École de Technologie Supérieure
[b] Department of Systems Engineering, École de Technologie Supérieure
1100 Notre-Dame West, Montreal, Quebec, Canada H3C 1K3
[c] Mila – Quebec AI Institute
6666 Saint-Urbain Street, Suite 200, Montreal, QC H2S 3H1, Canada
[d] Polytechnique Montréal
2500 chemin de Polytechnique, Montreal, QC H3T 1J4, Canada

Paper submitted to *The IEEE/CVF Winter Conference on Applications of Computer Vision (WACV)*, June 2026

Candidate contribution. The candidate contributed to the problem formulation, theory, implementation, experiments, analysis, and manuscript preparation. Co-authors contributed through technical discussions, feedback, and revisions.

This chapter addresses Objective O3, which aims to enhance the expressiveness of efficient token interaction mechanisms. To achieve this objective, we introduce LiSA, a linear attention module that captures higher-order token interaction while preserving linear computational complexity.

Abstract

Self-attention has become the standard for modern sequence modeling, enabling Transformers to model long-range dependencies across a wide range of modalities. Despite these advances, its quadratic cost with respect to sequence length remains a significant computational bottleneck. Recent work addresses this limitation by introducing linear variants of self-attention, namely with kernel approximations of the softmax operator. However, their first-order aggregation restricts expressivity and struggles to capture higher-order structure. We propose Linear Stochastic Attention (LiSA), a linear attention plug-in module that enables higher-order token interactions

via a stochastic diffusion process based on truncated random walks. LiSA incorporates adaptive locality, favoring semantically similar neighbors to preserve fine-grained structures, while maintaining stable and efficient training. We conduct extensive experiments on classification, semantic segmentation, and object detection tasks and show improved results over state-of-the-art methods.

4.1 Introduction

Transformers (Vaswani *et al.*, 2017) have become a central paradigm in visual representation learning, with self-attention (Lin *et al.*, 2017b) serving as a key mechanism for capturing long-range dependencies. Since the advent of ViTs (Dosovitskiy *et al.*, 2021; Touvron *et al.*, 2021), self-attention has delivered strong performance across image classification (Touvron *et al.*, 2021; Wang *et al.*, 2021b), object detection (Li, Mao, Girshick & He, 2022c; Carion *et al.*, 2020), segmentation (Cheng, Schwing & Kirillov, 2021; Xie *et al.*, 2021a), and multi-modal tasks (Radford *et al.*, 2021), highlighting the value of global context in modern vision pipelines. However, self-attention scales quadratically with token length, which limits applicability to high-resolution images.

A number of recent works seek to reduce the quadratic complexity of standard ViTs. One common approach restricts attention to a limited set of tokens, either through fixed-sized attention windows (Liu *et al.*, 2021; Tang, Zhang, Zhu & Tan, 2022), introducing sparsity (Wang *et al.*, 2021b; Xia, Pan, Song, Li & Huang, 2022) or representing the attention queries and keys with low rank matrices (Wang, Li, Khabsa, Fang & Ma, 2020b). By reducing the number of pairwise token interactions, these methods lower computational cost, but often at the expense of global context modeling.

Other approaches linearize self-attention using kernel-based approximations (Katharopoulos *et al.*, 2020; Bolya *et al.*, 2022b; Shen *et al.*, 2021; Qin *et al.*, 2022c), replacing the softmax with a positive feature map $\phi(\cdot)$. This enables an associative reordering of the Q, K, and V

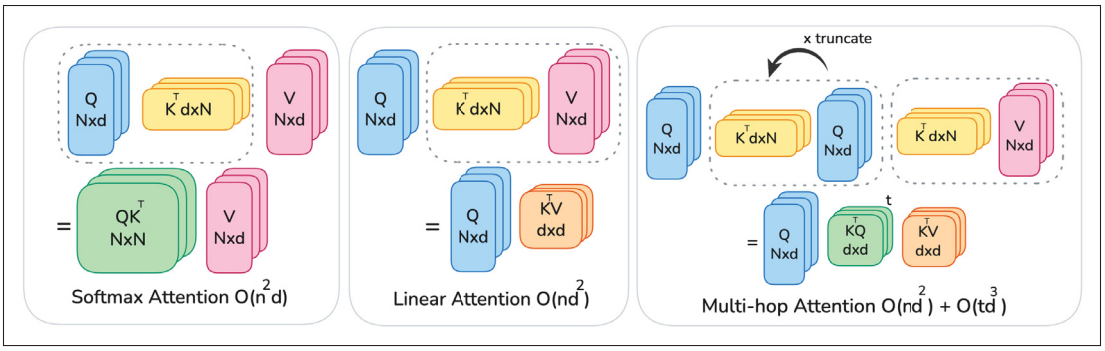


Figure 4.1 **Softmax vs. Linear vs. Higher-order attention.** Given $Q, K, V \in \mathbb{R}^{n \times d}$: softmax-attention forms $QK^\top$ then applies to V. Linear-attention computes $K^\top V$ then applies to Q. Higher-order attention (ours) uses a truncation length T, retaining multiple diffusion terms while remaining linear in n and expanding the effective receptive field

computation that avoids explicit pairwise interactions and reduces the complexity to linear in the sequence length. These methods retain a global receptive field without relying on downsampling.

In practice, however, an accuracy gap between kernelized linear-attention and standard self-attention persists. Recent works have reduced this gap by linearizing dot-product attention (Cai *et al.*, 2022), enriching the kernel with angular (You *et al.*, 2023) or focused parameterizations, and adding depthwise-convolutional components (Han *et al.*, 2023a) or auxiliary softmax branches (You *et al.*, 2023). However, these variants are almost always built on the same core backbone, vanilla kernelized linear-attention (Katharopoulos *et al.*, 2020). Our analysis points to a structural bottleneck in this baseline: its single-pass bilinear aggregation.

Vanilla linear-attention mixes values for each query once through direct query-key affinities. With the affinity matrix defined as $A = \phi(Q)\,\phi(K)^\top$, the output $Y = AV$ can be interpreted as a first-order message-passing step along the edges encoded by A. Crucially, indirect dependencies (i.e., using intermediate tokens) would appear only in higher powers (e.g., A^2V, A^3V), but vanilla linear-attention never materializes higher-order terms. As a result, when two regions are semantically related yet have weak first-order similarity (e.g., appearance changes due to lighting, occlusion, or viewpoint), their connection is under-weighted and the signal is diluted by many competing matches. In contrast, softmax-attention introduces competitive normalization that

can concentrate probability mass on high-scoring connections, which help preserve long-range links.

Our approach targets this limitation and proposes a novel plug-in *LiSA* module, which achieves efficient higher-order information propagation without sacrificing stability or locality. LiSA models attention as a linear stochastic process, where image patches are represented as graph nodes and edges encode transition probabilities among them during a truncated random walk. This formulation allows information to traverse intermediate tokens, enabling transitive reasoning. The random walk is truncated to a small number of steps to maintain linear complexity and numerical stability, providing a controllable balance between expressiveness and efficiency.

Overall, LiSA serves as a stronger drop-in module: it integrates cleanly with standard linear-attention refinements and often amplifies their benefits, delivering higher accuracy than vanilla linear-attention while preserving linear-time complexity and training stability. Our contributions are summarized as follows:

- We introduce LiSA, a plug-in linear-attention module that enables efficient higher-order token interactions.
- We formulate attention as a truncated random walk, enabling transitive information flow while preserving linear-time complexity and stable training.
- We evaluate our method across classification, segmentation, and detection, consistently outperforming both self-attention and linear-attention baselines.

4.2 Related Work

4.2.1 Efficient Vision Transformer

Although attention-based models have been highly successful in computer vision, directly applying self-attention to images is expensive as its computation and memory scale quadratically with the number of tokens. Existing approaches reduce this cost in two ways. First, methods such as ViT, VOLO, and T2T-ViT (Dosovitskiy *et al.*, 2021; Yuan, Hou, Jiang, Feng & Yan, 2022;

Yuan *et al.*, 2021) reduce the token count through patchification or progressive tokenization, with similar ideas extended to downstream tasks (Li *et al.*, 2022c). Second, hierarchical or sparse-attention models restrict token interactions: PVT/PVTv2 (Wang *et al.*, 2021b; Wang *et al.*, 2022b) reduce key/value resolution, Swin (Liu *et al.*, 2021) uses local windows, NAT (Hassani, Walton, Li, Li & Shi, 2023) adopts neighborhood attention, and BiFormer (Zhu, Wang, Ke, Zhang & Lau, 2023) uses bi-level routing. Despite these improvements, these methods rely on softmax-attention, whose quadratic scaling remains a bottleneck at high resolutions or limited compute budgets.

4.2.2 Linear Attention

An alternative to quadratic self-attention is linear attention (linear-attention) (Katharopoulos *et al.*, 2020), which factorizes the attention kernel over queries and keys and reorders the computation through $K^\top V$, reducing complexity from $O(n^2 d)$ to $O(nd^2)$ while retaining global context.

Several variants build on this formulation by improving the kernel approximation or attention weighting. Performer uses random features to approximate the softmax kernel (Choromanski *et al.*, 2021), CosFormer introduces cosine-based re-weighting (Qin *et al.*, 2022c), SOFT exploits low-rank approximations (Xiong *et al.*, 2021; Lu *et al.*, 2021a), and Efficient Attention applies separate normalization to queries and keys (Shen *et al.*, 2021). More recent methods address limitations such as weak selectivity, stability, and locality: TransNormer improves training stability (Qin *et al.*, 2022b), FLatten sharpens token selectivity through a focusing mechanism (Han *et al.*, 2023a), while Hydra and Castling-ViT explore alternative similarity functions (Bolya *et al.*, 2022b; You *et al.*, 2023). MLLA incorporates ideas from state-space models into linear-attention (Han *et al.*, 2024b).

Despite these advances, most linear-attention methods retain a single bilinear pass and model primarily first-order token interactions. We introduce LiSA, which enables higher-order,

transitive propagation through truncated random walks while preserving linear complexity and achieving competitive or better accuracy than softmax-attention.

4.2.3 Higher-Order Attention

Several works extend attention beyond direct interactions through diffusion or graph filtering. MAGNA (Wang, Ying, Huang & Leskovec, 2020a) diffuses attention over multiple hops in graph neural networks, while Diffuser (Feng, Li, Jiang & Ying, 2023) extends sparse Transformer attention with multi-hop diffusion. From a graph-signal-processing perspective, GFSA (Choi *et al.*, 2024) interprets self-attention as a first-order graph filter and generalizes it through polynomial filtering. More recently, AGF (Wi, Choi & Park, 2025) extends this perspective to linear Transformers by learning graph filters directly in the singular-value domain while retaining linear complexity. Attention Chains (Erel *et al.*, 2026) instead interprets existing softmax attention matrices as Markov chains, using repeated transitions to characterize indirect attention and stationary token importance.

Our approach is related through its use of higher-order interactions, but differs fundamentally in both formulation and role. MAGNA and Diffuser perform diffusion over explicit graph or sparse-attention structures, while GFSA and AGF formulate higher-order interactions as graph filtering. In contrast, LiSA constructs a stochastic transition operator directly from kernelized query–key affinities and uses its finite powers as the attention transformation itself. This also differs from Attention Chains, which applies repeated transitions to already-computed softmax attention matrices for analysis and token-importance estimation; in LiSA, multi-step propagation is trained end-to-end and directly produces the layer representations. Finally, rather than explicitly operating on the $n \times n$ transition matrix, LiSA exploits its low-rank query–key factorization to compute higher-order terms in the $d_h \times d_h$ space, preserving linear complexity in the number of tokens.

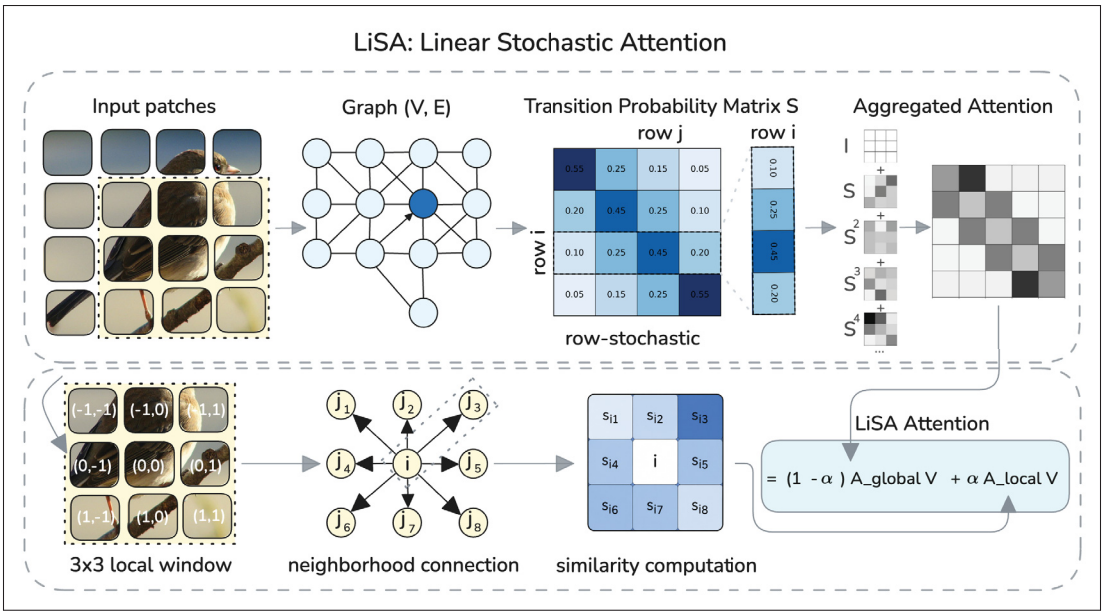


Figure 4.2 Overview of LiSA. We patchify the input image into tokens and treat them as nodes in a graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$, where edge weights define a row-stochastic transition matrix S. Aggregating the truncated terms on S yields a global attention map that captures transitive (higher-order) dependencies. In parallel, we compute content-based similarity within each token's local neighborhood to form a local attention map. The layer output is a convex combination of the two, blending sharp local context with long-range structure

4.3 Method

In Section 4.3.1 we give a brief definition of the self-attention mechanism and the linear version. We then proceed with the description and formulation of our proposed LiSA module in Section 4.3.2.

4.3.1 Preliminaries

Self-attention (SA). We start by presenting the self-attention model of transformers. Given a set of n tokens $X \in \mathbb{R}^{n \times d}$, each represented by a d-dimensional vector, self-attention maps X to a set of queries $Q = XW_Q \in \mathbb{R}^{n \times d_h}$, keys $K = XW_K \in \mathbb{R}^{n \times d_h}$ and values $V = XW_V \in \mathbb{R}^{n \times d}$, where W_Q, W_K and W_V are parameter matrices learned in training. The output of the self-attention

layer is obtained as:

$$Y \;=\; \mathrm{softmax}\Big(\frac{QK^\top}{\sqrt{d_h}}\Big) \tag{4.1}$$

The output can also be expressed for a specific token i as:

$$y_i \;=\; \sum_{j=1}^{N} a_{ij}\, v_j, \quad \text{where} \quad a_{ij} \;=\; \frac{\exp(q_i^\top k_j/\sqrt{d_h})}{\sum_{j'=1}^{N}\exp(q_i^\top k_{j'}/\sqrt{d_h})}, \tag{4.2}$$

and $y_i, v_j \in \mathbb{R}^d$. Self-attention requires computing the attention score a_{ij} between each pair of tokens before the softmax, leading to $O(n^2)$ time and memory complexity. This quadratic scaling makes high-resolution inputs and long sequences prohibitively expensive.

In this work, we seek an alternative to self-attention with the following properties: (1) **Scalability**: The complexity of the model in terms of memory and computation should increase *linearly* with respect to the number of tokens, during training as well as during inference. (2) **Expressiveness**: The model should also have expressiveness similar to that of self-attention while maintaining a comparable parameter budget. In particular, it should capture *global* and *non-linear* relations between input tokens.

Linear Attention (LA). linear-attention (Katharopoulos *et al.*, 2020) offers an effective alternative to softmax-attention by reducing the complexity from $O(n^2)$ to $O(n)$ in the sequence length n. The key idea is to replace the softmax similarity with a positive feature map $\phi(\cdot)$ so that the pairwise similarity can be written in a kernelized form $\phi(Q)\phi(K)^\top$. With this factorization, the output for token i becomes:

$$y_i = \frac{\sum_{j=1}^{n}\Big(\phi(q_i)\,\phi(k_j)^\top\Big)v_j}{\sum_{j=1}^{n}\Big(\phi(q_i)\,\phi(k_j)^\top\Big)} = \frac{\phi(q_i)\Big(\sum_{j=1}^{n}\phi(k_j)^\top v_j\Big)}{\phi(q_i)\Big(\sum_{j=1}^{n}\phi(k_j)^\top\Big)} \tag{4.3}$$

thereby avoiding the $n \times n$ attention matrix and yielding linear-time computation.

Despite their efficiency, current linear-attention approaches primarily perform a single-pass aggregation: $Y = \phi(Q)\big(\phi(K)^\top V\big)$ mixes each query with values exactly once via first-order (one-

hop) affinities. This design fails to materialize transitive dependencies involving intermediate tokens, which would arise from higher powers of $A = \phi(Q)\phi(K)^\top$.

4.3.2 Linear Stochastic Attention (LiSA)

Our LiSA approach is based on discrete random walks. This stochastic process models the probability of moving among the nodes $\mathcal{V}$ of a graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ with a time-invariant transition probability matrix $S \in \mathbb{R}^{|\mathcal{V}|\times|\mathcal{V}|}$ where each s_{ij}, for $(i, j) \in \mathcal{E}$ defines the probability of moving from node i to node j at any given time. Matrix S is row-stochastic as $\sum_j s_{ij} = 1, \forall i \in \mathcal{V}$.

At each step, the random walk can either continue by moving to another node with probability $0 \leq \beta < 1$, in which case the next node is selected based on the transition probability matrix S, or stop with probability $(1 - \beta)$. Denote by $p_{ij}^{(T)}$ the probability of ending the walk at node j when starting at node i under a truncation length T. Here, T denotes the truncation length, i.e., the number of terms retained in the diffusion expansion, corresponding to powers $S^0, \ldots, S^{T-1}$. The random walk probabilities can be expressed as a geometric series:

$$P^{(T)} = \left(\frac{1-\beta}{1-\beta^T}\right) \sum_{k=0}^{T-1} \beta^k S^k. \tag{4.4}$$

Proposition 1. *For walks of infinite length, probabilities can be computed analytically as:*

$$P^{(\infty)} = (1-\beta)\big(I - \beta S\big)^{-1}. \tag{4.5}$$

The proof is provided in the supplementary material.

To model global relationships between tokens, we follow the self-attention approach based on matrices representing the K, Q and V of tokens. However, we consider a more general formulation $Q = \phi(XW_Q) \in \mathbb{R}^{n\times d_h}$, $K = \phi(XW_K) \in \mathbb{R}^{n\times d_h}$, $V = \phi(XW_V) \in \mathbb{R}^{n\times d}$, where ϕ is a non-negative point-wise feature map.

Considering tokens as nodes in the graph, the transition probability from a token i to a token j is then computed as:

$$s_{ij} = \frac{q_i k_j^\top}{\sum_{j'} q_i k_{j'}^\top}. \tag{4.6}$$

This can be expressed in matrix form as: $S = (D^{-1}Q)K^\top = \tilde{Q}K^\top$ with $D = \mathrm{Diag}(Q(K^\top \mathbf{1}))$. Intuitively, the transition probability s_{ij} encodes information similar to the attention score a_{ij} of self-attention, where a higher value means a stronger relationship between tokens i and j.

To obtain the transformed features of a token i, we imagine a generative process where a random walk is initiated at node i and, once the walk stops at a node j, the value vector v_j is generated. For a potentially infinite random walk with stopping probability β, the expected features are then given by:

$$Y_{\text{global}} = A_{\text{global}} V = (1-\beta) \left(\overbrace{\left(I - \beta \tilde{Q} K^\top\right)^{-1}}^{\sum_{k=0}^{\infty} \beta^k (\tilde{Q}K^\top)^k} \right) V. \tag{4.7}$$

The formulation in Eq. (4.7) satisfies the desired property of expressiveness, as the function models an infinite-order relation between the input X and output Y with the same number of trained parameters as self-attention. Moreover, as described in the Supplemental Materials, it can be computed effectively, with a complexity linear in the number of tokens n, using the property that $S = \tilde{Q}K^\top$ is low rank. Nevertheless, matrix inversion, and in particular computing the derivative of this operation during back-propagation, remains expensive. In the next section, we present a more efficient solution based on truncated random walks.

Truncated Random Walk Approximation. To avoid matrix inversions, we retain the first T terms of the random-walk expansion. In this case, the computation of output features becomes:

$$\begin{aligned} Y_{\text{global}} &= \left(\frac{1-\beta}{1-\beta^T}\right)\left(\sum_{k=0}^{T-1}\beta^k S^k\right)V \\ &= \left(\frac{1-\beta}{1-\beta^T}\right)\left(I+\sum_{k=1}^{T-1}\beta^k \tilde{Q}(K^\top\tilde{Q})^{k-1}K^\top\right)V \\ &= \left(\frac{1-\beta}{1-\beta^T}\right)\left(V+\beta\tilde{Q}\underbrace{\left(\sum_{k=0}^{T-2}\beta^k(K^\top\tilde{Q})^k\right)}_{d_h\times d_h}K^\top V\right) \end{aligned} \tag{4.8}$$

This gives rise to the efficient computational process in Algorithm 4.1. Each loop of this algorithm multiples two $d_h \times d_h$ matrices, which has a complexity $O(d_h^3)$. Therefore, the total complexity of the algorithm is $O(T \cdot d_h^3) + O(n \cdot d \cdot d_h)$. Due to the added T factor, the truncation strategy has the important advantage of requiring only matrix multiplications that can be easily parallelized on GPU.

However, since this approach limits the random walk to a finite length, it can produce results that differ from those of the original infinite random walk model. Nevertheless, as shown in the following proposition, this discrepancy rapidly diminishes as the walk length increases.

Proposition 2. *Let R_T be the truncation residual obtained after retaining the first T terms in $P^{(\infty)}$. The L_1 norm of R_T decays exponentially as the truncation depth T increases.*

$$R_T = (1-\beta)\left(\sum_{k=T}^{\infty}\beta^k S^k\right). \tag{4.9}$$

The proof is provided in the supplementary material.

Learnable Step Weights. To further increase the expressiveness of the model, we allow the step weights in Eq. (4.8) to be learnable and input-dependent. Specifically, we replace the fixed geometric weights $\left(\frac{1-\beta}{1-\beta^T}\right)\beta^k$ with non-negative weights β_k predicted from the input and

normalized across the T retained terms. This yields:

$$Y_{\text{global}} = \beta_0 V + \sum_{k=0}^{T-2} \beta_{k+1}\, \tilde{Q}(K^\top \tilde{Q})^k K^\top V, \tag{4.10}$$

where $\beta_k \geq 0$ and $\sum_{k=0}^{T-1} \beta_k = 1$. Further details on the generation of the step weights are provided in the supplementary material.

Link to softmax in SA. Softmax attention adds nonlinearity through the *pointwise* exponential function: $\exp(x) = \sum_{k=0}^{\infty} \frac{x^k}{k!}$. In contrast, the random-walk (RW) diffusion probabilities are defined by a matrix power series over the *transition matrix* S: $P^{(\infty)} = (1-\beta) \sum_{k=0}^{\infty} \beta^k S^k$. Because each power S^k captures k-step transitions, RW diffusion models higher-order, non-pointwise relationships across the graph.

Adaptive Local Attention.

In LiSA, increasing the truncation length T retains higher-order powers of the row-stochastic transition operator, progressively extending information propagation across patches. Since each step updates $\mathbf{x}_i$ as a weighted average of other tokens, $\|\mathbf{x}_i - \mathbf{x}_j\|$ shrinks over steps, making distinct regions (e.g., foreground and background) harder to separate (over-smoothing). We add a local branch that restricts mixing to a 3×3 neighborhood to preserve spatial detail, while the global branch provides long-range context. This module models local interactions within a small neighborhood window around each token. For each patch token, affinities are computed only among its immediate spatial neighbors, with higher transition probabilities assigned to semantically similar regions. Let the spatial position of token i be $(h_i, w_i) \in \mathbb{Z}^2$ and define:

$$\begin{aligned} \mathcal{N}(i) = \big\{\, j : (h_j, w_j) = (h_i, w_i) + (\Delta_h, \Delta_w), \\ (\Delta_h, \Delta_w) \in \{-1, 0, 1\}^2 \,\big\}. \end{aligned} \tag{4.11}$$

Algorithm 4.1 Truncation-based computation

Require: Truncation length T
$M_1 \leftarrow I$
$M_2 \leftarrow I$
$M_3 \leftarrow \beta K^\top \tilde{Q}$
for $k = 1, \ldots, T-2$ **do**
1 $M_2 \leftarrow M_3 \cdot M_2$
$M_1 \leftarrow M_1 + M_2$
end for
$Y \leftarrow \beta \tilde{Q}\big(M_1(K^\top V)\big)$
$Y \leftarrow \left(\frac{1-\beta}{1-\beta^T}\right)(V + Y)$
return Y

We then form a sparse local affinity matrix $A^{\text{local}} \in \mathbb{R}^{n \times n}$ whose non-zero entries connect a node i only to its spatial neighbors:

$$A_{\text{local}} = M \odot \exp\left(\frac{QK^\top}{\sqrt{d_h}}\right) \tag{4.12}$$

where $M_{ij} = \mathbf{1}\{\, j \in \mathcal{N}(i) \,\}$. We row-normalize this affinity matrix to obtain the local attention transition. The neighborhood includes a self-transition through the $(0, 0)$ offset, which helps stabilize local aggregation. Without this self-loop, each token would be forced to replace itself with a weighted average of its neighboring tokens. All entries outside $\mathcal{N}(i)$ are zero and are never explicitly computed, so the local branch has linear complexity in the number of tokens.

Mixture of Operators. We combine the outputs of the global and local branches, allowing the model to trade off long-range higher-order aggregation with local neighborhood refinement using a learnable gate:

$$Y = (1 - \alpha) \underbrace{A_{\text{global}} V}_{Y_{\text{global}}} + \alpha \underbrace{A_{\text{local}} V}_{Y_{\text{local}}}, \tag{4.13}$$

where $\alpha \in [0, 1]$ is a learnable parameter that balances global and local information.

4.4 Experiments

We validate the effectiveness of our method on ImageNet-1K classification (Deng *et al.*, 2009), ADE20K semantic segmentation (Zhou *et al.*, 2019), and COCO object detection (Lin *et al.*, 2014), and include extensive ablations to assess each design component.

4.4.1 Image Classification

Datasets. We use the ImageNet-1K dataset (Deng *et al.*, 2009) for image classification. This dataset comprises 1.28M training images and 50K validation images across 1,000 categories.

Settings. We integrate our module into four Vision Transformer backbones, DeiT (Touvron *et al.*, 2021), PVT (Wang *et al.*, 2021b), Swin Transformer (Liu *et al.*, 2021), and BiFormer (Zhu *et al.*, 2023) and report Top-1 accuracy on the validation split. For a fair comparison, we adopt the same training recipe as the baseline. All models are trained from scratch for 300 epochs with AdamW (Loshchilov & Hutter, 2017), a cosine-decayed learning rate and a 20-epoch linear warmup. We initialize the learning rate at 1×10^{-3} and use a weight decay of 0.05. Data augmentation and regularization include RandAugment (Cubuk, Zoph, Shlens & Le, 2020), Mixup (Zhang, Cisse, Dauphin & Lopez-Paz, 2017), CutMix (Yun *et al.*, 2019), and Random Erasing (Zhong, Zheng, Kang, Li & Yang, 2020). The experiments were conducted on eight NVIDIA A100/H100 GPUs. We replace every self-attention layer with our LiSA module and use $T = 3$ for all main experiments. Since our method is a plug-in, it can be integrated into existing linear-attention architectures without other modifications. In these experiments, we use FLatten as the baseline linear attention, which combines a focusing function with a lightweight depthwise-convolution branch. In the ablation, we also evaluate additional baselines.

Results. Table 4.1 compares ImageNet Top-1 accuracy across DeiT, PVT, and Swin backbones. With comparable parameter counts and FLOPs, LiSA consistently outperforms the baseline and FLatten. On DeiT at 224 input resolution, LiSA improves DeiT-T/S by +2.34%/+0.98% and further surpasses FLatten-DeiT-T/S by +0.44%/+0.68%. On PVT, LiSA yields large gains

Table 4.1 Results of Classification. Comparison of LiSA across diverse vision backbones. Gains over the self-attention baseline are indicated by upward arrows and additionally highlighted in green

Model	Input	Params (M)	GFLOPs	Top-1 Acc. (%)
DeiT Family				
DeiT-T (Touvron *et al.*, 2021)	224	5.72	1.26	72.2
FLatten-DeiT-T (Han *et al.*, 2023a)	224	6.19	1.15	74.1
LiSA-DeiT-T	224	6.30	1.29	**74.54** (↑2.34)
DeiT-T (Touvron *et al.*, 2021)	384	5.79	4.70	74.9
FLatten-DeiT-T (Han *et al.*, 2023a)	384	7.14	3.37	76.53
LiSA-DeiT-T	384	7.24	3.75	**77.54** (↑2.64)
DeiT-S (Touvron *et al.*, 2021)	224	22.05	4.61	79.8
FLatten-DeiT-S (Han *et al.*, 2023a)	224	22.98	4.39	80.1
LiSA-DeiT-S	224	23.08	4.66	**80.78** (↑0.98)
PVT Family				
PVT-T (Wang *et al.*, 2021b)	224	13.23	1.94	75.1
FLatten-PVT-T (Han *et al.*, 2023a)	224	12.14	2.00	77.8
LiSA-PVT-T	224	12.16	2.12	**78.53** (↑3.43)
PVT-S (Wang *et al.*, 2021b)	224	24.48	3.82	79.8
FLatten-PVT-S (Han *et al.*, 2023a)	224	21.66	3.95	81.7
LiSA-PVT-S	224	21.69	4.18	**81.96** (↑2.16)
Swin Family				
Swin-T (Liu *et al.*, 2021)	224	28.29	4.49	81.3
FLatten-Swin-T (Han *et al.*, 2023a)	224	29.71	4.49	82.1
LiSA-Swin-T	224	29.74	4.72	**82.25** (↑0.95)

over the baselines (+3.43% on PVT-T and +2.16% on PVT-S) and improves over FLatten by +0.73%/+0.26%. On Swin, LiSA also brings +0.95% improvement compare to self-attention. At a higher resolution (384×384) on DeiT-T, LiSA reaches 77.54%, improving over FLatten by +1.01 points (76.53%), while using fewer FLOPs than the original DeiT-T at the same input size. Table 4.2 shows LiSA transfers to BiFormer as well, improving classification by +1.0%/+0.3%/+0.7% for T/S/B.

Table 4.2 Four-stage pyramid architecture: classification and segmentation results. Segmentation is with S-FPN. Dashes indicate values not reported. mIoU denotes mean Intersection over Union and mAcc denotes mean accuracy

Model	Par. (M)	Top-1 Acc. (%)	mIoU	mAcc
BiFormer-T (Zhu *et al.*, 2023)	13.1	81.4	—	—
LiSA-BiFormer-T	13.2	**82.4** (↑1.0)	—	—
BiFormer-S (Zhu *et al.*, 2023)	25.5	83.8	48.9	—
LiSA-BiFormer-S	25.6	**84.1** (↑0.3)	**49.5** (↑0.6)	**61.5**
BiFormer-B (Zhu *et al.*, 2023)	56.8	84.3	—	—
LiSA-BiFormer-B	56.8	**85.0** (↑0.7)	—	—

4.4.2 Semantic Segmentation

Datasets. For semantic segmentation we use ADE20K (Zhou *et al.*, 2019), the standard semantic-segmentation benchmark, which includes 20K training images, 2K validation images, and 150 categories.

Settings. For a fair comparison, we use the same baseline training setup. We plug our method into two representative segmentation frameworks, Semantic FPN (Kirillov *et al.*, 2019) and UPerNet (Xiao, Liu, Zhou, Jiang & Sun, 2018), implemented in MMSegmentation (MMSegmentation Contributors, 2020). Backbones are initialized from ImageNet-1K. We train Semantic FPN following the PVT recipe (Wang *et al.*, 2021b) and UPerNet using the default Swin configuration (Liu *et al.*, 2021).

Results. As shown in Table 4.3, our method delivers consistent gains across all settings. On PVT, it improves mIoU by more than 4.0 points and mAcc by more than 5.0 points over the baselines, under comparable FLOPs and parameter counts. On Swin, it improves mIoU and mAcc of the baseline by 0.5 and 0.8 points respectively. LiSA-BiFormer-S (Table 4.2) also improves the corresponding mIoU baseline by 0.6 points. These results show that local attention effectively captures informative features for segmentation.

Table 4.3 Results of semantic segmentation. FLOPs are computed with an input of 512×2048. S-FPN denotes Semantic FPN (Kirillov *et al.*, 2019). Dashes indicate values not reported

Semantic Segmentation on ADE20K					
Backbone	Method	GFLOPs	Par. (M)	mIoU	mAcc
PVT-T	S-FPN	158	17	36.57	46.72
FLatten-PVT-T	S-FPN	169	16	37.21	48.95
LiSA-PVT-T	S-FPN	170	16	**40.85** (↑4.28)	**51.75** (↑5.03)
Swin-T	UPerNet	945	60	44.51	55.61
FLatten-Swin-T	UPerNet	946	60	44.82	57.01
LiSA-Swin-T	UPerNet	947	60	**45.03** (↑0.52)	**56.4** (↑0.79)

4.4.3 Object Detection

Datasets. For object detection we use the COCO object detection and instance segmentation benchmark (Lin *et al.*, 2014), which includes 118K training images and 5K validation images.

Settings. We assess our method within Mask R-CNN (He, Gkioxari, Dollár & Girshick, 2017), initializing backbones with ImageNet pretraining. Experiments use 1× schedules with varying detection heads.

Results. As shown in Table 4.4, our method achieves the best accuracy with near-zero compute overhead. On PVT: box AP and mask AP improve by 3.8 and 2.5 points, respectively, compared with the PVT baseline. On Swin: box AP and mask AP have improve by 0.8 and 0.5 points, respectively, compared with the Swin baseline

4.4.4 Ablation Study

We conduct ablations to analyze the impact of individual components in our method. Unless stated otherwise, all ablation results are reported on the mini-ImageNet dataset (Vinyals *et al.*, 2016) ($T = 5$) for computational efficiency.

Table 4.4 Results of object detection. The FLOPs are computed over backbone, FPN and detection head with input resolution of 1280×800

Mask R-CNN Object Detection on COCO								
Method	GFLOPs	Sch.	AP^b	AP^b_{50}	AP^b_{75}	AP^m	AP^m_{50}	AP^m_{75}
PVT-T	240	1x	36.7	59.2	39.3	35.1	56.7	37.3
FLatten-PVT-T	244	1x	38.2	61.6	41.9	37.0	57.6	39.0
LiSA-PVT-T	245	1x	**40.5** (↑3.8)	**62.9** (↑3.8)	**44.1** (↑4.8)	**37.6** (↑2.5)	**59.7** (↑3.0)	**40.2** (↑2.9)
Swin-T	267	1x	43.7	66.6	47.7	39.8	63.3	42.7
FLatten-Swin-T	268	1x	44.2	67.3	48.5	40.2	63.8	43.0
LiSA-Swin-T	269	1x	**44.5** (↑0.8)	**67.4** (↑0.8)	**48.7** (↑1.0)	**40.3** (↑0.5)	**63.9** (↑0.6)	**43.1** (↑0.4)

Comparison with other linear-attention methods. For a fair comparison with prior linear-attention modules, we compare against six designs: Hydra Attention (Bolya *et al.*, 2022b), Vanilla linear-attention (Katharopoulos *et al.*, 2020), Efficient Attention (Shen *et al.*, 2021), Linear Angular Attention (You *et al.*, 2023), Enhanced linear-attention (Cai *et al.*, 2022), and Focused linear-attention (FLatten) (Han *et al.*, 2023a). Results in Table 4.5 show that LiSA outperforms all alternatives, including standard softmax-attention.

Table 4.5 Comparison of linear-attention variants on DeiT-S (ImageNet)

Linear Attention	Top-1 Acc. (%)
DeiT-S (Touvron *et al.*, 2021)	79.8
Hydra Attn (Bolya *et al.*, 2022b)	75.38
Vanilla Linear Attn (Katharopoulos *et al.*, 2020)	76.47
Efficient Attn (Shen *et al.*, 2021)	76.81
Linear Angular Attn (You *et al.*, 2023)	79.21
Enhanced Linear Attn (Cai *et al.*, 2022)	77.17
Focused Linear Attn (Han *et al.*, 2023a)	80.1
LiSA	**80.78** (↑0.98)

Ablation for α. Table 4.6 studies α, which balances LiSA's global and local branches. We compare global-only (α=0, F+D+G), local-only (α=1, F+D+L), and learned α. On both DeiT tiny and small, learning α performs best (76.3% and 80.7%), outperforming both individual

branches. Moreover, global-only exceeds local-only (73.7 vs. 73.4 on DeiT-T; 78.3 vs. 78.1 on DeiT-S), suggesting that long-range transitive aggregation provides a stronger standalone signal than local neighborhood matching.

Table 4.6 Component ablation on DeiT-T/DeiT-S classification (Top-1 %)

Backbone	F+D+G (α=0)	F+D+L (α=1)	Learned α
DeiT-T	73.7	73.4	**76.3**
DeiT-S	78.3	78.1	**80.7**

Number of truncations. We vary the truncation length T in Eq. 4.4. All LiSA components and settings are kept fixed; only the truncation length T is varied. Table 4.7 (run1) shows that accuracy peaks at a moderate truncation length ($T = 5$). Across three seeds, $T > 1$ consistently outperforms $T = 1$, isolating the benefit of higher-order propagation. Among the higher-order settings, $T = 3$ and $T = 5$ are comparable (75.61 ± 0.39 vs. 75.80 ± 0.44) and alternately achieve the best performance across runs. We therefore retain $T = 3$ for the main experiments as the more efficient truncation length, requiring fewer propagation terms.

Table 4.7 DeiT-T accuracy across three runs for different truncation length T. Bold values indicate the best T within each run

T	run1	run2	run3	Mean ± Std
1	73.52	73.88	73.03	73.48 ± 0.43
3	75.90	**75.76**	75.17	75.61 ± 0.39
5	**76.30**	75.58	**75.51**	75.80 ± 0.44
7	74.90	75.49	75.26	75.22 ± 0.30

Statistical variability. In Table 4.7, we added three-run results for all truncation depths T, reporting mean and standard deviation. The results confirm that higher-order propagation consistently improves over $T = 1$ (e.g., 75.61 ± 0.39 for $T = 3$ and 75.80 ± 0.44 for $T = 5$, versus 73.48 ± 0.43 for $T = 1$). The relatively small differences among the stronger higher-order

settings, together with their consistent gains over $T = 1$, indicate that the benefit is robust to run-to-run variability rather than driven by a single run.

Source of improvement. GL is our proposed attention core, a direct plug-in replacement for LA. F and D are additions borrowed from FLatten and are applied identically on top of both LA and GL. Our contribution is to replace the vanilla linear attention core (LA) with a higher-order global-local stochastic diffusion operator (GL). Table 4.8 isolates the contribution of our core: First, **GL** > **LA** (69.2 vs. 65.7), showing that our core already improves over vanilla linear attention before introducing F or D. Second, the same trend holds when each auxiliary component is added independently: **GL+F** > **LA+F** (69.3 vs. 65.9) and **GL+D** > **LA+D** (75.2 vs. 73.3). Notably, **GL+D** even surpasses the full FLatten baseline **LA+F+D** (75.2 vs. 74.4). Finally, when both F and D are applied equally to both cores, **GL+F+D** > **LA+F+D** (76.3 vs. 74.4), showing that LiSA outperforms FLatten under the same auxiliary design. These consistent gains indicate that the improvement comes from our proposed GL mechanism rather than from the borrowed auxiliary components.

Table 4.8 Component ablation on DeiT-T classification

LA	GL	LA+F	GL+F	LA+D	GL+D	**FLatten** LA+F+D	**LiSA** GL+F+D
65.7	69.2	65.9	69.3	73.3	75.2	74.4	**76.3**

Different linear-attention baseline. This experiment highlights the plug-in nature of our method and evaluates whether LiSA generalizes beyond FLatten. On DeiT-T, with linear angular attention (You *et al.*, 2023), the baseline reaches 66.0% Top-1, while adding LiSA improves it to 73.4%. Furthermore, integrating LiSA into Agent Attention (Han *et al.*, 2024c) improves DeiT-T from 73.40 to 75.77 (+2.37) and DeiT-S from 78.83 to 80.84 (+2.01), demonstrating that LiSA transfers effectively to other linear-attention variants.

a) DeiT-T attention map. The red square marks the query patch

b) Transitive reasoning visualized in DeiT-T attention maps

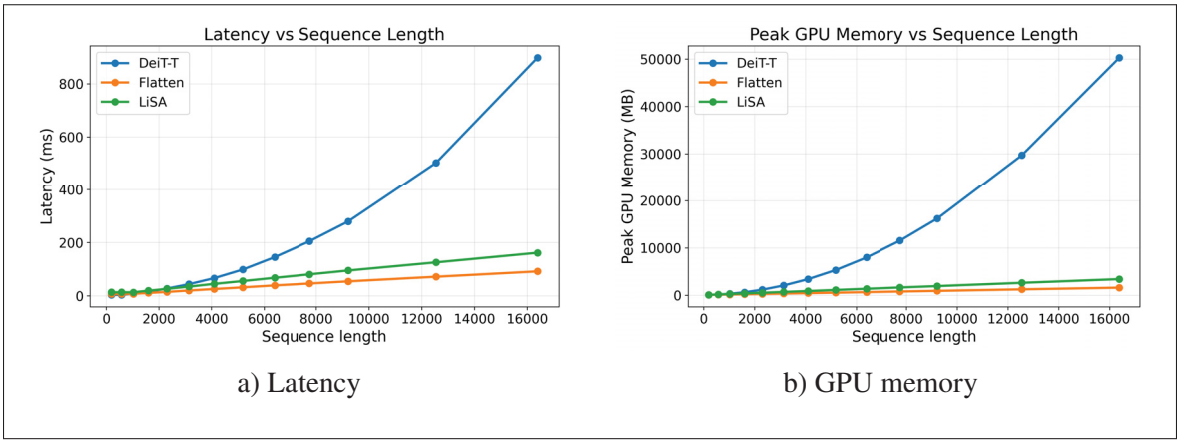
Figure 4.3 Visualization of attention behavior in DeiT-T

Distribution of linear-attention. As shown in Figure 4.3a, vanilla linear-attention produces diffuse maps: the weights are spread across many locations, so each query effectively aggregates an almost averaged representation of the whole image. In contrast, LiSA yields more structured, non-uniform patterns, where larger weights are assigned to patches on the same object as the query, while most background areas receive only small contributions.

Visualization of Transitive Reasoning. In Figure 4.3b, we start from a query patch on the bird (red box). Local attention remains restricted to nearby patches, so its influence stays

spatially confined. In contrast, LiSA expands its receptive field with hop depth: as we increase the truncation length (T=3, 5, 7), attention progressively propagates along semantically related regions, reaching distant yet correlated parts of the bird and eventually covering the tail and most of the object. Please zoom in for a clearer view.

Practical efficiency. We measure latency and peak GPU memory on a single NVIDIA H100 GPU across sequence lengths. The efficiency advantage of LiSA over softmax attention becomes more pronounced as sequence length increases. DeiT-T scales quadratically in both latency and memory, becoming prohibitively expensive at long sequences, while FLatten and LiSA maintain linear scaling. LiSA incurs a modest latency overhead compared to FLatten, which is expected given its higher-order random-walk modeling and adaptive local branch. This overhead reflects a trade-off between efficiency and expressiveness. Memory-wise, LiSA scales similarly to FLatten, remaining linear and far below softmax attention, indicating negligible additional memory overhead.



a) Latency b) GPU memory

Figure 4.4 Efficiency comparison across sequence lengths

4.5 Conclusion

We introduced LiSA, a linear-attention plug-in module that restores higher-order token interactions via a truncated random-walk formulation, while preserving stability and locality through an adaptive 3×3 neighborhood. This design directly overcomes the first-order

limitation of vanilla linear-attention, enabling richer transitive reasoning without sacrificing linear complexity. Across image classification, semantic segmentation, and object detection, LiSA yields consistent gains at near-identical compute budgets and transfers cleanly to diverse backbones (DeiT, PVT, Swin, BiFormer). Ablations confirm that each component contributes meaningfully and that a short truncation length already offers a strong accuracy–latency trade-off, positioning LiSA as a more powerful drop-in alternative to standard kernelized linear-attention. As future work, we plan to integrate LiSA into additional state-of-the-art linear-attention variants and study its behavior under diverse training recipes and data regimes.

CONCLUSION AND RECOMMENDATIONS

In this thesis, we investigated the hidden costs of efficient visual representation learning. Recent visual backbones have made substantial progress in reducing the computational cost of token interactions, enabling models to scale to larger images and dense prediction tasks. However, efficiency alone is not sufficient for reliable visual representation learning. Efficient models such as State Space Models (SSMs) and linear attention improve scalability, but they also introduce important trade-offs in robustness and expressiveness. The main objective of this thesis was therefore to improve the reliability of efficient visual representation learning by developing token interaction mechanisms that better balance computational scalability, robustness, and expressiveness. To achieve this main objective, we pursued three specific objectives.

The first objective was to reduce the sensitivity of SSM-based vision models to geometric transformations. Although SSMs provide efficient global modeling by processing image patches through sequential scan directions, this design imposes a fixed one-dimensional order on inherently two-dimensional visual data, making the learned representations sensitive to the chosen scan path and unstable under transformations such as rotation. To address this objective, Chapter 2 introduced Spectral VMamba, which uses the spectral decomposition of the graph Laplacian of image patches so that traversal follows the image's intrinsic structure rather than a fixed spatial direction. Combined with the Rotational Feature Normalizer, this approach improves rotation consistency while preserving the computational efficiency of SSM-based vision models. The first specific objective was therefore fulfilled.

The second objective was to strengthen the robustness of SSMs under deployment-time distribution shifts. In real-world deployment, test images often differ from the training data because of corruptions, domain changes, or variations in acquisition conditions. Such shifts can propagate through the SSM hidden state and corrupt subsequent patch representations. To address this objective, Chapter 3 introduced TRUST, a test-time adaptation method designed

specifically for Mamba-based vision models. TRUST generates diverse traversal permutations and uses uncertainty-guided selection to identify more stable views of the input. By adapting Mamba-specific parameters at inference time, TRUST improves robustness across multiple corruption benchmarks without requiring source data or retraining. The second specific objective was therefore fulfilled.

The third objective was to enhance the expressiveness of linear attention, whose standard formulations rely mainly on direct first-order interactions between tokens and therefore struggle to capture higher-order and transitive relationships between image regions. To address this objective, Chapter 4 introduced LiSA, a Linear Stochastic Attention module that reformulates attention as a truncated random walk over a graph of image patches. Through multi-hop information propagation, LiSA models more expressive token relationships while maintaining linear complexity and stable training. The third specific objective was therefore fulfilled.

Taken together, these three specific objectives advance the main objective of this thesis: developing efficient token-interaction mechanisms for visual representation learning that balance computational scalability with robustness and expressiveness, rather than optimizing computational cost alone. Our findings show that a model that is efficient but unstable under visual transformations, vulnerable to distribution shifts, or has limited expressivity may fail to provide reliable representations in practical settings. Future efficient vision models should therefore adopt a broader view of token-interaction quality.

Considering these contributions in the broader context of the field highlights three main limitations: the absence of a unified theory explaining the trade-off between efficiency, robustness, and expressiveness; the restriction of the robustness analysis to geometric transformations and corruption-based distribution shifts; and the focus of the proposed methods on specific visual recognition tasks and architectural settings, without examining whether similar token-interaction

principles generalize across tasks, domains, and modalities. These limitations motivate three broader research directions.

First, a unified theory of the efficiency–robustness–expressiveness trade-off in sub-quadratic token mixing. A key research direction is to develop a general theoretical framework that explains how efficient token-mixing mechanisms balance computational cost, robustness, and expressiveness. Such a framework could draw on information theory, spectral analysis, or both, and apply across SSMs, linear attention, and other sub-quadratic operators. It could provide architecture-independent design principles, establish theoretical performance limits, and clarify whether robustness and expressiveness can be improved jointly or whether they are fundamentally competing objectives.

Second, robustness to continual and adversarial distribution shifts. Future work could extend the robustness analysis to more challenging forms of non-stationarity, including gradual domain drift, continuously evolving data distributions, and adversarial perturbations. Extending methods such as TRUST to these settings would require models to adapt repeatedly without catastrophic forgetting. It would also be important to determine whether the architectural properties that enable sub-quadratic computation introduce specific vulnerabilities under adversarial conditions.

Third, general-purpose efficient token interaction across tasks and modalities. Future work could investigate whether a common efficient token-interaction mechanism can support different data structures and learning problems, including images, videos, graphs, and language, rather than relying on architectures designed for individual visual recognition tasks. This direction would study how token-interaction mechanisms can adapt to different spatial, temporal, and relational structures while maintaining computational efficiency.

APPENDIX I

SUPPLEMENTARY MATERIAL FOR THE PAPER TITLED SPECTRAL STATE SPACE MODEL FOR ROTATION-INVARIANT VISUAL REPRESENTATION LEARNING

1. Implementation Details

In this section, we give the pseudo-code for the RFN and STS modules. This pseudo-code provides a concise summary of the key steps involved in our approach, offering a high-level abstraction of the implementation. It is designed to complement the detailed explanations in the main paper and can serve as a reference for reproducing our results.

We begin with Algorithm I-1, which outlines the steps for implementing the RFN module. As discussed in the paper, this module promotes a consistent representation of image features under the four canonical quarter-turn rotations. This module comprises four key steps: rotation, patchification, back-rotation, and a max operation to aggregate features across all orientations.

Algorithm-A I-1 Rotational Feature Normalizer Module

Require: Input image $\mathcal{I} \in \mathbb{R}^{H\times W\times 3}$; rotation angles: $\{\theta_r\}_{r=1}^{4} = \{0°, 90°, 180°, 270°\}$;
stem module: Stem; patch size: p
Ensure: Aggregated feature map $\mathcal{F}$.

Initialize: $\{\theta_r\}$ and $\{-\theta_r\}$.
for $r \in \{1, \ldots, R\}$ **do**
  $\mathcal{I}_r \leftarrow \mathcal{R}_{\theta_r}(\mathcal{I})$ — *Rotate image by θ_r*
  $\mathcal{F}_r \leftarrow \text{Stem}(\mathcal{I}_r)$ — *Extract features*
  $\mathcal{F}_r^{\text{unrotated}} \leftarrow \mathcal{R}_{-\theta_r}(\mathcal{F}_r)$ — *Back-rotate feature map*
  Append $\mathcal{F}_r^{\text{unrotated}}$ to $\mathcal{F}_r$.
end for
$\mathcal{F} \leftarrow \max_{r\in\{1,\ldots,R\}} \mathcal{F}_r$ — *Max operation*
Output: Aggregated feature map $\mathcal{F}$.

Algorithm I-2 represents the STS strategy for determining the traversal order of image patches based on spectral decomposition. The process begins by constructing the adjacency matrix $\mathbf{W}$ using the k-Nearest Neighbors (KNN) algorithm, based on the Euclidean distances between image patches. This matrix captures the relationships and similarities between the patches.

Subsequently, the degree matrix $\mathbf{D}$ is computed, where each diagonal entry D_{ii} represents the sum of the weights of edges connected to node i. Using $\mathbf{W}$ and $\mathbf{D}$, the symmetric normalized Laplacian matrix $\mathbf{L}_{\text{sym}}$ is calculated. Spectral decomposition is then applied to $\mathbf{L}_{\text{sym}}$, yielding the eigenvalues $\boldsymbol{\lambda}$ and eigenvectors $\mathbf{U}$. Finally, the patches are reordered based on the spectral information, resulting in the ordered sequence of patches P.

Algorithm-A I-2 Spectral Traversal Scan Module

Require: patch feature $\mathbf{f}$, number of neighbors k, and eigenvectors m
Ensure: Traversal sequence $\mathcal{P}$

Step 1: *Compute Weighted Adjacency Matrix*
for each pair of patches $(\mathbf{x}_i, \mathbf{x}_j)$ **do**
  if $i \in knn_j$ OR $j \in knn_i$ **then**
    $W_{ij} = \exp\left(-\frac{\|\mathbf{f}_i - \mathbf{f}_j\|^2}{2\sigma^2}\right)$
  else
    $W_{ij} = 0$
  end if
end for
Step 2: *Compute Normalized Laplacian*
$\mathbf{L}_{\text{sym}} = \mathbf{I} - \mathbf{D}^{-\frac{1}{2}}\mathbf{W}\mathbf{D}^{-\frac{1}{2}}$
Step 3: *Compute m Smallest Eigenvectors of* $\mathbf{L}_{sym}$
$[\boldsymbol{\lambda}, \mathbf{U}] = \text{eigenSolver}(L_{\text{sym}}, m)$
Step 4: *Building Traversal Sequences*
Sort $\boldsymbol{\lambda}$ from smallest to largest.
Select the corresponding m eigenvectors from $\mathbf{U}$.
for $j = 1$ to m **do**
  P_1^j = sorting f using increasing order of $\mathbf{v}^{(j)}$
  P_2^j = sorting f using decreasing order of $\mathbf{v}^{(j)}$
  $\mathcal{P} = \left[P_1^{(j)},\ P_2^{(j)}\right]^{j=1,\ldots,m}$
end for
Output: Traversal sequence $\mathcal{P}$.

2. Downsampling Strategy

Similar to VMamba, the architecture of our network consists of four layers. Each layer contains a different number of Spectral VMamba blocks. For example, in the tiny scale configuration, we use the setup [2, 2, 5, 2], indicating that the first layer has two Spectral VMamba blocks, the second layer also has two blocks, the third layer has five blocks, and the final layer contains two

blocks. As previously mentioned, the STS module is applied only once, in the first Spectral VMamba block of the first layer. Additionally, downsampling occurs at the end of each layer.

The eigenvectors are initially computed in the STS using the original 14×14 spatial features. However, after downsampling, the eigenvectors derived from the 14×14 patches no longer align with the downsampled patches in subsequent layers, which requires careful handling to maintain consistency. To resolve this alignment issue, we save the indices used during the max pooling operation, which is responsible for downsampling the spatial features. By leveraging the saved indices, we extract the corresponding eigenvectors, generating a new set that matches the shape of the downsampled patches. This alignment ensures that the eigenvectors and patches remain correctly paired, enabling us to order the downsampled patches using eigenvectors of compatible dimensions. We repeat this process at each subsequent layer to maintain proper alignment throughout the network.

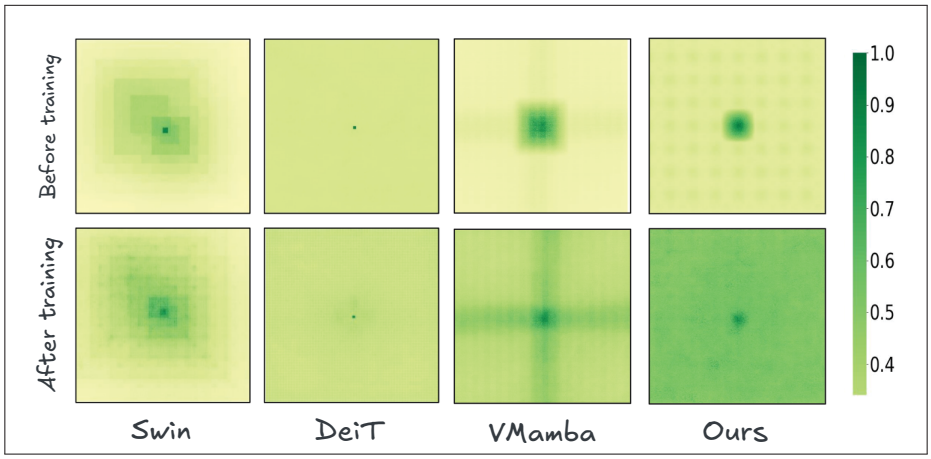


Figure-A I-1 Comparison of Effective Receptive Fields (ERF) Before and After Training

3. Visualization of Activation Maps

ERF in CNNs refer to the region of the input image that significantly influences the activation of a particular neuron in a deeper layer of the network. Unlike the theoretical receptive field, which considers the full extent of influence regardless of intensity, the ERF focuses on the practical impact, often showing that only a central portion of the theoretical receptive field has substantial

influence. Analyzing ERFs helps in refining network designs to ensure that neurons capture relevant information efficiently, enhancing performance in tasks such as object detection and image segmentation.

We conducted experiments to compare our model with VMamba, focusing on the ERF of the central pixel before and after training. Our results in Figure I-1 indicate that our model exhibits more global ERFs compared to VMamba. Specifically, after training, the areas colored dark green, which represent regions of high influence, are more extensive in our method than in VMamba. This suggests that our model is better at capturing broader contextual information from the input image. The increased dark green regions in our model demonstrate its enhanced capability to integrate information over larger portions of the input, potentially leading to improved performance in tasks requiring a comprehensive understanding of the image content.

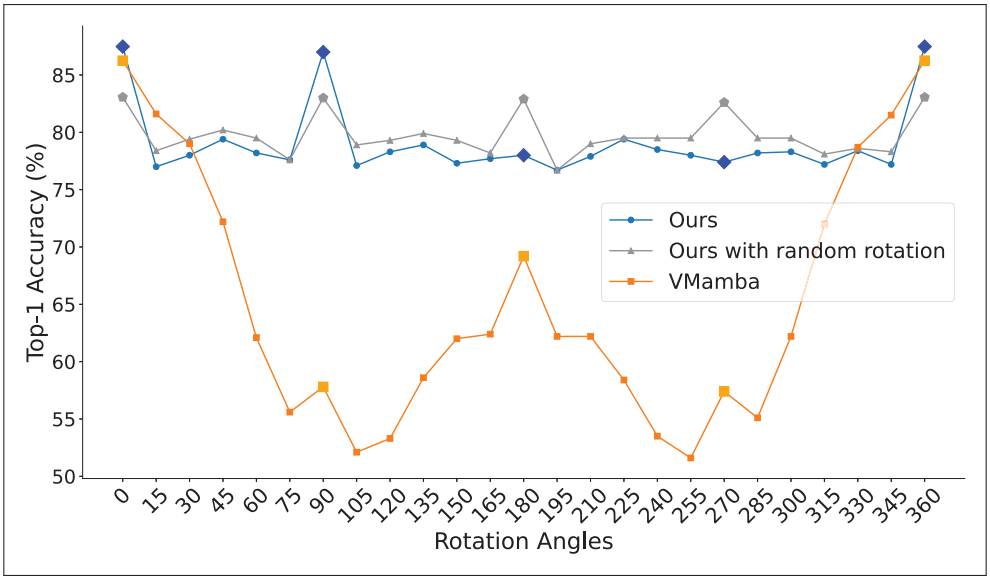


Figure-A I-2 Effect of Random Rotation in RFN Module

4. Training Dynamics

Figure I-3 illustrates the comparison of training dynamics, measured using the maximum accuracy with an Exponential Moving Average, across three model scales: Tiny, Small, and Base. Each plot represents the accuracy progression over 300 epochs for our proposed method

and the VMamba model. The plots represents our method demonstrates faster convergence across all scales, achieving high accuracy earlier in training compared to VMamba. The faster convergence not only highlights the efficiency of our approach but also reduces training time, making it highly suitable for practical applications where computational resources or time are constrained.

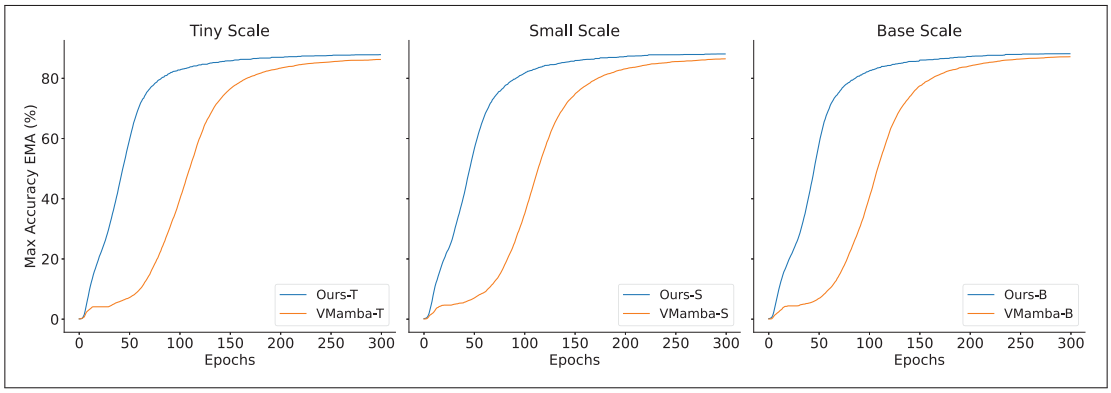


Figure-A I-3 Training Speed Comparison

5. RFN with Random Rotations

For the RFN module, we utilized four *canonical* angles: $0°, 90°, 180°, 270°$, corresponding to quarter turns. Additionally, we tested the method with four random rotations selected at each iteration from the ranges [0, 90], [91, 180], [181, 270], and [271, 360]. Unlike quarter turns, most random rotations necessitate interpolation. The results, presented in Figure I-2, demonstrate that even with random rotations, the model exhibits strong empirical robustness to arbitrary rotations. However, a slight accuracy drop was observed when using random rotations (83.06%) compared to the canonical quarter turns (87.48%), which can be due to the loss of details resulting from interpolation.

6. Downstream tasks

To demonstrate the versatility of our pre-trained model and its applicability to tasks beyond classification, we extended its use to semantic segmentation. Specifically, we fine-tuned the

model—originally pre-trained on the *mini*ImageNet dataset—to perform segmentation tasks on ADE20K dataset (Zhou *et al.*, 2019). ADE20K includes 150 fine-grained semantic categories and comprises 20,000 training images, 2,000 validation images, and 3,000 test images. For optimization, we use AdamW with a weight decay of 0.01 and a total batch size of 2 per GPU. The training schedule features an initial learning rate of 6×10^{-5}, linear decay, a 1500-iteration linear warmup, and a total of 160,000 iterations. We apply standard data augmentations such as random horizontal flipping, random scaling within a ratio range of 0.5 to 2.0, and random photometric distortion.

Table I-1 shows that our method outperforms VMamba by +2.81 (tiny), +1.68 (small), and +1.49 (base) on single-scale (SS) testing, and by +3.74 (tiny), +2.21 (small), and +0.58 (base) on multi-scale testing. The backbone used for the segmentation task was initialized from a classification checkpoint which was trained on the mini-ImageNet dataset. This explains the relatively lower range of segmentation performance compared to other papers, which often use classification models pre-trained on ImageNet-1k.

Table-A I-1 Results of semantic segmentation on ADE20K. mIoU denotes mean Intersection over Union; SS and MS denote single-scale and multi-scale testing, respectively

Method	**mIoU (SS)**	**mIoU (MS)**
VMamba-T	22.77	23.98
VMamba-S	25.84	27.13
VMamba-B	26.32	28.41
Ours-T	25.58 **(↑2.81)**	27.72 **(↑3.74)**
Ours-S	27.52 **(↑1.68)**	29.34 **(↑2.21)**
Ours-B	27.81 **(↑1.49)**	28.99 **(↑0.58)**

APPENDIX II

SUPPLEMENTARY MATERIAL FOR THE PAPER TITLED TRUST: TEST-TIME REFINEMENT USING UNCERTAINTY-GUIDED SSM TRAVERSES

1. Pseudo-code

In this section, we give the pseudo-code for our proposed test-time adaptation method, TRUST. This pseudo-code provides a concise summary of the key steps involved in our approach, offering a high-level abstraction of the implementation. Algorithm 1 outlines the overall TRUST procedure for test-time adaptation. For each corruption in the evaluation set, the model is first reset to its original (pre-adaptation) weights. The input x is then processed using the FORWARD_AND_ADAPT function (Algorithm 2), which outputs the adapted prediction and a list of model parameter sets θ, each corresponding to a different traversal permutation used during adaptation. These parameters are averaged to obtain the final adapted parameters, θ_{final}, which are loaded back into the model. The final prediction is then computed by re-evaluating the model on the same input x.

Algorithm-A II-1 TRUST

```
   for each corruption do
     model.reset()
     (out, θ) ← model.forward_and_adapt(x)
1    θ_final ← mean(θ)
     model.load_state_dict(θ_final)
     out ← model.evaluate(x)
   end for
```

Algorithm 2 defines the FORWARD_AND_ADAPT function. For each traversal permutation $\pi_{i_k} \in \mathcal{P}$, the model performs a forward pass with input x and permutation π_{i_k}. The cross-entropy loss between the model's prediction and the pseudo-labels is computed, and a gradient descent step is performed to update the model parameters. Each updated parameter state is stored in the list θ. After all permutations have been processed, the method returns the final output and the list of different model parameters.

Algorithm-A II-2 FORWARD_AND_ADAPT(x)

```
θ ← empty list
for each π_{i_k} ∈ P do
    out ← model(x, π_{i_k})
    loss ← CrossEntropy(out, pseudo_labels)
    loss.backward()
    optimizer.step()
    optimizer.zero_grad()
    θ.append(model.parameters)
end for
return (out, θ)
```

2. Finetuning Process

3. Efficient Parallel Implementation

We mentioned in the main paper (Computational Overhead section) that our method supports an efficient parallel adaptation strategy. Figure II-1 provides a detailed diagram of this process. In parallel mode, we handle K traversal permutations simultaneously. A batch $\mathcal{B}$ is first split into K subsets, each corresponding to a different permutation π_{i_k}. Each subset is then passed to an independent SS2D TRUST Version block, where the SS2D parameters are adapted in parallel while the rest of the model remains shared across all paths, this design significantly reduces memory usage.

After adaptation, the outputs are concatenated back into a single batch, which allows for efficient GPU utilization. For evaluation, we perform a weight averaging step across all adapted SS2D modules, producing a single unified SS2D TRUST Version block. This averaged block is then used for inference on the full batch. By leveraging parallelism in both data and traversal space, our method achieves scalable and low-overhead test-time adaptation while preserving performance across permutations.

4. Online adaptation protocol

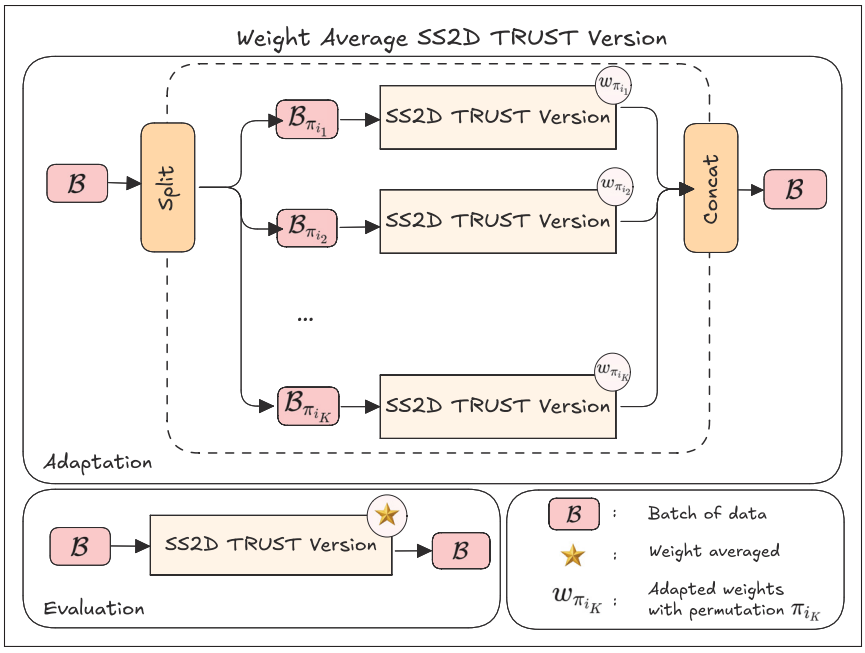


Figure-A II-1 Detailed diagram of TRUST in Parallel mode

All methods follow the same online test-time adaptation protocol. Corruptions are evaluated in a fixed, identical order across all methods, and within each corruption, test samples are drawn from a shared stream shuffled with a fixed seed, ensuring identical batch composition and order. Adapted parameters are carried forward across batches without resetting. At each corruption boundary, the model is restored to its source weights and the optimizer state is reinitialized, ensuring that corruptions are evaluated independently. All baselines follow the same test stream and reset policy while retaining their original adaptation procedures.

5. Standard Mode

Table II-1 reports Top-1 accuracy under various corruption types at severity level 5 for CIFAR10-C, CIFAR100-C, and ImageNet-C in the standard setting, where the model is reset for each test batch. On CIFAR10-C, TRUST achieves the highest mean accuracy of 66.2%, surpassing Tent and SHOT by 0.2%. On CIFAR100-C, it reaches 41.7%, outperforming all baselines by at least 0.4%, and improving over its naive variant by 0.5%, underscoring the benefits of permutation-based refinement. On the large-scale ImageNet-C, TRUST attains 39.9%, exceeding Tent by

1.1%, and outperforming its naive counterpart by 1.2%. These results highlight the scalability and robustness of permutation-aware adaptation across diverse datasets. Table II-1 evaluates performance under domain-level distribution shifts. TRUST achieves 31.6% on **ImageNet-S**, 62.3% on **ImageNet-V2**, and 31.5% on **ImageNet-R**. On the **PACS-Photo** benchmark, it obtains 71.3%, outperforming its naive variant by 4.6%. While gains are narrower in these less corrupted settings, TRUST consistently improves generalization, particularly under significant domain shifts.

Table-A II-1 Top-1 classification accuracy (%) across datasets in standard setting. For CIFAR10-C, CIFAR100-C, and ImageNet-C, values are averaged over all corruptions; for ImageNet-S, V2, R, and PACS-Photo, they reflect test set accuracy. Increases/decreases in mean accuracy compared to performing no adaptation (Source only) is highlighted in green/red color

Method	CIFAR10-C	CIFAR100-C	ImageNet-C	ImageNet-S	ImageNet-V2	ImageNet-R	PACS-Photo
Source only	65.9	41.2	38.7	31.4	62.2	31.3	66.7
ETA	65.8 (↓0.1)	41.2	38.8 (↑0.1)	31.4	62.2	31.4 (↑0.1)	66.7
LAME	65.9	41.2	38.8 (↑0.1)	31.4	62.2	31.4	66.7
SAR	66.0 (↑0.1)	41.3 (↑0.1)	38.8 (↑0.1)	31.4	62.2	31.4 (↑0.1)	67.1 (↑0.4)
SHOT	66.0 (↑0.1)	41.3 (↑0.1)	38.9 (↑0.2)	31.4	62.2	31.4 (↑0.1)	67.3 (↑0.6)
Tent	66.0 (↑0.1)	41.3 (↑0.1)	38.8 (↑0.1)	31.4	62.2	31.4 (↑0.1)	67.2 (↑0.5)
TRUST naive	65.9	41.2	38.9 (↑0.2)	31.5 (↑0.1)	62.3 (↑0.1)	31.5 (↑0.2)	66.8 (↑0.1)
TRUST	**66.2** (↑0.3)	**41.7** (↑0.5)	**39.9** (↑1.2)	**31.6** (↑0.2)	**62.3** (↑0.1)	**31.5** (↑0.2)	**71.3** (↑4.6)

6. TRUST with Batch Normalization Adaptation

In this experiment, we compare two adaptation strategies within our framework: updating only the BN layers versus updating the SS2D parameters. As shown in Table II-2, our method outperforms Tent in both settings, by 3.1% when using BN adaptation and by 11.8% with SS2D adaptation. We opt to proceed with SS2D adaptation, as traversal permutations directly influence Mamba-specific parameters encoded in the SS2D blocks. Updating these parameters is therefore essential to fully capture the effects of traversal-based modifications.

Table-A II-2 Comparison of adaptation strategies using BN and SS2D parameters on ImageNet-C dataset. Increases/decreases in mean accuracy compared to Tent are indicated by upward/downward arrows and additionally highlighted in green/red

	Method	gaussian	shot	impulse	defocus	glass	motion	zoom	frost	snow	fog	brightness	contrast	elastic	pixelate	JPEG	Mean
	Source only	24.3	26.1	25.1	22.2	23.2	35.4	43.2	49.3	48.4	56.9	70.0	26.8	45.1	43.7	41.4	38.7
BN	Tent	27.8	30.0	28.8	24.9	25.9	38.0	45.5	51.0	51.3	59.1	70.6	30.0	48.2	47.8	45.7	41.7
	TRUST	32.8	35.1	34.0	26.8	28.5	40.8	47.7	52.9	53.8	61.1	71.3	34.1	50.7	51.8	50.2	44.8 (↑3.1)
SS2D	Tent	29.6	31.7	34.7	25.1	22.0	45.7	44.8	46.0	55.8	62.2	69.8	32.5	52.8	56.8	54.3	44.3
	TRUST	46.8	49.4	48.5	42.8	40.8	57.1	57.9	57.3	61.7	66.8	71.9	54.9	61.4	63.6	60.2	56.1 (↑11.8)

7. Mean Entropy of Different Traversal Permutations

As illustrated in Figure II-2, varying the order of spatial traversals in VMamba leads to substantial differences in mean entropy across datasets, revealing the sensitivity of the model's internal representations to traversal patterns. Higher entropy values indicate less confident predictions, often aligning with reduced robustness under distributional shifts. This highlights the critical role of traversal selection in ensuring reliable adaptation. To offer a comprehensive analysis, we report entropy values for all permutations across CIFAR10-C, CIFAR100-C, ImageNet-C, ImageNet-S, ImageNet-V2, ImageNet-R, and PACS-Photo. Notably, the top-2, top-4, and top-6 traversal permutations with the lowest entropy, highlighted with green cross-hatching (i.e., diagonal lines overlaid on the heatmap cells), consistently reappear across datasets. This pattern suggests that certain traversal orders inherently yield more stable and confident model outputs, providing a principled foundation for selecting effective traversal subsets in our approach.

8. Combination of TRUST with Augmentation

According to Figure 4 in the main paper, Jitter augmentation yields the highest performance among the augmentation strategies evaluated. Building on this, we examine the effectiveness of combining Jitter with our proposed method, TRUST, under two augmentation settings: (1) applying different Jitter augmentations for each traversal permutation, and (2) applying different

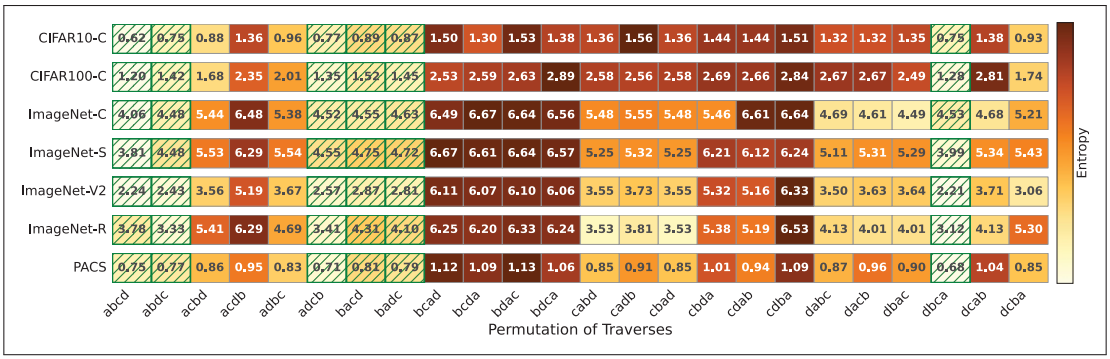


Figure-A II-2 Mean entropy of different traversal permutation across seven benchmarks

Jitter augmentations per batch. The former setting achieves a modest improvement of 0.3% over the source-only baseline, while the latter results in a slightly higher gain of 1.8%. However, both improvements are substantially lower than the performance boost achieved by the original TRUST, which yields an 11.6% increase. These results highlights that although augmentation contributes to performance, the core strength of TRUST lies in its ability to adapt representations based on traversal-specific dynamics rather than augmentation diversity alone.

Table-A II-3 Comparison of augmentation impact on CIFAR10-C dataset. Increases/decreases in mean accuracy compared to performing no adaptation (Source only) is highlighted in green/red color

Method	gaussian	shot	impulse	defocus	glass	motion	zoom	frost	snow	fog	brightness	contrast	elastic	pixelate	JPEG	Mean
Source only	46.8	48.4	45.0	73.5	52.6	73.0	78.7	71.8	75.8	77.3	85.7	69.6	63.7	67.9	59.0	65.9
TRUST + Jitter per permutation	53.6	54.2	54.0	63.8	48.0	69.3	77.1	75.4	75.2	77.2	85.8	76.6	59.1	66.5	57.8	66.2 (↑0.3)
TRUST + Jitter per batch	57.0	53.4	57.2	70.0	48.9	65.6	79.6	77.5	77.9	77.3	88.2	79.3	57.9	65.4	60.6	67.7 (↑1.8)
TRUST	63.1	67.8	70.3	81.0	64.5	81.4	85.0	83.2	85.4	85.8	90.1	85.7	72.1	79.1	68.6	77.5 (↑11.6)

9. High Entropy Traversal Permutations

We have also tested the performance of TRUST using the top-K high-entropy (i.e., less confident) traversal permutations. As shown in Table II-4, this setting leads to a significant performance drop of 12.5% compared to the source-only baseline. This highlights the critical role of traversal selection in our method. High-entropy permutations, which correspond to uncertain

and unstable model predictions, introduce noise into the adaptation process and hinder effective generalization. These findings further support our strategy of entropy-based traversal filtering, where low-entropy permutations are prioritized to ensure reliable and robust adaptation.

Table-A II-4 Comparison of TRUST performance with low and high entropy traversal permutations on CIFAR10-C dataset

Method	gaussian	shot	impulse	defocus	glass	motion	zoom	frost	snow	fog	brightness	contrast	elastic	pixelate	JPEG	Mean
Source only	46.8	48.4	45.0	73.5	52.6	73.0	78.7	71.8	75.8	77.3	85.7	69.6	63.7	67.9	59.0	65.9
TRUST (top-*K* high entropy)	38.5	32.2	36.4	57.9	37.7	61.5	64.9	52.5	64.0	70.3	74.1	49.3	57.9	56.2	48.2	53.4 (↓12.5)
TRUST (top-*K* low entropy)	63.1	67.8	70.3	81.0	64.5	81.4	85.0	83.2	85.4	85.8	90.1	85.7	72.1	79.1	68.6	77.5 (↑11.6)

10. TRUST Application beyond Classification Settings

We conducted additional experiments on segmentation tasks using various datasets, including Pascal VOC21 (V21) (Everingham, Van Gool, Williams, Winn & Zisserman, 2010) and Pascal Context 59 (P59) (Mottaghi *et al.*, 2014), applying corruptions following the protocol of (Noori *et al.*, 2025b). The results demonstrate that TRUST performs well in segmentation, outperforming methods like Tent. This further supports the generalizability and effectiveness of our approach beyond classification settings.

Table-A II-5 Segmentation performance under different corruptions of V21 and P59 datasets

Dataset	Method	gaussian noise	shot noise	impulse noise	defocus blur	glass blur	motion blur	zoom blur	frost	snow	fog	brightness	contrast	elastic	pixelate	JPEG compression	Mean
V21	Source only	29.1	33.1	28.3	21.0	8.2	33.1	25.4	50.9	50.3	70.7	76.5	63.9	25.5	22.2	59.2	39.8
	Tent	33.0	35.7	32.0	22.3	14.7	38.2	25.3	46.5	49.0	60.2	63.9	66.2	38.5	28.8	43.9	39.9
	TRUST	**38.8**	**42.0**	**38.7**	**29.8**	**22.6**	**45.1**	**29.8**	**50.5**	**53.5**	**63.4**	**66.4**	**68.5**	**45.1**	**37.7**	**48.6**	**45.4** (↑5.6)
P59	Source only	17.1	19.6	17.4	27.4	14.9	29.2	19.5	30.2	28.5	42.1	50.8	41.0	23.9	30.4	38.4	28.7
	Tent	17.6	18.9	17.8	22.2	15.9	27.5	17.9	26.9	30.0	36.7	41.9	42.9	25.8	28.2	28.3	26.6
	TRUST	**24.4**	**27.4**	**25.4**	**24.6**	**21.2**	**30.1**	**19.8**	**29.8**	**32.8**	**39.2**	**42.4**	**43.2**	**31.5**	**36.1**	**31.6**	**30.6** (↑1.9)

11. Performance Comparison with ViT-based Method

To provide a fair evaluation, we adapted the ViT-based method from (Niu *et al.*, 2024) by replacing its backbone with VMamba and modifying it to be compatible with VMamba backbone. Specifically, we replaced the classification (CLS) token (absent in VMamba) with the mean of all tokens and adjusted the number of learnable tokens to match VMamba's dimensionality. This ensures that both methods operate under the same architectural constraints. Our experimental results demonstrate that TRUST, which is specifically designed to exploit VMamba's traversal mechanism, significantly outperforms the adapted baseline, highlighting its robustness under distribution shift.

Table-A II-6 Performance comparison across corruption types with the ViT-based Method

Method	gaussian	shot	impulse	defocus	glass	motion	zoom	frost	snow	fog	brightness	contrast	elastic	pixelate	JPEG	Mean
Source only	24.3	26.1	25.1	22.2	23.2	35.4	43.2	49.3	48.4	56.9	70.0	26.8	45.1	43.7	41.4	38.7
Tent	27.8	30.0	28.8	24.9	25.9	38.0	45.5	51.0	51.3	59.1	70.6	30.0	48.2	47.8	45.7	41.7
FOA	17.8	19.4	18.5	15.1	18.2	22.7	28.6	38.7	33.9	44.3	57.4	22.7	38.2	40.9	41.7	30.5 (↓8.2)
TRUST	46.8	49.4	48.5	42.8	40.8	57.1	57.9	57.3	61.7	66.8	71.9	54.9	61.4	63.6	60.2	56.1 (↑17.4)

12. Weight Diversity Across Traversals

To analyze the consistency of the SS2D block parameters under different traversal permutations, we compute statistics based on the L2 norms of each parameter tensor across six traversal-adapted models. We extract each weight or bias tensor from each model, flatten it, and compute its L2 norm. We then calculate the mean of these norms to reflect the overall magnitude across traversals. To quantify variability, we also compute the standard deviation across the corresponding elements of these tensors and report the average of these deviations.

Figure II-3 shows the mean and standard deviation of the L2 norm for the bias parameters, while Figure II-4 shows the same for the weight parameters. These visualizations indicate that although the magnitude of parameters varies across layers, the variation across traversals

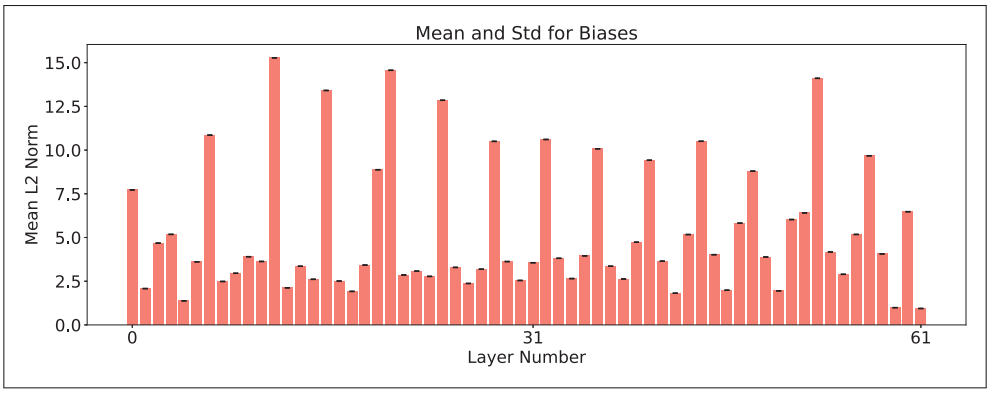


Figure-A II-3 Mean and standard deviation of the L2 norm for the bias parameters

remains relatively low. This suggests that SS2D block parameters adapted with different traversal orders remain geometrically close in parameter space. Such consistency is conceptually aligned with the findings in the Model Stock method (Jang, Yun & Han, 2024a), which emphasizes the benefits of maintaining proximity in the weight space across fine-tuned or adapted models, such as enabling effective weight averaging and improving generalization. This experiment is conducted using the CIFAR-10C dataset.

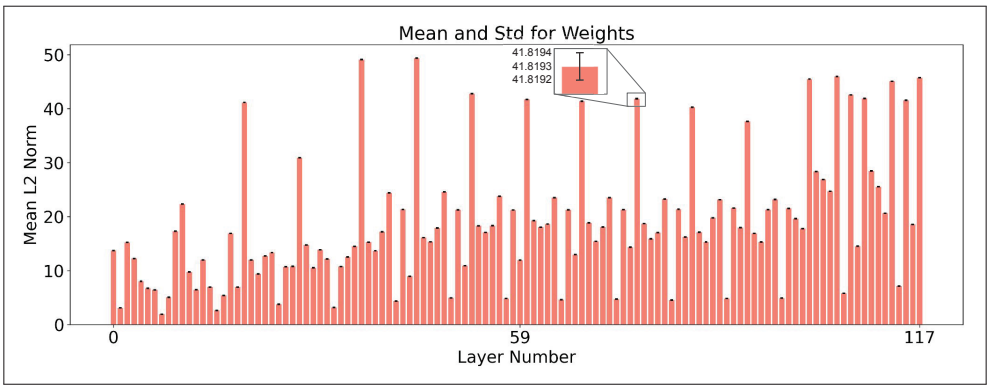


Figure-A II-4 Mean and standard deviation of the L2 norm for the weight parameters

13. Sensitivity to Traversal Order

SSMs are inherently direction-sensitive, meaning that the order in which input patches are traversed has a significant impact on model performance. According to the Mamba formulation Eq. 3.1, the hidden state at each step $h(t-1)$ directly influences the next $h(t)$. Therefore,

the choice of traversal order propagates its effects through the entire sequence of hidden-state updates. This directional dependency also comes from the fact that the hidden states learned during training are aligned with the original traversal order. When a different patch order is used at test time, the model is exposed to a sequence it has not encountered before, resulting in a mismatch between learned and test-time dynamics. As noted in the Mamba paper, this directional bias can cause the model to encode information more effectively along the original traversal path.

14. Offline Phase Scalability

The scalability of the offline selection phase is mainly determined by the number of traversal permutations. Each permutation requires only a forward pass through the model, with no backpropagation or parameter updates. As a result, the computational cost grows linearly with the number of candidate permutations. Memory usage, however, does not grow with the number of permutations, since the same model is reused for each forward pass. The required memory is therefore equivalent to running a single model evaluation. This makes the permutation search computationally predictable, while keeping the memory footprint fixed.

15. Theoretical Analysis of Confidence-Based Traversal Paths

We provide a deeper theoretical justification of why confidence-based paths improve generalization based on SWAD (Cha *et al.*, 2021). In our setting, each traversal permutation π_i defines a distinct causal ordering through which VMamba processes input patches, resulting in different trajectories of hidden states $h(t)$. Low-entropy (high-confidence) permutations consistently correspond to stable recurrent dynamics, where the influence of corruptions is minimized. This stabilizes the hidden-state evolution, yielding low-variance predictions and reducing sensitivity to domain-specific perturbations. From a loss landscape perspective, such paths correspond to regions of low curvature, as smoother predictions typically reflect flatter neighborhoods around θ_i (parameters of adapted model with π_i).

Therefore, selecting low-entropy permutations acts as a proxy for identifying solutions that are both dynamically stable (in terms of hidden-state recurrence) and geometrically robust (in terms of curvature). Averaging over the adapted weights from these confidence paths further concentrates the model in a flat, low-loss region of parameter space. This synergy between entropy-based selection and weight averaging aligns directly with the generalization theory of SWAD (Cha *et al.*, 2021), and provides a principled explanation for the observed robustness of TRUST across domains.

16. TRUST Latency

Our method consists of three stages: Offline phase: We evaluate our pre-trained model under different traversal permutations and select the top-K permutations with the lowest entropy values, corresponding to the most confident predictions. Since no model adaptation occurs at this stage, the procedure is equivalent to a forward-pass evaluation and is computationally lightweight. For instance, on the PACS dataset, evaluating for a single permutation takes only 4 seconds for all images in this dataset. Extending this across all 24 permutations results in a total evaluation time of 1.5 minutes, which is a one-time cost. Our experiments show that the top-K permutations remain consistent across all test datasets, as visualized in Figure II-2 of the supplementary material. This consistency stems from the similarity of these permutations to the original traversal order used during pre-training, as reflected in their high confidence scores. Therefore, this procedure can be performed only once on a single dataset to determine the appropriate value of K, and then used across all test datasets. Adaptation stage: We measured the latency for the batch size of 128 used in our experiments for forward and backward passes, which results in approximately 1.2 second per permutation. Evaluation stage: In this phase, we first perform weight averaging, which involves a simple averaging of the model weights across selected permutations (0.018 second). The subsequent evaluation functions similarly to the offline stage, no adaptation is performed, and we simply test the model (0.2 second). Our overall latency per permutation is: $1.2 + 0.018 + 0.2 = 1.41$ second.

APPENDIX III

SUPPLEMENTARY MATERIAL FOR THE PAPER TITLED LISA: LINEAR STOCHASTIC ATTENTION FOR LONG-RANGE VISUAL MODELING

1. Proofs of the Propositions

This section provides proofs for the propositions introduced in the methodology.

Proposition 3. *For walks of infinite length, probabilities can be computed analytically as*

$$P^{(\infty)} = (1-\beta)\big(I - \beta S\big)^{-1} \tag{A III-1}$$

Proof. Consider the sum over the first T terms in Eq. (4). Pre-multiplying this sum by $(I - \beta S)$ gives

$$\begin{aligned}(I - \beta S)P^{(T)} &= \left(\frac{1-\beta}{1-\beta^T}\right)\left(\sum_{k=0}^{T-1}\beta^k S^k - \sum_{k=1}^{T}\beta^k S^k\right) \\ &= \left(\frac{1-\beta}{1-\beta^T}\right)\left(I - (\beta S)^T\right).\end{aligned} \tag{A III-2}$$

Since $\beta < 1$, we have that $(\beta S)^T \to 0$ and $\beta^T \to 0$ when $T \to \infty$, hence

$$\begin{aligned}(I - \beta S)P^{(\infty)} &= (1-\beta) \\ P^{(\infty)} &= (1-\beta)\big(I - \beta S\big)^{-1}.\end{aligned} \tag{A III-3}$$

Proposition 4. *Let R_T be the truncation residual obtained after retaining the first T terms in $P^{(\infty)}$. The L_1 norm of R_T decays exponentially as the truncation depth T increases.*

$$R_T = (1-\beta)\left(\sum_{k=T}^{\infty}\beta^k S^k\right). \tag{A III-4}$$

Proof. Here, $\|\cdot\|_1$ denotes the entrywise matrix L_1 norm, $\|A\|_1 = \sum_{i,j} |A_{ij}|$. Since S is stochastic, we have that S^k is also stochastic, hence $\|S^k\|_1 = n$. Therefore,

$$\begin{aligned}
\|R_T\|_1 &= (1-\beta) \sum_{k=T}^{\infty} \beta^k \|S^k\|_1 \\
&= n(1-\beta) \sum_{k=T}^{\infty} \beta^k \\
&= n(1-\beta) \left(\sum_{k=0}^{\infty} \beta^k - \sum_{k=0}^{T-1} \beta^k \right) \\
&= n(1-\beta) \left(\frac{1}{1-\beta} - \frac{1-\beta^T}{1-\beta} \right) \\
&= n\beta^T. \qquad \text{(A III-5)}
\end{aligned}$$

For $\beta < 1$, $\|R_T\|_1$ therefore decays exponentially to zero as the truncation length T increases.

2. Additional Ablation Study

Unless stated otherwise, all ablation results are reported on the mini-ImageNet dataset **?** ($T = 5$) for computational efficiency.

Impact of kernel functions. In Table III-1, we ablate the feature-map non-linearity used in linear-attention. Rectified Linear Unit (ReLU) achieves the best performance (76.3% Top-1), slightly outperforming Exponential Linear Unit (ELU) (76.0%). We therefore use ReLU by default in all experiments.

Table-A III-1 Impact of different kernel functions (DeiT-T)

Kernel Function	ELU	ReLU
Accuracy (%)	76.0	**76.3**

Ablation for local window size. In Table III-2 we vary the window size in local attention on DeiT-T. A compact 3×3 window achieves the best Top-1 (76.3%) while being less computationally

expensive, whereas larger windows slightly reduce accuracy (75.7% at 5×5, 76.1% at 7×7) and increase cost.

Table-A III-2 Ablation study on local window size using DeiT-T

Local Window Size	3×3	5×5	7×7
Accuracy (%)	**76.3**	75.7	76.1

3. Learnable Step Weights Details

Here, the scalar β denotes the random-walk continuation probability in the geometric formulation, whereas $\beta_i^{(j)}$ denotes the input-conditioned vector of learnable step weights for token i and head j. The step weights in Eq. (10) are predicted per token and per head, allowing each token to adaptively weight the retained diffusion terms through a lightweight gate applied to its query and key:

$$\beta_i^{(j)} = \mathrm{softmax}\Big(W_2\,\mathrm{ReLU}\Big(W_1\left[q_i^{(j)}; k_i^{(j)}\right]\Big)\Big) \in \mathbb{R}^T, \tag{A III-6}$$

where $W_1 \in \mathbb{R}^{d_h \times 2d_h}$ and $W_2 \in \mathbb{R}^{T \times d_h}$. The softmax guarantees $\beta_k \geq 0$ and $\sum_{k=0}^{T-1} \beta_k = 1$, so the output is a convex combination of the truncation length T. The gate parameters are shared across heads and instantiated independently in each layer, while the weights are predicted separately for each token and head. The gate adds $O(n\,d\,d_h + n\,T\,d)$ operations, preserving linear complexity in the number of tokens and leaving the asymptotic complexity of Algorithm 1 unchanged.

4. Failure Cases

Excessive higher-order diffusion can be detrimental due to over-smoothing: since S is row-stochastic, deep walks converge toward a query-independent stationary distribution, making token representations increasingly similar and weakening discriminative local structure. This behavior is reflected in our truncation-depth ablation (Table 7, run1), where performance peaks at moderate T ($T = 5$ (76.3%) and decreases at larger depths 74.9% at $T = 7$).

5. Effective Regimes

LiSA provides the largest gains precisely where long-range modeling is hardest: plain columnar backbones with weak first-order cores (+3.43 on PVT-T; +7.4 when added to linear angular attention), high-resolution inputs (+2.64 at 384^2), and dense prediction tasks that require spatial reasoning (+4.28 mIoU on ADE20K, +3.8 box AP on COCO). Gains remain consistent but naturally smaller for backbones that already encode hierarchy and locality (e.g. +0.95 on Swin-T), where part of the long-range signal is available by design. Its additional cost is therefore least justified at short sequence lengths, where the $O(T\, d_h^3)$ term is not amortized.

6. Detailed Efficiency Measurements

To complement the efficiency plots in the main paper, we additionally report the exact latency, throughput, and peak GPU memory values in Table III-3. The measurements span input resolutions from 224^2 to 2048^2, corresponding to sequence lengths from 197 to 16, 385 tokens, and compare DeiT-T, FLatten, and LiSA under the same setting. Consistent with Fig. 4, DeiT-T exhibits rapidly increasing latency and memory as the sequence length grows, whereas FLatten and LiSA scale much more favorably. LiSA incurs additional latency over FLatten due to its higher-order propagation and adaptive local branch, while retaining substantially lower latency and memory than softmax attention at long sequences.

7. Model Architecture

We summarize the architectures of the three Transformer models used in the main paper, DeiT, PVT, and Swin Transformer, in Table III-4, III-5, and III-6. In all cases, we replace each original self-attention block with our proposed LiSA block.

Table-A III-3 Latency, throughput, and peak GPU memory across input resolutions

Input		DeiT-T			Flatten			LiSA		
Res.	**Seq.**	**Latency (ms)**	**Throughput (img/s)**	**GPU Memory (MB)**	**Latency (ms)**	**Throughput (img/s)**	**GPU Memory (MB)**	**Latency (ms)**	**Throughput (img/s)**	**GPU Memory (MB)**
224	197	3.50	2288.00	74.0	7.15	1118.91	73.2	13.53	591.49	95.4
384	577	4.07	1967.60	152.7	7.34	1090.32	110.2	13.52	591.60	172.1
512	1025	8.06	993.06	312.9	7.40	1080.62	151.1	13.48	593.68	263.4
640	1601	15.56	514.19	627.9	11.11	720.42	206.6	19.47	410.90	382.0
768	2305	27.69	288.93	1178.7	14.74	542.73	275.7	26.15	305.90	529.2
896	3137	43.80	182.66	2061.4	19.96	400.87	352.4	34.86	229.49	697.4
1024	4097	66.51	120.29	3394.2	25.59	312.58	441.9	44.89	178.21	890.9
1152	5185	98.78	80.99	5317.0	31.79	251.65	546.2	55.58	143.95	1115.9
1280	6401	146.13	54.74	7976.9	39.05	204.88	663.9	67.58	118.38	1367.2
1408	7745	205.52	38.93	11545.5	46.43	172.29	791.3	80.73	99.10	1642.5
1536	9217	279.54	28.62	16210.5	54.77	146.07	926.4	95.24	83.99	1937.2
1792	12545	502.82	15.91	29690.0	71.92	111.24	1244.8	126.33	63.33	2623.8
2048	16385	898.92	8.90	50281.7	91.84	87.11	1605.2	161.71	49.47	3409.1

Table-A III-4 Architectures of LiSA-DeiT models. All original DeiT transformer blocks are removed (#DeiT blocks = 0) and replaced with LiSA blocks

Stage	Resolution	LiSA-DeiT-T		LiSA-DeiT-S	
		LiSA blocks (res, dim, heads, #layers)	#DeiT blocks	LiSA blocks (res, dim, heads, #layers)	#DeiT blocks
Encoder	14×14	$(14 \times 14,\ 192,\ 3) \times 12$	0	$(14 \times 14,\ 384,\ 6) \times 12$	0

Table-A III-5 Architectures of LiSA-PVT models (part 1). At all stages, the original PVT transformer blocks are removed (#PVT blocks = 0) and replaced with LiSA blocks

Stage	Output	LiSA-PVT-T		LiSA-PVT-S	
		LiSA blocks (res, dim, heads, #layers)	#PVT blocks	LiSA blocks (res, dim, heads, #layers)	#PVT blocks
res1	56×56	$(56 \times 56,\ 64,\ 2) \times 2$	0	$(56 \times 56,\ 64,\ 2) \times 3$	0
res2	28×28	$(28 \times 28,\ 128,\ 4) \times 2$	0	$(28 \times 28,\ 128,\ 4) \times 4$	0
res3	14×14	$(14 \times 14,\ 320,\ 10) \times 2$	0	$(14 \times 14,\ 320,\ 10) \times 6$	0
res4	7×7	$(7 \times 7,\ 512,\ 16) \times 2$	0	$(7 \times 7,\ 512,\ 16) \times 3$	0

Table-A III-6 Architecture of the LiSA-Swin model. At all stages, the original Swin transformer blocks are removed (#Swin blocks = 0) and replaced with LiSA blocks

Stage	Output	LiSA-Swin-T	
		LiSA blocks (win, dim, heads, #layers)	#Swin blocks
res1	56×56	$(7 \times 7,\ 96,\ 3) \times 2$	0
res2	28×28	$(7 \times 7,\ 192,\ 6) \times 2$	0
res3	14×14	$(7 \times 7,\ 384,\ 12) \times 6$	0
res4	7×7	$(7 \times 7,\ 768,\ 24) \times 2$	0

BIBLIOGRAPHY

Abnar, S. & Zuidema, W. (2020). Quantifying attention flow in transformers. *ACL*, pp. 4190–4197.

Ali, A., Touvron, H., Caron, M., Bojanowski, P., Douze, M., Joulin, A., Laptev, I., Neverova, N., Synnaeve, G., Verbeek, J. et al. (2021). Xcit: Cross-covariance image transformers. *NeurIPS*, 34, 20014–20027.

Arar, M., Shamir, A. & Bermano, A. H. (2022). Learned Queries for Efficient Local Attention. *CVPR*.

Assran, M., Caron, M., Misra, I., Bojanowski, P., Bordes, F., Vincent, P., Joulin, A., Rabbat, M. & Ballas, N. (2022). Masked siamese networks for label-efficient learning. *ECCV*, pp. 456–473.

Baevski, A., Hsu, W.-N., Xu, Q., Babu, A., Gu, J. & Auli, M. (2022). Data2vec: A general framework for self-supervised learning in speech, vision and language. *ICML*, pp. 1298–1312.

Bahri, A., Yazdanpanah, M., Dastani, S., Noori, M., Vargas Hakim, G. A., Osowiechi, D., Beizaee, F., Ayed, I. B. & Desrosiers, C. (2025a). SMART-PC: Skeletal Model Adaptation for Robust Test-Time Training in Point Clouds. *arXiv preprint arXiv:2505.19546*.

Bahri, A., Yazdanpanah, M., Noori, M., Dastani, S., Cheraghalikhani, M., Osowiechi, D., Beizaee, F., Vargas Hakim, G. A., Ayed, I. B. & Desrosiers, C. (2025b). Test-time adaptation in point clouds: Leveraging sampling variation with weight averaging. *WACV*, pp. 266–275.

Bao, H., Dong, L. & Wei, F. (2021). Beit: Bert pre-training of image transformers. *arXiv preprint arXiv:2106.08254*.

Bayliss, A., Goldstein, C. I. & Turkel, E. (1983). An iterative method for the Helmholtz equation. *Journal of Computational Physics*, 49(3), 443–457.

Bell, H. E. (1965). Gershgorin's theorem and the zeros of polynomials. *The American Mathematical Monthly*, 72(3), 292–295.

Blelloch, G. E. (1990). Prefix sums and their applications.

Bodla, N., Singh, B., Chellappa, R. & Davis, L. S. (2017). Soft-NMS–improving object detection with one line of code. *ICCV*, pp. 5561–5569.

Bolya, D., Fu, C.-Y., Dai, X., Zhang, P., Feichtenhofer, C. & Hoffman, J. (2022a). Token merging: Your vit but faster. *arXiv preprint arXiv:2210.09461*.

Bolya, D., Fu, C.-Y., Dai, X., Zhang, P. & Hoffman, J. (2022b). Hydra attention: Efficient attention with many heads. *arXiv preprint arXiv:2209.07484*.

Boudiaf, M., Mueller, R., Ben Ayed, I. & Bertinetto, L. (2022). Parameter-free online test-time adaptation. *CVPR*, pp. 8344–8353.

Brown, T., Mann, B., Ryder, N., Subbiah, M., Kaplan, J. D., Dhariwal, P., Neelakantan, A., Shyam, P., Sastry, G., Askell, A. et al. (2020). Language models are few-shot learners. *NeurIPS*, 33, 1877–1901.

Cai, H., Gan, C. & Han, S. (2022). EfficientViT: Enhanced Linear Attention for High-Resolution Low-Computation Visual Recognition. *arXiv preprint arXiv:2205.14756*.

Cai, Z. & Vasconcelos, N. (2019). Cascade R-CNN: high quality object detection and instance segmentation. *TPAMI*, 43(5), 1483–1498.

Carion, N., Massa, F., Synnaeve, G., Usunier, N., Kirillov, A. & Zagoruyko, S. (2020). End-to-end object detection with transformers. *ECCV*, pp. 213–229.

Caron, M., Misra, I., Mairal, J., Goyal, P., Bojanowski, P. & Joulin, A. (2020). Unsupervised learning of visual features by contrasting cluster assignments. *NeurIPS*, 33, 9912–9924.

Caron, M., Touvron, H., Misra, I., Jégou, H., Mairal, J., Bojanowski, P. & Joulin, A. (2021). Emerging properties in self-supervised vision transformers. *ICCV*, pp. 9650–9660.

Cha, J., Chun, S., Lee, K., Cho, H.-C., Park, S., Lee, Y. & Park, S. (2021). SWAD: Domain generalization by seeking flat minima. *NeurIPS*, 34, 22405–22418.

Chen, C.-F., Fan, Q. & Panda, R. (2021a). Crossvit: Cross-attention multi-scale vision transformer for image classification. *ICCV*, pp. 347–356.

Chen, C.-F. R., Panda, R. & Fan, Q. (2022a). RegionViT: Regional-to-Local Attention for Vision Transformers. *ICLR*.

Chen, D., Wang, D., Darrell, T. & Ebrahimi, S. (2022b). Contrastive test-time adaptation. *CVPR*, pp. 295–305.

Chen, K., Wang, J., Pang, J., Cao, Y., Xiong, Y., Li, X., Sun, S., Feng, W., Liu, Z., Xu, J., Zhang, Z., Cheng, D., Zhu, C., Cheng, T., Zhao, Q., Li, B., Lu, X., Zhu, R., Wu, Y., Dai, J., Wang, J., Shi, J., Ouyang, W., Loy, C. C. & Lin, D. (2019). MMDetection: Open MMLab Detection Toolbox and Benchmark. *arXiv preprint arXiv:1906.07155*.

Chen, S., Ge, C., Tong, Z., Wang, J., Song, Y., Wang, J. & Luo, P. (2022c). Adaptformer: Adapting vision transformers for scalable visual recognition. *NeurIPS*, 35, 16664–16678.

Chen, T., Kornblith, S., Norouzi, M. & Hinton, G. (2020). A simple framework for contrastive learning of visual representations. *ICML*, pp. 1597–1607.

Chen, X., Ding, M., Wang, X., Xin, Y., Mo, S., Wang, Y., Han, S., Luo, P., Zeng, G. & Wang, J. (2024). Context autoencoder for self-supervised representation learning. *IJCV*, 132(1), 208–223.

Chen, X., Xie, S. & He, K. (2021b). An empirical study of training self-supervised visual transformers. *arXiv preprint arXiv:2104.02057*.

Chen, Y., Liu, Y., Jiang, D., Zhang, X., Dai, W., Xiong, H. & Tian, Q. (2022d). Sdae: Self-distillated masked autoencoder. *ECCV*, pp. 108–124.

Chen, Y., Dai, X., Chen, D., Liu, M., Dong, X., Yuan, L. & Liu, Z. (2022e). Mobile-former: Bridging mobilenet and transformer. *CVPR*, pp. 5270–5279.

Chen, Y., Li, J., Xiao, H., Jin, X., Yan, S. & Feng, J. (2017). Dual path networks. *NeurIPS*, 30, 4467–4475.

Cheng, B., Schwing, A. & Kirillov, A. (2021). Per-pixel classification is not all you need for semantic segmentation. *NeurIPS*, 34, 17864–17875.

Choi, J., Wi, H., Kim, J., Shin, Y., Lee, K., Trask, N. & Park, N. (2024). Graph convolutions enrich the self-attention in transformers! *NeurIPS*, 37, 52891–52936.

Chollet, F. (2017). Xception: Deep learning with depthwise separable convolutions. *CVPR*, pp. 1251–1258.

Choromanski, K., Likhosherstov, V., Dohan, D., Song, X., Gane, A., Sarlos, T., Hawkins, P., Davis, J., Mohiuddin, A., Kaiser, L. et al. (2021). Rethinking attention with performers. *ICLR*.

Cohen, T. & Welling, M. (2016). Group equivariant convolutional networks. *ICML*, pp. 2990–2999.

Cohen, T. S. & Welling, M. (2017). Steerable CNNs. *ICLR*.

Courant, R. & Hilbert, D. (1937). *Methods of mathematical physics: partial differential equations*. Interscience Publishers.

Cubuk, E. D., Zoph, B., Shlens, J. & Le, Q. V. (2020). Randaugment: Practical automated data augmentation with a reduced search space. *CVPRW*, pp. 702–703.

Dai, J., Qi, H., Xiong, Y., Li, Y., Zhang, G., Hu, H. & Wei, Y. (2017). Deformable convolutional networks. *ICCV*, pp. 764–773.

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

Dao, T. (2023). FlashAttention-2: Faster Attention with Better Parallelism and Work Partitioning. *ICLR*.

Dao, T., Fu, D., Ermon, S., Rudra, A. & Ré, C. (2022). Flashattention: Fast and memory-efficient exact attention with io-awareness. *NeurIPS*, 35, 16344–16359.

Dastani, S., Bahri, A., Vargas Hakim, G. A., Yazdanpanah, M., Noori, M., Osowiechi, D., Barbeau, S., Ayed, I., Lombaert, H. & Desrosiers, C. (2025a). TRUST: Test-Time Refinement using Uncertainty-Guided SSM Traverses. *NeurIPS*, 38, 4151–4175.

Dastani, S., Bahri, A., Yazdanpanah, M., Noori, M., Osowiechi, D., Vargas Hakim, G. A., Beizaee, F., Cheraghalikhani, M., Mondal, A. K., Lombaert, H. et al. (2025b). Spectral State Space Model for Rotation-Invariant Visual Representation Learning. *CVPR*, pp. 23881–23890.

Deng, J., Dong, W., Socher, R., Li, L.-J., Li, K. & Fei-Fei, L. (2009). Imagenet: A large-scale hierarchical image database. *CVPR*, pp. 248–255.

Devlin, J., Chang, M.-W., Lee, K. & Toutanova, K. (2018). Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*.

Dieleman, S., De Fauw, J. & Kavukcuoglu, K. (2016). Exploiting cyclic symmetry in convolutional neural networks. *ICML*.

Ding, X., Zhang, X., Han, J. & Ding, G. (2022). Scaling up your kernels to 31x31: Revisiting large kernel design in cnns. *CVPR*, pp. 11963–11975.

Doersch, C., Gupta, A. & Efros, A. A. (2015). Unsupervised visual representation learning by context prediction. *ICCV*, pp. 1422–1430.

Dong, X., Bao, J., Zhang, T., Chen, D., Zhang, W., Yuan, L., Chen, D., Wen, F. & Yu, N. (2021). PeCo: Perceptual Codebook for BERT Pre-training of Vision Transformers. *arXiv preprint arXiv:2111.12710*.

Dong, X., Bao, J., Zhang, T., Chen, D., Zhang, W., Yuan, L., Chen, D., Wen, F. & Yu, N. (2022). Bootstrapped Masked Autoencoders for Vision BERT Pretraining. *ECCV*, pp. 247–264.

Dosovitskiy, A., Beyer, L., Kolesnikov, A. et al. (2021). An Image is Worth 16x16 Words: Transformers for Image Recognition at Scale. *ICLR*.

Duda, R. O., Hart, P. E. & Stork, D. G. (2000). *Pattern Classification* (ed. 2nd). John Wiley and Sons.

Elfwing, S., Uchibe, E. & Doya, K. (2018). Sigmoid-weighted linear units for neural network function approximation in reinforcement learning. *Neural Networks*, 107, 3–11.

Erel, Y., Dünkel, O., Dabral, R., Golyanik, V., Theobalt, C. & Bermano, A. (2026). Attention (as discrete-time markov) chains. *NeurIPS*, 38, 54658–54690.

Everingham, M., Van Gool, L., Williams, C. K., Winn, J. & Zisserman, A. (2010). The pascal visual object classes (VOC) challenge. *IJCV*, 88(2), 303–338.

Fan, H., Xiong, B., Mangalam, K., Li, Y., Yan, Z., Malik, J. & Feichtenhofer, C. (2021). Multiscale vision transformers. *ICCV*, pp. 6824–6835.

Fan, Q., Huang, H., Zhou, X. & He, R. (2023). Lightweight Vision Transformer with Bidirectional Interaction. *NeurIPS*.

Fan, Q., Huang, H., Chen, M., Liu, H. & He, R. (2024). RMT: Retentive Networks Meet Vision Transformers. *CVPR*.

Fang, J., Xie, L., Wang, X., Zhang, X., Liu, W. & Tian, Q. (2022). Msg-transformer: Exchanging local spatial information by manipulating messenger tokens. *CVPR*, pp. 12063–12072.

Feng, A., Li, I., Jiang, Y. & Ying, R. (2023). Diffuser: efficient transformers with multi-hop attention diffusion for long sequences. *AAAI*, 37(11), 12772–12780.

Fu, D. Y., Dao, T., Saab, K. K., Thomas, A. W., Rudra, A. & Ré, C. (2022). Hungry hungry hippos: Towards language modeling with state space models. *arXiv preprint arXiv:2212.14052*.

Fu, D. Y., Epstein, E. L., Nguyen, E., Thomas, A. W., Zhang, M., Dao, T., Rudra, A. & Ré, C. (2023). Simple Hardware-Efficient Long Convolutions for Sequence Modeling. *ICML*, pp. 10373–10391.

Gao, P., Ma, T., Li, H., Lin, Z., Dai, J. & Qiao, Y. (2022). MCMAE: Masked convolution meets masked autoencoders. *NeurIPS*, 35, 35632–35644.

Gao, Z., Zhang, X.-Y. & Liu, C.-L. (2024, June). Unified Entropy Optimization for Open-Set Test-Time Adaptation. *CVPR*, pp. 23975–23984.

Gidaris, S., Singh, P. & Komodakis, N. (2018). Unsupervised Representation Learning by Predicting Image Rotations. *ICLR*.

Goel, K., Gu, A., Donahue, C. & Ré, C. (2022). It's raw! audio generation with state-space models. *ICML*, pp. 7616–7633.

Golub, G. H. & Van Loan, C. F. (2013). *Matrix computations*. JHU press.

Gong, T., Kim, Y., Lee, T., Chottananurak, S. & Lee, S.-J. (2024). SoTTA: Robust Test-Time Adaptation on Noisy Data Streams. *NeurIPS*, 36.

Goyal, S., Sun, M., Raghunanthan, A. & Kolter, Z. (2022). Test-time adaptation via conjugate pseudo-labels. *NeurIPS*.

Grill, J.-B., Strub, F., Altché, F., Tallec, C., Richemond, P., Buchatskaya, E., Doersch, C., Avila Pires, B., Guo, Z., Gheshlaghi Azar, M. et al. (2020). Bootstrap your own latent-a new approach to self-supervised learning. *NeurIPS*, 33, 21271–21284.

Gu, A. & Dao, T. (2023). Mamba: Linear-time sequence modeling with selective state spaces. *arXiv preprint arXiv:2312.00752*.

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

Gu, A., Goel, K. & Ré, C. (2021a). Efficiently modeling long sequences with structured state spaces. *arXiv preprint arXiv:2111.00396*.

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

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

Guo, J., Han, K., Wu, H., Xu, C., Tang, Y., Xu, C. & Wang, Y. (2022). CMT: Convolutional neural networks meet vision transformers. *CVPR*.

Gupta, A., Gu, A. & Berant, J. (2022). Diagonal state spaces are as effective as structured state spaces. *NeurIPS*, 35, 22982–22994.

Han, D., Pan, X., Han, Y., Song, S. & Huang, G. (2023a). FLatten Transformer: Vision Transformer using Focused Linear Attention. *ICCV*.

Han, D., Pu, Y., Xia, Z., Han, Y., Pan, X., Li, X., Lu, J., Song, S. & Huang, G. (2024a). Bridging the Divide: Reconsidering Softmax and Linear Attention. *NeurIPS*.

Han, D., Wang, Z., Xia, Z., Han, Y., Pu, Y., Ge, C., Song, J., Song, S., Zheng, B. & Huang, G. (2024b). Demystify Mamba in Vision: A Linear Attention Perspective. *NeurIPS*.

Han, D., Ye, T., Han, Y., Xia, Z., Song, S. & Huang, G. (2024c). Agent attention: On the integration of softmax and linear attention. *ECCV*.

Han, Q., Fan, Z., Dai, Q., Sun, L., Cheng, M.-M., Liu, J. & Wang, J. (2021a). On the Connection between Local Attention and Dynamic Depth-wise Convolution. *ICLR*.

Han, Q., Fan, Z., Dai, Q., Sun, L., Cheng, M.-M., Liu, J. & Wang, J. (2021b). Demystifying local vision transformer: Sparse connectivity, weight sharing, and dynamic weight. *arXiv preprint arXiv:2106.04263*.

Han, Y., Huang, G., Song, S., Yang, L., Wang, H. & Wang, Y. (2021c). Dynamic neural networks: A survey. *TPAMI*.

Han, Y., Pu, Y., Lai, Z., Wang, C., Song, S., Cao, J., Huang, W., Deng, C. & Huang, G. (2022). Learning to weight samples for dynamic early-exiting networks. *ECCV*.

Han, Y., Han, D., Liu, Z., Wang, Y., Pan, X., Pu, Y., Deng, C., Feng, J., Song, S. & Huang, G. (2023b). Dynamic Perceiver for Efficient Visual Recognition. *ICCV*.

Hasani, R., Lechner, M., Wang, T.-H., Chahine, M., Amini, A. & Rus, D. (2022). Liquid Structural State-Space Models. *ICLR*.

Hassani, A., Walton, S., Li, J., Li, S. & Shi, H. (2023). Neighborhood Attention Transformer. *CVPR*.

Hatamizadeh, A. & Kautz, J. (2024). Mambavision: A hybrid mamba-transformer vision backbone. *arXiv preprint arXiv:2407.08083*.

Hatamizadeh, A., Yin, H., Heinrich, G., Kautz, J. & Molchanov, P. (2023). Global context vision transformers. *ICML*.

He, K., Zhang, X., Ren, S. & Sun, J. (2016). Deep residual learning for image recognition. *CVPR*, pp. 770–778.

He, K., Gkioxari, G., Dollár, P. & Girshick, R. (2017). Mask r-cnn. *ICCV*.

He, K., Fan, H., Wu, Y., Xie, S. & Girshick, R. (2020). Momentum contrast for unsupervised visual representation learning. *CVPR*, pp. 9729–9738.

He, K., Chen, X., Xie, S., Li, Y., Dollár, P. & Girshick, R. (2022). Masked autoencoders are scalable vision learners. *CVPR*, pp. 16000–16009.

Hendrycks, D. & Dietterich, T. (2019). Benchmarking Neural Network Robustness to Common Corruptions and Perturbations. *ICLR*.

Hendrycks, D., Basart, S., Mu, N., Kadavath, S., Wang, F., Dorundo, E., Desai, R., Zhu, T., Parajuli, S., Guo, M. et al. (2021). The many faces of robustness: A critical analysis of out-of-distribution generalization. *ICCV*, pp. 8340–8349.

Hinton, G., Vinyals, O. & Dean, J. (2015). Distilling the knowledge in a neural network. *arXiv preprint arXiv:1503.02531*.

Hinton, G. E., Krizhevsky, A. & Wang, S. D. (2018). Matrix capsules with EM routing. *ICLR*.

Howard, A., Sandler, M., Chu, G., Chen, L.-C., Chen, B., Tan, M., Wang, W., Zhu, Y., Pang, R., Vasudevan, V. et al. (2019). Searching for mobilenetv3. *ICCV*, pp. 1314–1324.

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

Hu, E. J., Shen, Y., Wallis, P., Allen-Zhu, Z., Li, Y., Wang, S., Wang, L., Chen, W. et al. (2022). Lora: Low-rank adaptation of large language models. *ICLR*, 1(2), 3.

Hua, B.-S., Tran, M.-K. & Yeung, S.-K. (2018). Pointwise convolutional neural networks. *CVPR*, pp. 984–993.

Huang, G., Liu, Z., Van Der Maaten, L. & Weinberger, K. Q. (2017). Densely connected convolutional networks. *CVPR*, pp. 4700–4708.

Huang, L., You, S., Zheng, M., Wang, F., Qian, C. & Yamasaki, T. (2022). Green hierarchical vision transformer for masked image modeling. *NeurIPS*, 35, 19997–20010.

Huang, T., Pei, X., You, S., Wang, F., Qian, C. & Xu, C. (2024). Localmamba: Visual state space model with windowed selective scan. *arXiv preprint arXiv:2403.09338*.

Huo, Z., Gu, B. & Huang, H. (2021). Large Batch Optimization for Deep Learning Using New Complete Layer-Wise Adaptive Rate Scaling. *AAAI*, pp. 7883–7890.

Ildiz, M. E., Huang, Y., Li, Y., Rawat, A. S. & Oymak, S. (2024). From self-attention to markov models: Unveiling the dynamics of generative transformers. *arXiv preprint arXiv:2402.13512*.

Izmailov, P., Podoprikhin, D., Garipov, T., Vetrov, D. & Wilson, A. G. (2018). Averaging weights leads to wider optima and better generalization. *UAI*, pp. 876–885.

Jacob, B., Kligys, S., Chen, B., Zhu, M., Tang, M., Howard, A., Adam, H. & Kalenichenko, D. (2018). Quantization and training of neural networks for efficient integer-arithmetic-only inference. *CVPR*, pp. 2704–2713.

Jaderberg, M., Simonyan, K., Zisserman, A. & Kavukcuoglu, K. (2015). Spatial transformer networks. *NeurIPS*, pp. 2017–2025.

Jang, D.-H., Yun, S. & Han, D. (2024a). Model stock: All we need is just a few fine-tuned models. *ECCV*, pp. 207–223.

Jang, M., Chung, S.-Y. & Chung, H. W. (2024b). Test-Time Adaptation via Self-Training with Nearest Neighbor Information. *ICLR*.

Kaba, S.-O., Mondal, A. K., Zhang, Y., Bengio, Y. & Ravanbakhsh, S. (2023). Equivariance with Learned Canonicalization Functions. *ICLR*.

Kakogeorgiou, I., Gidaris, S., Psomas, B., Avrithis, Y., Bursuc, A., Karantzalos, K. & Komodakis, N. (2022). What to hide from your students: Attention-guided masked image modeling. *ECCV*, pp. 300–318.

Karmanov, A., Guan, D., Lu, S., El Saddik, A. & Xing, E. (2024). Efficient test-time adaptation of vision-language models. *CVPR*, pp. 14162–14171.

Katharopoulos, A., Vyas, A., Pappas, N. & Fleuret, F. (2020). Transformers are rnns: Fast autoregressive transformers with linear attention. *ICML*, pp. 5156–5165.

Keskar, N. S., Nocedal, J., Tang, P. T. P., Mudigere, D. & Smelyanskiy, M. (2017). On large-batch training for deep learning: Generalization gap and sharp minima. *ICLR*.

Kirillov, A., Girshick, R., He, K. & Dollár, P. (2019). Panoptic feature pyramid networks. *CVPR*.

Krishnamoorthi, R. (2018). Quantizing deep convolutional networks for efficient inference: A whitepaper. *arXiv preprint arXiv:1806.08342*.

Kálmán, R. E. (1960). A new approach to linear filtering and prediction problems. *Journal of Basic Engineering*, 82(1), 35–45.

Ladner, R. E. & Fischer, M. J. (1980). Parallel prefix computation. *JACM*, 27(4), 831–838.

Langley, P. (2000). Crafting Papers on Machine Learning. *ICML*, pp. 1207–1216.

Laptev, D., Savinov, N., Buhmann, J. M. & Pollefeys, M. (2016). TI-pooling: transformation-invariant pooling for feature learning in convolutional neural networks. *CVPR*, pp. 289–297.

LeCun, Y., Boser, B., Denker, J. S., Henderson, D., Howard, R. E., Hubbard, W. & Jackel, L. D. (1989). Backpropagation applied to handwritten zip code recognition. *Neural computation*, 1(4), 541–551.

LeCun, Y., Bottou, L., Bengio, Y. & Haffner, P. (1998). Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11), 2278–2324.

LeCun, Y., Bengio, Y. & Hinton, G. (2015). Deep learning. *nature*, 436–444.

Lee, J., Das, D., Choo, J. & Choi, S. (2023, October). Towards Open-Set Test-Time Adaptation Utilizing the Wisdom of Crowds in Entropy Minimization. *ICCV*, pp. 16380–16389.

Lee, Y., Kim, J., Willette, J. & Hwang, S. J. (2022). MPViT: Multi-Path Vision Transformer for Dense Prediction. *CVPR*.

Li, D., Yang, Y., Song, Y.-Z. & Hospedales, T. M. (2017). Deeper, broader and artier domain generalization. *ICCV*, pp. 5542–5550.

Li, D., Yang, Y., Song, Y.-Z. & Hospedales, T. (2018). Learning to generalize: Meta-learning for domain generalization. *AAAI*, 32(1).

Li, G., Zheng, H., Liu, D., Wang, C., Su, B. & Zheng, C. (2022a). Semmae: Semantic-guided masking for learning masked autoencoders. *NeurIPS*, 35, 14290–14302.

Li, X., Ge, Y., Yi, K., Hu, Z., Shan, Y. & Duan, L.-Y. (2022b). mc-beit: Multi-choice discretization for image bert pre-training. *ECCV*, pp. 231–246.

Li, Y., Wu, C.-Y., Fan, H., Mangalam, K., Xiong, B., Malik, J. & Feichtenhofer, C. (2021a). Improved multiscale vision transformers for classification and detection. *arXiv preprint arXiv:2112.01526*.

Li, Y., Xie, S., Chen, X., Dollar, P., He, K. & Girshick, R. (2021b). Benchmarking detection transfer learning with vision transformers. *arXiv:2111.11429*.

Li, Y., Mao, H., Girshick, R. & He, K. (2022c). Exploring plain vision transformer backbones for object detection. *ECCV*, pp. 280–296.

Li, Y., Wu, C.-Y., Fan, H., Mangalam, K., Xiong, B., Malik, J. & Feichtenhofer, C. (2022d). MViTv2: Improved multiscale vision transformers for classification and detection. *CVPR*.

Li, Y., Xu, S., Zhang, B., Cao, X., Gao, P. & Guo, G. (2022e). Q-vit: Accurate and fully quantized low-bit vision transformer. *NeurIPS*, 35, 34451–34463.

Li, Y., Yuan, G., Wen, Y., Hu, J., Evangelidis, G., Tulyakov, S., Wang, Y. & Ren, J. (2022f). Efficientformer: Vision transformers at mobilenet speed. *NeurIPS*, 35, 12934–12949.

Li, Y., Cai, T., Zhang, Y., Chen, D. & Dey, D. (2022g). What Makes Convolutional Models Great on Long Sequence Modeling? *ICLR*.

Liang, J., Hu, D. & Feng, J. (2020). Do we really need to access the source data? source hypothesis transfer for unsupervised domain adaptation. *ICML*, pp. 6028–6039.

Liang, Y., Ge, C., Tong, Z., Song, Y., Wang, J. & Xie, P. (2022). Not all patches are what you need: Expediting vision transformers via token reorganizations. *arXiv preprint arXiv:2202.07800*.

Lin, T.-Y., Maire, M., Belongie, S., Hays, J., Perona, P., Ramanan, D., Dollár, P. & Zitnick, C. L. (2014). Microsoft coco: Common objects in context. *ECCV*.

Lin, T., Dollár, P., Girshick, R. B., He, K., Hariharan, B. & Belongie, S. J. (2017a). Feature Pyramid Networks for Object Detection. *CVPR*, pp. 936–944.

Lin, Z., Feng, M., dos Santos, C. N., Yu, M., Xiang, B., Zhou, B. & Bengio, Y. (2017b). A Structured Self-attentive Sentence Embedding. Retrieved from: https://arxiv.org/abs/1703.03130.

Liu, F., Zhang, X., Peng, Z., Guo, Z., Wan, F., Ji, X. & Ye, Q. (2023a). Integrally migrating pre-trained transformer encoder-decoders for visual object detection. *ICCV*, pp. 6825–6834.

Liu, H., Jiang, X., Li, X., Guo, A., Hu, Y., Jiang, D. & Ren, B. (2023b). The devil is in the frequency: Geminated gestalt autoencoder for self-supervised visual pre-training. *AAAI*, 37(2), 1649–1656.

Liu, J., Tripathi, S., Kurup, U. & Shah, M. (2020). Pruning algorithms to accelerate convolutional neural networks for edge applications: A survey. *arXiv preprint arXiv:2005.04275*.

Liu, K., Wu, T., Liu, C. & Guo, G. (2022a). Dynamic Group Transformer: A General Vision Transformer Backbone with Dynamic Group Attention. *IJCAI*.

Liu, S., Chen, T., Chen, X., Chen, X., Xiao, Q., Wu, B., Kärkkäinen, T., Pechenizkiy, M., Mocanu, D. C. & Wang, Z. (2023c). More ConvNets in the 2020s: Scaling up Kernels Beyond 51x51 using Sparsity. *ICLR*.

Liu, X., Zhou, J., Kong, T., Lin, X. & Ji, R. (2024a). Exploring target representations for masked autoencoders. *ICLR*.

Liu, Y., Tian, Y., Zhao, Y., Yu, H., Xie, L., Wang, Y., Ye, Q. & Liu, Y. (2024b). VMamba: Visual State Space Model. *arXiv preprint arXiv:2401.10166*.

Liu, Z., Lin, Y., Cao, Y., Hu, H., Wei, Y., Zhang, Z., Lin, S. & Guo, B. (2021). Swin Transformer: Hierarchical Vision Transformer using Shifted Windows. *ICCV*, pp. 10012–10022.

Liu, Z., Mao, H., Wu, C.-Y., Feichtenhofer, C., Darrell, T. & Xie, S. (2022b). A convnet for the 2020s. *CVPR*, pp. 11976–11986.

Long, S., Zhou, Q., Li, X., Lu, X., Ying, C., Luo, Y., Ma, L. & Yan, S. (2024). DGMamba: Domain generalization via generalized state space model. *ACM MM*, pp. 3607–3616.

Loshchilov, I. & Hutter, F. (2017). Decoupled weight decay regularization. *arXiv preprint arXiv:1711.05101*.

Lu, C., Schroecker, Y., Gu, A., Parisotto, E., Foerster, J., Singh, S. & Behbahani, F. (2023). Structured state space models for in-context reinforcement learning. *NeurIPS*, 36.

Lu, J., Yao, J., Zhang, J., Zhu, X., Xu, H., Gao, W., Xu, C., Xiang, T. & Zhang, L. (2021a). SOFT: softmax-free transformer with linear complexity. *NeurIPS*, 34, 21297–21309.

Lu, J., Mottaghi, R., Kembhavi, A. et al. (2021b). Container: Context Aggregation Networks. *NeurIPS*, 34, 19160–19171.

Luo, W., Li, Y., Urtasun, R. & Zemel, R. (2016). Understanding the effective receptive field in deep convolutional neural networks. *NeurIPS*, 29, 4898–4906.

Ma, J., Li, F. & Wang, B. (2024a). U-mamba: Enhancing long-range dependency for biomedical image segmentation. *arXiv preprint arXiv:2401.04722*.

Ma, N., Goldstein, M., Albergo, M. S., Boffi, N. M., Vanden-Eijnden, E. & Xie, S. (2024b). Sit: Exploring flow and diffusion-based generative models with scalable interpolant transformers. *ECCV*, pp. 23–40.

Ma, X., Zhou, C., Kong, X., He, J., Gui, L., Neubig, G., May, J. & Zettlemoyer, L. (2022). Mega: Moving Average Equipped Gated Attention. *ICLR*.

Martin, E. & Cundy, C. (2018). Parallelizing Linear Recurrent Neural Nets Over Sequence Length. *ICLR*.

Mehta, H., Gupta, A., Cutkosky, A. & Neyshabur, B. (2022). Long range language modeling via gated state spaces. *arXiv preprint arXiv:2206.13947*.

Mehta, S. & Rastegari, M. (2022). Mobilevit: light-weight, general-purpose, and mobile-friendly vision transformer. *ICLR*.

Michel, P., Levy, O. & Neubig, G. (2019). Are sixteen heads really better than one? *NeurIPS*, 32.

MMSegmentation Contributors. (2020). MMSegmentation: OpenMMLab Semantic Segmentation Toolbox and Benchmark. Retrieved from: https://github.com/open-mmlab/mmsegmentation.

Mondal, A. K., Panigrahi, S. S., Kaba, O., Mudumba, S. R. & Ravanbakhsh, S. (2023). Equivariant adaptation of large pretrained models. *NeurIPS*, 36, 50293–50309.

Mottaghi, R., Chen, X., Liu, X., Cho, N.-G., Lee, S.-W., Fidler, S., Urtasun, R. & Yuille, A. (2014). The role of context for object detection and semantic segmentation in the wild. *CVPR*, pp. 891–898.

Ng, A., Jordan, M. & Weiss, Y. (2001). On spectral clustering: Analysis and an algorithm. *NeurIPS*, 14.

Nguyen, E., Goel, K., Gu, A., Downs, G., Shah, P., Dao, T., Baccus, S. & Ré, C. (2022). S4nd: Modeling images and videos as multidimensional signals with state spaces. *NeurIPS*, 35, 2846–2861.

Ni, Z., Wang, Y., Yu, J., Jiang, H., Cao, Y. & Huang, G. (2023). Deep Incubation: Training Large Models by Divide-and-Conquering. *ICCV*.

Niu, S., Wu, J., Zhang, Y., Chen, Y., Zheng, S., Zhao, P. & Tan, M. (2022, 17–23 Jul). Efficient Test-Time Model Adaptation without Forgetting. *ICML*, 162(Proceedings of Machine Learning Research), 16888–16905.

Niu, S., Wu, J., Zhang, Y., Wen, Z., Chen, Y., Zhao, P. & Tan, M. (2023). Towards Stable Test-Time Adaptation in Dynamic Wild World. *ICLR*.

Niu, S., Miao, C., Chen, G., Wu, P. & Zhao, P. (2024). Test-Time Model Adaptation with Only Forward Passes. *ICML*, pp. 38298–38315.

Noori, M., Cheraghalikhani, M., Bahri, A., Vargas Hakim, G. A., Osowiechi, D., Yazdanpanah, M., Ben Ayed, I. & Desrosiers, C. (2025a, February). FDS: Feedback-Guided Domain Synthesis with Multi-Source Conditional Diffusion Models for Domain Generalization. *WACV*, pp. 8493–8503.

Noori, M., Osowiechi, D., Vargas Hakim, G. A., Bahri, A., Yazdanpanah, M., Dastani, S., Beizaee, F., Ayed, I. B. & Desrosiers, C. (2025b). Test-Time Adaptation of Vision-Language Models for Open-Vocabulary Semantic Segmentation. *arXiv preprint arXiv:2505.21844*.

Noroozi, M. & Favaro, P. (2016). Unsupervised learning of visual representations by solving jigsaw puzzles. *ECCV*, pp. 69–84.

Orvieto, A., Smith, S. L., Gu, A., Fernando, A., Gulcehre, C., Pascanu, R. & De, S. (2023). Resurrecting recurrent neural networks for long sequences. *ICML*, pp. 26670–26698.

Osowiechi, D., Noori, M., Vargas Hakim, G. A., Yazdanpanah, M., Bahri, A., Cheraghalikhani, M., Dastani, S., Beizaee, F., Ayed, I. B. & Desrosiers, C. (2024). WATT: Weight Average Test Time Adaptation of CLIP. *NeurIPS*.

Pan, X., Ge, C., Lu, R., Song, S., Chen, G., Huang, Z. & Huang, G. (2022). On the integration of self-attention and convolution. *CVPR*, pp. 815–825.

Pathak, D., Krahenbuhl, P., Donahue, J., Darrell, T. & Efros, A. A. (2016). Context encoders: Feature learning by inpainting. *CVPR*, pp. 2536–2544.

Peng, B., Alcaide, E., Anthony, Q., Albalak, A., Arcadinho, S., Cao, H., Cheng, X., Chung, M., Grella, M., GV, K. K. et al. (2023). RWKV: Reinventing RNNs for the Transformer Era. *EMNLP*, pp. 14048–14077.

Peng, Z., Huang, W., Gu, S., Xie, L., Wang, Y., Jiao, J. & Ye, Q. (2021). Conformer: Local Features Coupling Global Representations for Visual Recognition. *ICCV*.

Peng, Z., Dong, L., Bao, H., Ye, Q. & Wei, F. (2022). Beit v2: Masked image modeling with vector-quantized visual tokenizers. *arXiv preprint arXiv:2208.06366*.

Poli, M., Massaroli, S., Nguyen, E., Fu, D. Y., Dao, T., Baccus, S., Bengio, Y., Ermon, S. & Ré, C. (2023). Hyena hierarchy: Towards larger convolutional language models. *ICML*, pp. 28043–28078.

Puny, O., Atzmon, M., Ben-Hamu, H., Misra, I., Grover, A., Smith, E. J. & Lipman, Y. (2021). Frame averaging for invariant and equivariant network design. *arXiv preprint arXiv:2110.03336*.

Qin, Z., Han, X., Sun, W., He, B., Li, D., Li, D., Dai, Y., Kong, L. & Zhong, Y. (2022a). Toeplitz Neural Network for Sequence Modeling. *ICLR*.

Qin, Z., Han, X., Sun, W., Li, D., Kong, L., Barnes, N. & Zhong, Y. (2022b). The devil in linear transformer. *arXiv preprint arXiv:2210.10340*.

Qin, Z., Sun, W., Deng, H., Li, D., Wei, Y., Lv, B., Yan, J., Kong, L. & Zhong, Y. (2022c). cosFormer: Rethinking Softmax In Attention. *ICLR*. Retrieved from: https://openreview.net/forum?id=Bl8CQrx2Up4.

Radford, A., Kim, J. W., Hallacy, C., Ramesh, A., Goh, G., Agarwal, S., Sastry, G., Askell, A., Mishkin, P., Clark, J. et al. (2021). Learning transferable visual models from natural language supervision. *ICML*, pp. 8748–8763.

Radosavovic, I., Kosaraju, R. P., Girshick, R., He, K. & Dollár, P. (2020). Designing network design spaces. *CVPR*, pp. 10428–10436.

Ramesh, A., Pavlov, M., Goh, G., Gray, S., Voss, C., Radford, A., Chen, M. & Sutskever, I. (2021). Zero-shot text-to-image generation. *ICML*, pp. 8821–8831.

Rao, Y., Zhao, W., Liu, B., Lu, J., Zhou, J. & Hsieh, C.-J. (2021). Dynamicvit: Efficient vision transformers with dynamic token sparsification. *NeurIPS*, 34, 13937–13949.

Rao, Y., Zhao, W., Tang, Y., Zhou, J., Lim, S. N. & Lu, J. (2022). Hornet: Efficient high-order spatial interactions with recursive gated convolutions. *NeurIPS*, 35, 10353–10366.

Recht, B., Roelofs, R., Schmidt, L. & Shankar, V. (2019). Do imagenet classifiers generalize to imagenet? *ICML*, pp. 5389–5400.

Ren, S., He, K., Girshick, R. B. & Sun, J. (2015). Faster R-CNN: Towards Real-Time Object Detection with Region Proposal Networks. *NeurIPS*, 91–99.

Reuter, M., Wolter, F.-E. & Peinecke, N. (2005). Laplace-spectra as fingerprints for shape matching. *Proceedings of the 2005 ACM symposium on Solid and physical modeling*, pp. 101–106.

Reuter, M., Wolter, F.-E. & Peinecke, N. (2006). Laplace–Beltrami spectra as 'Shape-DNA'of surfaces and solids. *Computer-Aided Design*, 38(4), 342–366.

Sabour, S., Frosst, N. & Hinton, G. E. (2017). Dynamic routing between capsules. *NeurIPS*, 30.

Sandler, M., Howard, A., Zhu, M., Zhmoginov, A. & Chen, L.-C. (2018). Mobilenetv2: Inverted residuals and linear bottlenecks. *CVPR*, pp. 4510–4520.

Saon, G., Gupta, A. & Cui, X. (2023). Diagonal state space augmented transformers for speech recognition. *ICASSP*, pp. 1–5.

Schirmer, M., Zhang, D. & Nalisnick, E. (2024). Temporal Test-Time Adaptation with State-Space Models. *arXiv preprint arXiv:2407.12492*.

Shazeer, N. (2020). Glu variants improve transformer. *arXiv preprint arXiv:2002.05202*.

Shen, Z., Zhang, M., Zhao, H., Yi, S. & Li, H. (2021). Efficient Attention: Attention with Linear Complexities. *WACV*.

Shi, J. & Malik, J. (2000). Normalized cuts and image segmentation. *TPAMI*, 22(8), 888–905.

Shi, J., Wang, K. A. & Fox, E. (2023). Sequence modeling with multiresolution convolutional memory. *ICML*, pp. 31312–31327.

Shi, Y., Dong, M. & Xu, C. (2024). Multi-Scale VMamba: Hierarchy in Hierarchy Visual State Space Model. *arXiv preprint arXiv:2405.14174*.

Shu, M., Nie, W., Huang, D.-A., Yu, Z., Goldstein, T., Anandkumar, A. & Xiao, C. (2022). Test-time prompt tuning for zero-shot generalization in vision-language models. *NeurIPS*, 35, 14274–14289.

Si, C., Yu, W., Zhou, P., Zhou, Y., Wang, X. & YAN, S. (2022). Inception Transformer. *NeurIPS*.

Simonyan, K. & Zisserman, A. (2014). Very deep convolutional networks for large-scale image recognition. *ICLR*.

Smith, J. T., Warrington, A. & Linderman, S. W. (2022). Simplified state space layers for sequence modeling. *arXiv preprint arXiv:2208.04933*.

Srinivas, A., Lin, T.-Y., Parmar, N., Shlens, J., Abbeel, P. & Vaswani, A. (2021). Bottleneck transformers for visual recognition. *CVPR*, pp. 16519–16529.

Sun, H., Xu, L., Jin, S., Luo, P., Qian, C. & Liu, W. (2024). Program: Prototype graph model based pseudo-label learning for test-time adaptation. *ICLR*.

Sun, W., Qin, Z., Deng, H., Wang, J., Zhang, Y., Zhang, K., Barnes, N., Birchfield, S., Kong, L. & Zhong, Y. (2023a). Vicinity vision transformer. *TPAMI*.

Sun, Y., Dong, L., Huang, S., Ma, S., Xia, Y., Xue, J., Wang, J. & Wei, F. (2023b). Retentive network: A successor to transformer for large language models. *arXiv preprint arXiv:2307.08621*.

Szegedy, C., Vanhoucke, V., Ioffe, S., Shlens, J. & Wojna, Z. (2016). Rethinking the inception architecture for computer vision. *CVPR*, pp. 2818–2826.

Tan, M. & Le, Q. V. (2019). Efficientnet: Rethinking model scaling for convolutional neural networks. *ICML*, pp. 6105–6114.

Tang, S., Zhang, J., Zhu, S. & Tan, P. (2022). Quadtree Attention for Vision Transformers. *ICLR*. Retrieved from: https://openreview.net/forum?id=fR-EnKWL_Zb.

Tarvainen, A. & Valpola, H. (2017). Mean teachers are better role models: Weight-averaged consistency targets improve semi-supervised deep learning results. *NeurIPS*, 30, 1195–1204.

Tian, Y., Xie, L., Zhang, X., Fang, J., Xu, H., Huang, W., Jiao, J., Tian, Q. & Ye, Q. (2021). Semantic-Aware Generation for Self-Supervised Visual Representation Learning. *arXiv preprint arXiv:2111.13163*.

Tian, Y., Xie, L., Fang, J., Shi, M., Peng, J., Zhang, X., Jiao, J., Tian, Q. & Ye, Q. (2022). Beyond Masking: Demystifying Token-Based Pre-Training for Vision Transformers. *arXiv preprint arXiv:2203.14313*.

Tian, Y., Xie, L., Wang, Z., Wei, L., Zhang, X., Jiao, J., Wang, Y., Tian, Q. & Ye, Q. (2023). Integrally Pre-Trained Transformer Pyramid Networks. *CVPR*, pp. 18610–18620.

Tolstikhin, I. O., Houlsby, N., Kolesnikov, A., Beyer, L., Zhai, X., Unterthiner, T., Yung, J., Steiner, A., Keysers, D., Uszkoreit, J. et al. (2021). Mlp-mixer: An all-mlp architecture for vision. *NeurIPS*, 34, 24261–24272.

Touvron, H., Cord, M., Douze, M., Massa, F., Sablayrolles, A. & Jégou, H. (2021). Training data-efficient image transformers & distillation through attention. *ICML*, pp. 10347–10357.

Touvron, H., Bojanowski, P., Caron, M., Cord, M., El-Nouby, A., Grave, E., Izacard, G., Joulin, A., Synnaeve, G., Verbeek, J. et al. (2022). Resmlp: Feedforward networks for image classification with data-efficient training. *TPAMI*, 45(4), 5314–5321.

Vargas Hakim, G. A., Osowiechi, D., Noori, M., Cheraghalikhani, M., Bahri, A., Yazdanpanah, M., Ben Ayed, I. & Desrosiers, C. (2025, February). CLIPArTT: Adaptation of CLIP to New Domains at Test Time. *WACV*, pp. 7092–7101.

Vaswani, A., Shazeer, N., Parmar, N. et al. (2017). Attention is all you need. *NeurIPS*.

Vaswani, A., Ramachandran, P., Srinivas, A., Parmar, N., Hechtman, B. & Shlens, J. (2021). Scaling local self-attention for parameter efficient visual backbones. *CVPR*, pp. 12894–12904.

Vinyals, O., Blundell, C., Lillicrap, T., Wierstra, D. et al. (2016). Matching networks for one shot learning. *NeurIPS*, 29.

Wang, D., Shelhamer, E., Liu, S., Olshausen, B. & Darrell, T. (2021a). Tent: Fully Test-Time Adaptation by Entropy Minimization. *ICLR*.

Wang, G., Ying, R., Huang, J. & Leskovec, J. (2020a). Multi-hop attention graph neural network. *arXiv preprint arXiv:2009.14332*.

Wang, H., Ge, S., Lipton, Z. & Xing, E. P. (2019). Learning robust global representations by penalizing local predictive power. *NeurIPS*, 32.

Wang, J., Zhu, W., Wang, P., Yu, X., Liu, L., Omar, M. & Hamid, R. (2023a). Selective structured state-spaces for long-form video understanding. *CVPR*, pp. 6387–6397.

Wang, Q., Fink, O., Van Gool, L. & Dai, D. (2022a). Continual test-time domain adaptation. *CVPR*, pp. 7201–7211.

Wang, S., Li, B. Z., Khabsa, M., Fang, H. & Ma, H. (2020b). Linformer: Self-Attention with Linear Complexity.

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

Wang, W., Xie, E., Li, X., Fan, D.-P., Song, K., Liang, D., Lu, T., Luo, P. & Shao, L. (2022b). Pvtv2: Improved baselines with pyramid vision transformer. *Computational Visual Media*, 8(3), 1–10.

Wang, W., Yao, L., Chen, L., Lin, B., Cai, D., He, X. & Liu, W. (2022c). CrossFormer: A Versatile Vision Transformer Hinging on Cross-scale Attention. *ICLR*.

Wang, Y., Yue, Y., Lu, R., Liu, T., Zhong, Z., Song, S. & Huang, G. (2023b). Efficienttrain: Exploring generalized curriculum learning for training visual backbones. *ICCV*.

Wei, C., Xie, L., Ren, X., Xia, Y., Su, C., Liu, J., Tian, Q. & Yuille, A. L. (2019). Iterative reorganization with weak spatial constraints: Solving arbitrary jigsaw puzzles for unsupervised representation learning. *CVPR*, pp. 1910–1919.

Wei, C., Fan, H., Xie, S., Wu, C.-Y., Yuille, A. & Feichtenhofer, C. (2022a). Masked feature prediction for self-supervised visual pre-training. *CVPR*, pp. 14668–14678.

Wei, L., Xie, L., Zhou, W., Li, H. & Tian, Q. (2022b). Mvp: Multimodality-guided visual pre-training. *ECCV*, pp. 337–353.

Wei, Y., Hu, H., Xie, Z., Zhang, Z., Cao, Y., Bao, J., Chen, D. & Guo, B. (2022c). Contrastive Learning Rivals Masked Image Modeling in Fine-tuning via Feature Distillation. *arXiv preprint arXiv:2205.14141*.

Weiler, M. & Cesa, G. (2019). General E(2)-Equivariant Steerable CNNs. *NeurIPS*.

Weiler, M., Hamprecht, F. A. & Storath, M. (2018). Learning steerable filters for rotation equivariant CNNs. *CVPR*, pp. 849–858.

Wi, H., Choi, J. & Park, N. (2025). Learning advanced self-attention for linear transformers in the singular value domain. *arXiv preprint arXiv:2505.08516*.

Wortsman, M., Ilharco, G., Gadre, S. Y., Roelofs, R., Gontijo-Lopes, R., Morcos, A. S., Namkoong, H., Farhadi, A., Carmon, Y., Kornblith, S. et al. (2022). Model soups: averaging weights of multiple fine-tuned models improves accuracy without increasing inference time. *ICML*, pp. 23965–23998.

Wu, H., Xiao, B., Codella, N., Liu, M., Dai, X., Yuan, L. & Zhang, L. (2021). Cvt: Introducing convolutions to vision transformers. *ICCV*, pp. 22–31.

Wu, K., Zhang, J., Peng, H., Liu, M., Xiao, B., Fu, J. & Yuan, L. (2022). Tinyvit: Fast pretraining distillation for small vision transformers. *ECCV*, pp. 68–85.

Xia, Z., Pan, X., Song, S., Li, L. E. & Huang, G. (2022). Vision transformer with deformable attention. *CVPR*, pp. 4794–4803.

Xiao, T., Liu, Y., Zhou, B., Jiang, Y. & Sun, J. (2018). Unified perceptual parsing for scene understanding. *ECCV*, pp. 418–434.

Xiao, T., Singh, M., Mintun, E., Darrell, T., Dollár, P. & Girshick, R. (2021). Early convolutions help transformers see better. *NeurIPS*, 34, 30392–30400.

Xie, E., Wang, W., Yu, Z., Anandkumar, A., Alvarez, J. M. & Luo, P. (2021a). SegFormer: Simple and efficient design for semantic segmentation with transformers. *NeurIPS*, 34, 12077–12090.

Xie, F., Zhang, W., Wang, Z. & Ma, C. (2024). QuadMamba: Learning Quadtree-based Selective Scan for Visual State Space Model. *arXiv preprint arXiv:2410.06806*.

Xie, S., Girshick, R., Dollár, P., Tu, Z. & He, K. (2017). Aggregated residual transformations for deep neural networks. *CVPR*, pp. 1492–1500.

Xie, Z., Lin, Y., Zhang, Z., Cao, Y., Lin, S. & Hu, H. (2021b). Propagate yourself: Exploring pixel-level consistency for unsupervised visual representation learning. *CVPR*, pp. 16684–16693.

Xie, Z., Zhang, Z., Cao, Y., Lin, Y., Bao, J., Yao, Z., Dai, Q. & Hu, H. (2022). Simmim: A simple framework for masked image modeling. *CVPR*, pp. 9653–9663.

Xiong, Y., Zeng, Z., Chakraborty, R., Tan, M., Fung, G., Li, Y. & Singh, V. (2021). Nyströmformer: A nyström-based algorithm for approximating self-attention. *AAAI*, 35(16), 14138–14148.

Yang, C., Qiao, S., Yu, Q. et al. (2023a). MOAT: Alternating mobile convolution and attention brings strong vision models. *ICLR*.

Yang, C., Chen, Z., Espinosa, M., Ericsson, L., Wang, Z., Liu, J. & Crowley, E. J. (2024). Plainmamba: Improving non-hierarchical mamba in visual recognition. *arXiv preprint arXiv:2403.17695*.

Yang, H., Tang, S., Chen, M., Wang, Y., Zhu, F., Bai, L., Zhao, R. & Ouyang, W. (2022a). Domain Invariant Masked Autoencoders for Self-supervised Learning from Multi-domains. *ECCV*, pp. 151–168.

Yang, R., Ma, H., Wu, J., Tang, Y., Xiao, X., Zheng, M. & Li, X. (2022b). ScalableViT: Rethinking the context-oriented generalization of vision transformer. *ECCV*.

Yang, S., Wang, B., Shen, Y., Panda, R. & Kim, Y. (2023b). Gated linear attention transformers with hardware-efficient training. *arXiv preprint arXiv:2312.06635*.

Yazdanpanah, M., Bahri, A., Noori, M., Dastani, S., Vargas Hakim, G. A., Osowiechi, D., Ayed, I. B. & Desrosiers, C. (2025). Purge-Gate: Backpropagation-Free Test-Time Adaptation for Point Clouds Classification via Token Purging. *arXiv preprint arXiv:2509.09785*.

You, H., Xiong, Y., Dai, X., Wu, B., Zhang, P., Fan, H., Vajda, P. & Lin, Y. (2023). Castling-ViT: Compressing Self-Attention via Switching Towards Linear-Angular Attention During Vision Transformer Inference. *CVPR*.

Yu, F. & Koltun, V. (2015). Multi-scale context aggregation by dilated convolutions. *arXiv preprint arXiv:1511.07122*.

Yu, Y., Sheng, L., He, R. & Liang, J. (2024). STAMP: Outlier-Aware Test-Time Adaptation with Stable Memory Replay. *ECCV*, pp. 375–392.

Yuan, L., Chen, Y., Wang, T., Yu, W., Shi, Y., Jiang, Z.-H., Tay, F. E., Feng, J. & Yan, S. (2021). Tokens-to-token vit: Training vision transformers from scratch on imagenet. *ICCV*, pp. 558–567.

Yuan, L., Hou, Q., Jiang, Z., Feng, J. & Yan, S. (2022). Volo: Vision outlooker for visual recognition. *TPAMI*.

Yuan, L., Xie, B. & Li, S. (2023). Robust test-time adaptation in dynamic scenarios. *CVPR*, pp. 15922–15932.

Yun, S., Han, D., Oh, S. J., Chun, S., Choe, J. & Yoo, Y. (2019). Cutmix: Regularization strategy to train strong classifiers with localizable features. *ICCV*, pp. 6023–6032.

Zhang, H., Cisse, M., Dauphin, Y. N. & Lopez-Paz, D. (2017). mixup: Beyond empirical risk minimization. *arXiv preprint arXiv:1710.09412*.

Zhang, R., Isola, P. & Efros, A. A. (2016). Colorful image colorization. *ECCV*, pp. 649–666.

Zhang, X., Zhao, J. & LeCun, Y. (2015). Character-level convolutional networks for text classification. *NeurIPS*, 28, 649–657.

Zhang, X., Tian, Y., Xie, L., Huang, W., Dai, Q., Ye, Q. & Tian, Q. (2023). Hivit: A simpler and more efficient design of hierarchical vision transformer. *ICLR*.

Zhang, X., Chen, J., Yuan, J., Chen, Q., Wang, J., Wang, X., Han, S., Chen, X., Pi, J., Yao, K. et al. (2022). CAE v2: Context Autoencoder with CLIP Target. *arXiv preprint arXiv:2211.09799*.

Zhang, Y., Mehra, A., Niu, S. & Hamm, J. (2024). Dpcore: Dynamic prompt coreset for continual test-time adaptation. *arXiv preprint arXiv:2406.10737*.

Zhao, W., Wang, W. & Tian, Y. (2022). Graformer: Graph-oriented transformer for 3d pose estimation. *CVPR*, pp. 20438–20447.

Zheng, B., Ma, N., Tong, S. & Xie, S. (2025). Diffusion transformers with representation autoencoders. *arXiv preprint arXiv:2510.11690*.

Zhong, Z., Zheng, L., Kang, G., Li, S. & Yang, Y. (2020). Random erasing data augmentation. *AAAI*, 34(07), 13001–13008.

Zhou, B., Zhao, H., Puig, X., Fidler, S., Barriuso, A. & Torralba, A. (2017a). Scene parsing through ade20k dataset. *CVPR*, pp. 5122–5130.

Zhou, B., Zhao, H., Puig, X., Xiao, T., Fidler, S., Barriuso, A. & Torralba, A. (2019). Semantic understanding of scenes through the ade20k dataset. *IJCV*, 127, 302–321.

Zhou, D., Kang, B., Jin, X., Yang, L., Lian, X., Jiang, Z., Hou, Q. & Feng, J. (2021a). Deepvit: Towards deeper vision transformer. *arXiv preprint arXiv:2103.11886*.

Zhou, J., Wei, C., Wang, H., Shen, W., Xie, C., Yuille, A. & Kong, T. (2021b). ibot: Image bert pre-training with online tokenizer. *arXiv preprint arXiv:2111.07832*.

Zhou, K., Yang, Y., Qiao, Y. & Xiang, T. (2021c). Domain Generalization with MixStyle. *ICLR*.

Zhou, Y., Ye, Q., Qiu, Q. & Jiao, J. (2017b). Oriented response networks. *CVPR*, pp. 519–528.

Zhu, L., Wang, X., Ke, Z., Zhang, W. & Lau, R. (2023). BiFormer: Vision Transformer with Bi-Level Routing Attention. *CVPR*.

Zhu, L., Liao, B., Zhang, Q., Wang, X., Liu, W. & Wang, X. (2024). Vision mamba: Efficient visual representation learning with bidirectional state space model. *arXiv preprint arXiv:2401.09417*.

Zhu, X., Hu, H., Lin, S. & Dai, J. (2019). Deformable convnets v2: More deformable, better results. *CVPR*, pp. 9308–9316.

Zoph, B. & Le, Q. V. (2016). Neural architecture search with reinforcement learning. *arXiv preprint arXiv:1611.01578*.