LEARNING TO SEE THROUGH MNEMONIC RECALL

Anonymous authors
Paper under double-blind review

ABSTRACT

Modern visual backbones largely follow an operator-centric view of representation learning, where prior experience shapes the transformations applied to inputs. Yet experience can also contribute as persistent visual content that is selectively recalled to inform current perception. Inspired by memory-based reinstatement in human perception, we formulate visual representation learning as the joint contribution of dynamic computation and conditional recall. This work introduces REMEVIT, a family of visual Transformer backbones equipped with visual pattern memory that retains recurring local patterns as persistent and addressable states. Current visual cues can selectively evoke relevant patterns, while context-conditioned gating adapts their contribution to the current observation. The separation of stored content from contextual use also allows a shared source of visual experience across different backbone architectures. We instantiate REMEVIT on both DeiT and Swin, which yield consistent performance gains across classification and dense prediction tasks. Further results support visual pattern memory as a source of reusable perceptual priors that benefit new tasks, strengthen recognition under occlusion, and enhance representations learned without supervision.

1 INTRODUCTION

Modern visual backbones are built upon an *operator-centric view* for representation learning. Given a specific input, the convolutional filters map local regularities (He et al., 2016), self-attention aggregates evidence across positions (Dosovitskiy et al., 2021), and feed-forward networks recombine the feature channels (Vaswani et al., 2017). Stacked over depth, the operators form a scalable foundation to transform sensory evidences into increasingly abstract visual representations. In such architecture, operator parameters are shaped by prior experience, and determine how new inputs are processed.

Prior experience, however, can participate in representation formation in another computational form. To main perceptual stability, humans retain persistent representations of recurring regularities and events (what we refer to as ***memory***), that can be reinstated by relevant sensory cues (Wheeler et al., 2000) and, in turn, modulate the current perceptual processing (Hindy et al., 2016). This formulation is particularly natural in vision, where changes in viewpoint, illumination, scale, and occlusion can make the observations ambiguous or incomplete (Lowe, 2004). Yet local visual patterns (*e.g.*, textures, edge configurations, and part-like structures) recur across different images, rendering them natural candidates for reusable content (Olshausen & Field, 1996; Lowe, 2004). Reinstating such regularities could serve as a stabilizing prior that constrains and refines the interpretation of new sensory evidences (Kok et al., 2012; Hindy et al., 2016).

Viewed through this lens, the operator-centric design in modern visual backbones generally overlook memory-based reinstatement. Although learnable tokens and prototypes introduce persistent states, the former tend to supply generic working context (Darcet et al., 2024), while the latter focus on a few representative anchors for relating visual instances (Caron et al., 2020) or differentiating semantic groups (Zhou et al., 2022). In contrast, visual memory aims to preserve a broader range of local patterns for subsequent reuse, including both shared structures and their recurring variations across observations. Mainstream backbones still lack a persistent store of such pattern content that can be selectively evoked by the current input, and complement the interpretation of new visual evidence.

In this work, we reformulate visual representation learning as the joint contribution of dynamic computation and conditional recall. To instantiate this view, we introduce **REMEVIT**, a family of visual Transformer backbones equipped with persistent ***visual pattern memory***. As illustrated in

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

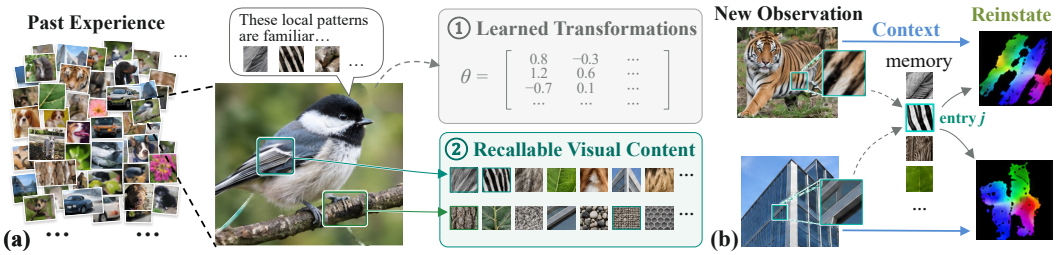

Figure 1: **Past experience as transformations and persistent memory.** (a) REMEVIT complements learned transformations with visual pattern memory. (b) Different observations can evoke the same memory entry j, while the current context determines how the recalled content contributes.

Fig. 1, in REMEVIT, prior experience is represented both as operators that process of the current input and visual states that externalizes reusable local pattern content. The visual pattern memory satisfies two requirements simultaneously: ❶ *content-addressable*, allowing a local visual cue to selectively retrieve only items relevant to the current observation from massive stored patterns; ❷ *context-conditioned*, allowing the same retrieved content to contribute varying responses depending on the current contexts. To achieve this, we associate each memory entry with a *retrieval code* for continuous addressing, a *memory descriptor* for post-retrieval contextual gating, and a *memory value* to carry the persistent pattern content. A lightweight 2D encoder first maps each patch neighborhood to a continuous query, and matched over learnable retrieval codes to aggregate the associated memory entries. The current input and memory descriptor then jointly produce a channel-wise gate to determine whether and how strongly the retrieved value should affect the representation. Finally, this gate modulates the memory value of retrieved entries to form a context-adapted response. This establishes a context-aware selection, which decouples memory retrieval from utilization.

A deeper insight of this context-conditioned recall is that memory need not be bound to context in which it is consumed, as its contribution is determined by the current representation at recall time. This motivates us to maintain all recurring patterns in a shared latent space, rather than a private bank tied to the representation semantics of each insertion layer (Sandler et al., 2022b; Wang et al., 2026). Each layer remains free to reinterpret the stored content according to its own needs via contextual gating between current features and retrieved descriptors. Such disentanglement also provides the basis for an ***architecture-agnostic*** memory formulation, where recalled pattern content is added to backbone features through residual connections, leaving the original model structure intact.

For a thorough evaluation, we instantiate REMEVIT in both Swin (Liu et al., 2021) and DeiT (Touvron et al., 2021a) . Our experiments span image classification on ImageNet-1K (Russakovsky et al., 2015), object detection and instance segmentation on MS COCO (Lin et al., 2014), and semantic segmentation on ADE20K (Zhou et al., 2017), where the memory-enhanced variants achieve 2.9%/3.9%/7.6% gains in top-1 accuracy, mAP, and mIoU, respectively. Beyond standard benchmarks, we further examine whether learned pattern memory can be reused across tasks without updating its contents, support recognition under occlusion, and complement self-supervised learning. We hope this work encourages a broader view of visual representation learning, in which persistent visual pattern memory could serve as an active foundation for robust perception.

2 RELATED WORK

Visual Representation Learning. Viewed through an operator-centric lens, a central trajectory in visual representation learning has been the design of increasingly expressive transformations to aggregate, propagate, and recombine visual evidence. Convolutional networks instantiate spatially local and weight-shared operators, which progresses from deep architectures (Krizhevsky et al., 2012) and residual learning (He et al., 2016) to modernized ConvNet designs (Liu et al., 2022). Vision Transformers employ self-attention for content-dependent interactions among image patches (Dosovitskiy et al., 2021; Touvron et al., 2021a). Subsequent variants reorganize these operators through hierarchical feature structures (Wang et al., 2021; Liu et al., 2021), locality-aware inductive biases (Wu et al., 2021; D'Ascoli et al., 2021), and multiscale or neighborhood attention to regulate the interaction ranges (Fan et al., 2021; Hassani et al., 2023). MLP-based token mixers (Tolstikhin et al., 2021) further explore alternative spatial-mixing operators. More recently, visual state-space

models realize long-range spatial propagation through bi- or multi-directional scan-based state transitions (Zhu et al., 2024; Liu et al., 2024). Despite the architectural differences, these approaches largely share an operator-bound organization of prior experience, where regularities acquired during training are encoded in the parameters. Standard backbone formulations do not explicitly maintain learned visual regularities as a separate store of reusable content. REMEVIT complements these operators with visual pattern memory, which retains recurring local patterns as persistent, addressable content that can be selectively recalled and recontextualized according to the current visual inputs.

Memory-Augmented Neural Networks. Early neural memory models introduce explicit memory structures to retain information beyond ordinary hidden-state computation. Memory Networks represent supporting information as memory slots, and retrieve relevant entries through query-dependent matching (Weston et al., 2014; Sukhbaatar et al., 2015). Key-Value Memory Networks further decouple the representations used for addressing from the contents returned by the memory (Miller et al., 2016). In parallel, Neural Turing Machines and Differentiable Neural Computers equipped neural controllers with differentiable read-write access to external memory (Graves et al., 2014; 2016). In vision, feature banks and temporal memories retain observed samples or episode histories (Wu et al., 2018a; He et al., 2020; Oh et al., 2019; Ryoo et al., 2023), whereas learnable slots, tokens, and prototypes provide persistent task context or representative anchors (Gong et al., 2019; Sandler et al., 2022a). More recent work incorporates memory directly into the representation pathway of visual backbones through shared memory matrices that serve as token-mixing operators (Guo et al., 2022), image-conditioned workspaces that are repeatedly read and updated (Jajal et al., 2025), or associative mechanisms that retrieve entries from prototype banks to refine token representations (Sun et al., 2025; Wang et al., 2026). REMEVIT adopt a different organization. Rather than retaining observed samples or layer-specific prototypes, it learns a persistent bank of recurring visual patterns. This bank is shared across representation depths through layer-specific interfaces and serves as complement to, rather than replacing, the original backbone operators.

Conditional and Sparse Retrieval. A common strategy for increasing model capacity is to make resource allocation input-dependent, such that only the components relevant to a given input are activated or retrieved. Along this direction, dynamic networks adapt the execution paths by selectively activating layers or branches (Wu et al., 2018b; Wang et al., 2018), whereas dynamic and conditionally parameterized convolutions generate or combine kernels according to the current input (De Brabandere et al., 2016; Yang et al., 2019; Chen et al., 2020). Mixture-of-Experts models further scale this principle through token-wise routing to a subset of expert transformations (Shazeer et al., 2017; Fedus et al., 2022; Riquelme et al., 2021). Collectively, these approaches condition which computation is performed. On the other hand, sparse key-value memories make access to stored content by retrieving a small number of vectors from large learned tables (Lample et al., 2019). Engram applies this to language modeling by using recurring n-grams to address a subset of persistent embeddings (Cheng et al., 2026). REMEVIT extends this content-selection perspective. For sparse recall, each patch neighborhood forms a continuous query and activates only the top-k entries in a persistent pattern bank. For conditional utilization, retrieval codes determine *what* is recalled, whereas the current representation and retrieved descriptors determine *how* the associated values contribute. This recall-recontextualization decomposition goes beyond a standard sparse key-value readout, in which the selected values constitute the memory response directly, and allows the same recalled patterns to produce different responses under varying visual contexts.

3 METHOD

Overview. We formulate visual representation learning as the joint contribution of input-dependent computation and conditional recall. For a standard visual backbone without an explicit memory pathway, given the current representation $\boldsymbol{X}_l$, the l-th block produces the next representation as:

$$\boldsymbol{X}_{l+1} = \mathcal{B}_{\theta_l}(\boldsymbol{X}_l), \tag{1}$$

where $\mathcal{B}_{\theta_l}$ denotes the transformation parameterized by θ_l. In this formulation, inputs reuse prior experience through shared transformations. As shown in Fig. 2, we additionally make the visual content available for reuse through a persistent and addressable visual pattern memory $\mathcal{M}$, which is learned jointly with the backbone. For a block equipped with memory access, the update becomes:

$$\begin{aligned} \boldsymbol{m}_l &= \mathcal{R}_l(\boldsymbol{X}_l; \mathcal{M}), \\ \boldsymbol{X}_{l+1} &= \mathcal{B}_{\theta_l}(\boldsymbol{X}_l + \mathcal{U}_l(\boldsymbol{X}_l, \boldsymbol{m}_l)). \end{aligned} \tag{2}$$

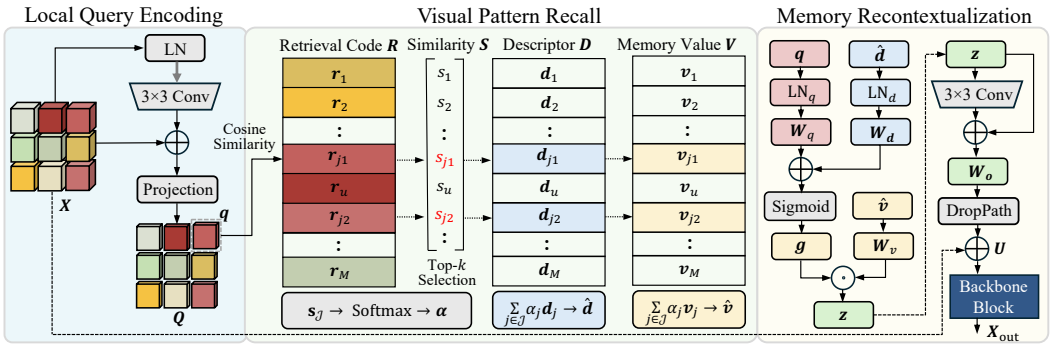

Figure 2: **Overview of REMEVIT.** The encoded local visual cues recall stored patterns, while context-conditioned gating adapts their contribution before residual integration into the backbone.

Here, $\mathcal{R}_l$ uses local visual cues to form a memory readout $\boldsymbol{m}_l$, while $\mathcal{U}_l$ transforms it into a residual response conditioned on $\boldsymbol{X}_l$. The memory response is added to $\boldsymbol{X}_l$ before the original backbone block is applied. When the memory response is zero, Eq. 2 reduces to the standard update in Eq. 1.

This formulation comprises three steps, *i.e.*, **Retain** preserves reusable pattern content in $\mathcal{M}$; **Recall** makes relevant content available through $\mathcal{R}_l$; and **Recontextualize** determines how that content contributes through $\mathcal{U}_l$. We then detail these steps in §3.1, while §3.2 describes how the persistent memory is shared across representation depths through layer-specific interfaces.

3.1 VISUAL PATTERN MEMORY

① **RETAIN.** We learn a persistent visual pattern memory $\mathcal{M}$ with M entries to accommodate recurring local visual structures and their variations across observations. To support both selective recall and context-dependent usage, each entry associates a memory value with two separately learned descriptors, *i.e.*, a retrieval code for locating the entry, and a memory descriptor for contextual modulation after retrieval. We organize these components into three parameter tables:

$$\mathcal{M} = \{\boldsymbol{R}, \boldsymbol{D}, \boldsymbol{V}\}, \qquad \boldsymbol{R}, \boldsymbol{D} \in \mathbb{R}^{M \times d}, \qquad \boldsymbol{V} \in \mathbb{R}^{M \times d_v} \tag{3}$$

where d is the dimension of retrieval codes and descriptors, and d_v is the value dimension. The j-th rows form one entry $(\boldsymbol{r}_j, \boldsymbol{d}_j, \boldsymbol{v}_j)$. The retrieval code $\boldsymbol{r}_j$ is matched against local visual queries to select relevant entries. The memory descriptor $\boldsymbol{d}_j$ is combined with the current input to establish a channel-wise gate. The value $\boldsymbol{v}_j$ stores the pattern content to form the gated memory response.

In standard key-value readout, stored values are aggregated using weights determined by query-key matching, and the weighted value readout are directly as the memory response (Miller et al., 2016; Lample et al., 2019). In contrast, Our formulation separates which patterns are recalled (matching $\boldsymbol{R}$ against $\boldsymbol{D}$) from how they contribute to the current representation (modulating $\boldsymbol{V}$ with $\boldsymbol{D}$). This allows the same memory readout to produce different responses under different visual contexts.

② **RECALL.** We recall stored patterns using local visual cues. To make memory access depend explicitly on local spatial structures, we construct each query from a patch feature and its spatial neighborhood. Let $\boldsymbol{X} \in \mathbb{R}^{N \times C}$ denote the patches at a memory access layer, arranged on an $H \times W$ grid with $N = HW$. We omit the layer indices for clarity. The local queries are computed as:

$$\begin{aligned} \boldsymbol{F} &= \mathrm{Reshape}_{H,W}(\mathrm{LayerNorm}_q(\boldsymbol{X})), \\ \boldsymbol{Q} &= \mathrm{Flatten}(\mathrm{PWConv}_{1\times1}(\boldsymbol{F} + \mathrm{DWConv}_{3\times3}(\boldsymbol{F}))), \end{aligned} \tag{4}$$

where $\boldsymbol{F} \in \mathbb{R}^{H \times W \times C}$ is the normalized feature map. The depthwise convolution (Howard et al., 2017) (DWConv) aggregates neighboring patches, with a residual connection to combine the outputs. The pointwise convolution (PWConv) then projects the combined features to the addressing dimension d. After flattening, $\boldsymbol{Q} \in \mathbb{R}^{N \times d}$ contains one local visual query $\boldsymbol{q}_i \in \mathbb{R}^d$ for each patch i.

Considering a local observation may resemble several stored entries, we form a weighted readout over top-k related entries. The queries are matched against the retrieval codes in a addressing space:

$$\boldsymbol{S} = \tau \widehat{\boldsymbol{Q}} \widehat{\boldsymbol{R}}^\top, \qquad \tau = \exp(\rho). \tag{5}$$

Here $\widehat{\boldsymbol{Q}}$ and $\widehat{\boldsymbol{R}}$ denote row-wise ℓ_2-normalized queries and retrieval codes. ρ is a learnable scalar to control the concentration of the retrieval weights. We then keep the k highest-scoring entries:

$$\mathcal{I}_i = \mathrm{TopK}_k(\boldsymbol{s}_i), \qquad \alpha_{ij} = \frac{\exp(s_{ij})}{\sum_{t\in\mathcal{I}_i}\exp(s_{it})}, \quad j \in \mathcal{I}_i, \tag{6}$$

where $\boldsymbol{s}_i \in \mathbb{R}^M$ is the i-th row of $\boldsymbol{S}$, $\mathcal{I}_i$ contains the selected memory indices, and α_{ij} is the normalized retrieval weight assigned to entry j. We then aggregate the descriptors and values:

$$\widetilde{\boldsymbol{d}}_i = \textstyle\sum_{j\in\mathcal{I}_i}\alpha_{ij}\boldsymbol{d}_j, \qquad \widetilde{\boldsymbol{v}}_i = \textstyle\sum_{j\in\mathcal{I}_i}\alpha_{ij}\boldsymbol{v}_j. \tag{7}$$

Although the addressing resembles prototype matching (Caron et al., 2020; Zhou et al., 2022), the **difference lies in the role of stored content**. Our visual pattern memory are optimized for context-dependent use, without summarizing those features as centroids. This shifts the emphasis from representative anchors to reusable pattern content. The bank is intended to accommodate both shared local structures and recurring variations that help interpret new observations. Specifically, a variation can be useful for recall even when it is not characteristic of an entire class or cluster.

③ **RECONTEXTUALIZE.** Recall identifies stored pattern content from local visual cues, but the retrieved content need not contribute in the same way to every occurrence. Rather than treating the readout as a ready-made feature response, we determine its contribution using the current patch state. Specifically, given the patch feature $\boldsymbol{x}_i \in \mathbb{R}^C$ and the aggregated descriptor $\widetilde{\boldsymbol{d}}_i$ and value $\widetilde{\boldsymbol{v}}_i$, we compute a channel-wise gate and the corresponding memory response as:

$$\begin{aligned} \boldsymbol{g}_i &= \sigma(\boldsymbol{W}_x\,\mathrm{LayerNorm}_x(\boldsymbol{x}_i) + \boldsymbol{W}_d\,\mathrm{LayerNorm}_d(\widetilde{\boldsymbol{d}}_i)),\\ \boldsymbol{z}_i &= \boldsymbol{g}_i \odot \boldsymbol{W}_v\widetilde{\boldsymbol{v}}_i, \end{aligned} \tag{8}$$

where $\boldsymbol{W}_x$, $\boldsymbol{W}_d$, and $\boldsymbol{W}_v$ are learnable projections to the value dimension d_v, σ denotes the sigmoid function, and $\odot$ denotes element-wise multiplication. The aggregated descriptors summarize the recalled content, while $\boldsymbol{x}_i$ supplies the context for its use. Together, the resulted gate $\boldsymbol{g}_i \in (0,1)^{d_v}$ determine the channel-wise contribution of the projected value, and produces $\boldsymbol{z}_i \in \mathbb{R}^{d_v}$. As such, different patch states can produce different responses even with the same mnemonic readout.

We then enrich the response by incorporating the recalled content from its spatial neighbors on the patch grid. Specifically, we stack the gated responses $\{\boldsymbol{z}_i\}_{i=1}^N$ as rows of $\boldsymbol{Z} \in \mathbb{R}^{N\times d_v}$, restore their two-dimensional spatial layout, and then compute:

$$\begin{aligned} \boldsymbol{Z}_{\text{2D}} &= \mathrm{Reshape}_{H,W}(\mathrm{Stack}_{i=1}^N(\boldsymbol{z}_i)),\\ \boldsymbol{Z}_{\text{ref}} &= \boldsymbol{Z}_{\text{2D}} + \mathrm{DWConv}_{3\times3}(\boldsymbol{Z}_{\text{2D}}),\\ \Delta\boldsymbol{X} &= \mathrm{Flatten}(\boldsymbol{Z}_{\text{ref}})\boldsymbol{W}_o^\top, \end{aligned} \tag{9}$$

where $\boldsymbol{Z}_{\text{2D}}, \boldsymbol{Z}_{\text{ref}} \in \mathbb{R}^{H\times W\times d_v}$. DWConv serves to combine neighboring memory responses within each value channel. $\boldsymbol{W}_o \in \mathbb{R}^{C\times d_v}$ then projects the refined responses to the backbone feature space, yielding $\Delta\boldsymbol{X} \in \mathbb{R}^{N\times C}$ which provides one memory response for each patch. Finally, we integrate the recontextualized memory response with the current input through a scaled residual connection:

$$\boldsymbol{U} = \mathrm{DropPath}(\boldsymbol{\gamma}\odot\Delta\boldsymbol{X}), \qquad \boldsymbol{X}_{\text{out}} = \mathcal{B}(\boldsymbol{X}+\boldsymbol{U}). \tag{10}$$

where $\boldsymbol{\gamma}\in\mathbb{R}^C$ is a learnable channel-wise scale (Touvron et al., 2021b) shared across inputs at this layer. DropPath (Huang et al., 2016) is applied to the memory residual during training as a branch regularizer. The resulting $\boldsymbol{U}\in\mathbb{R}^{N\times C}$ implements $\mathcal{U}_l(\boldsymbol{X}_l, \boldsymbol{m}_l)$ in Eq. 2 and is then added to $\boldsymbol{X}$.

3.2 CROSS-DEPTH MEMORY SHARING

The recontextualization mechanism in §3.1 separates stored pattern content from its contribution to the current representation. Building on this separation, we share visual pattern memory across network depths while keeping its recall and use layer-specific. Different layers can thus draw on a common pool of entries without being constrained to produce the same memory response.

Specifically, let $\mathcal{S}$ denote the set of memory access layers. As illustrated in Fig. 3, all layers in $\mathcal{S}$ access a single memory $\mathcal{M} = \{\boldsymbol{R}, \boldsymbol{D}, \boldsymbol{V}\}$, while retaining their own recall and recontextualization transformations. The memory dimensions d and d_v are fixed independently of the backbone width.

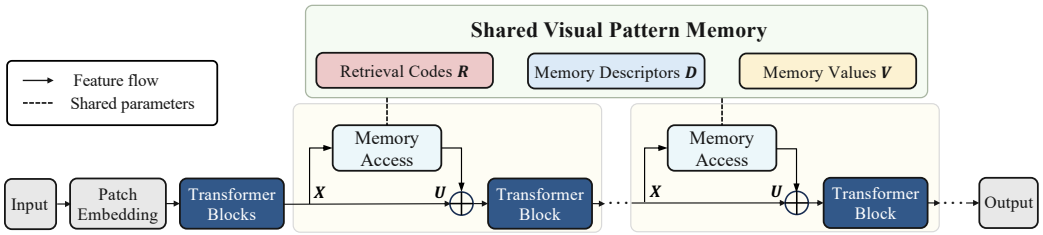

Figure 3: **Cross-depth memory sharing in REMEVIT.** A single visual pattern memory is shared across network depths. Layer-specific access interfaces recall and recontextualize stored patterns, producing context-dependent responses that are integrated into the backbone.

Layer-specific local encoders map patch neighborhoods into the shared addressing space, while contextual gating and output projections adapt the recalled content to the current layer representation and feature space. As such, rather than by allocating a separate memory bank to each layer, depth-specific differences are handled by these interfaces. Shared memory also couples the learning of memory entries across depths. For the weighted value readout in Eq. 7, the gradient of an entry $\boldsymbol{v}_j$ accumulates contributions from all layer–patch pairs that retrieve it:

$$\frac{\partial \mathcal{L}_{\text{task}}}{\partial \boldsymbol{v}_j} = \sum_{l \in \mathcal{S}} \sum_{i:\, j \in \mathcal{I}_i^l} \alpha_{ij}^l \frac{\partial \mathcal{L}_{\text{task}}}{\partial \widetilde{\boldsymbol{v}}_i^l}, \tag{11}$$

where $\mathcal{I}_i^l$ and α_{ij}^l denote the selected memory indices and retrieval weights for patch i at layer l, respectively. Each contribution is weighted by retrieval strength, while the readout gradient reflects the effect of the recalled value on the task through its context-dependent use. The retrieval codes and descriptors are likewise jointly optimized through their respective access paths.

3.3 IMPLEMENTATION DETAILS

Backbone instantiation. We instantiate REMEVIT in both constant-width and hierarchical visual Transformers. For constant-width ViTs, we insert memory after Transformer blocks and set the value dimension to $d_v = 192$. For hierarchical backbones, one memory module is placed after each of the four stages and use $d_v = 64$, with stage-specific output projections mapping the memory responses to the corresponding channels. All access layers share a single memory bank. The memory branch operates only on spatial patch tokens, while prefix tokens receive zero memory residual.

Memory configuration. By default, the memory bank contains 1,024 entries. The dimension of retrieval code and memory descriptor is set to 256. Top-4 entries is retrieved per patch. We initialize the channel-wise residual scale γ to 10^{-3} and cap the learnable similarity scale $\tau = \exp(\rho)$ at 100.

4 EXPERIMENTS

4.1 EXPERIMENTAL SETUP

Datasets. For standard vision tasks, we use ImageNet-1K (Russakovsky et al., 2015) for image classification, MS COCO (Lin et al., 2014) for object detection and instance segmentation, and ADE20K (Zhou et al., 2017) for semantic segmentation. To evaluate cross-task memory reuse, we select three datasets (*i.e.*, DTD (Cimpoi et al., 2014), Flowers102 (Nilsback & Zisserman, 2008), and EuroSAT (Helber et al., 2019)) from VTAB-1k (Zhai et al., 2019). Recognition from incomplete visual cues is evaluated on Occluded-PASCAL3D+ (Wang et al., 2020).

Evaluation Metric. We report Top-1 accuracy and throughput (img/s) on ImageNet-1K, bounding-box AP (AP^{box}) and mask AP (AP^{mask}) averaged over IoU thresholds from 0.50 to 0.95 on COCO, and mean intersection over union (mIoU) on ADE20K. Memory-reuse results are reported separately for each target dataset, while occlusion results include accuracy at each occlusion level. We additionally report model parameters and FLOPs to quantify computational overhead.

Table 1: Quantitative results for image classification with ConvNeXt (Liu et al., 2022), VMamba (Liu et al., 2024), HiViT (Zhang et al., 2023), XCiT (Ali et al., 2021), EAMLP (Guo et al., 2022), V-HMN (Wang et al., 2026), DeiT (Touvron et al., 2021a), Swin (Liu et al., 2021) on ImageNet-1K (Russakovsky et al., 2015).

Model	Params (M)	FLOPs (G)	Throughput	Top-1
Standard Image Classification Models				
ConvNeXt-T	28	4.5	1198	82.1
ConvNeXt-S	50	8.7	684	83.1
HiViT-T	19	4.6	1393	82.1
HiViT-S	38	9.1	712	83.5
XCiT-S24	48	9.2	671	82.6
XCiT-M24	84	16.2	423	82.7
VMamba-T	30	4.9	1686	82.6
VMamba-S	50	8.7	877	83.6
Explicit Memory Counterparts				
EAMLP-19	54.6	-	-	79.4
V-HMN	26.3	-	-	76.7
DeiT-T	5.7	1.3	4214	72.2
+ REMEVIT	5.8	1.3	3982	**75.1** ↑2.9
Swin-Min	11.8	1.6	3455	72.2
+ REMEVIT	12.3	1.7	3312	**74.9** ↑2.7
DeiT-S	22.1	4.2	1761	79.8
+ REMEVIT	22.3	4.6	1638	**81.8** ↑2.0
Swin-T	28.3	4.5	1244	81.2
+ REMEVIT	28.5	4.9	1154	**82.9** ↑1.7
Swin-S	50.2	8.7	718	82.9
+ REMEVIT	50.6	9.2	675	**83.8** ↑0.9

Table 2: Results for semantic segmentation with UPerNet (Xiao et al., 2018), MCIBI (Jin et al., 2021), ProtoSeg (Zhou et al., 2022) on ADE20K (Zhou et al., 2017).

Method	Backbone	Params(M)	FLOPs (G)	mIoU (SS)↑
MCIBI	ResNet-50	67	953	44.3
ProtoSeg	Swin-B	91	-	48.6
ProtoSeg	MiT-B4	64	-	51.7
UPerNet	Swin-Min	43.4	221	33.5
UPerNet + REMEVIT		44.2	222	**37.4** ↑3.9
UPerNet	DeiT-S	57.6	1217	43.8
UPerNet + REMEVIT		58.9	1221	**46.6** ↑2.8
UPerNet	Swin-T	60.2	945	44.5
UPerNet + REMEVIT		61.4	952	**47.0** ↑2.5
UPerNet	Swin-S	81.5	1039	47.6
UPerNet + REMEVIT		83.0	1051	**49.4** ↑1.8

Table 3: Object detection and instance segmentation on MS COCO with Mask R-CNN (He et al., 2017).

Backbone	Params (M)	FLOPs (G)	AP^{box} ↑	AP^{mask} ↑
ResNet-50	44.2	260	38.2	34.7
VMamba-T	50	271	47.3	42.7
VMamba-S	70	384	48.7	43.7
Swin-Min	31.3	212	31.4	30.3
+ REMEVIT	32.1	218	**37.9** ↑7.6	**36.0** ↑5.7
Swin-T	47.6	267	42.7	39.3
+ REMEVIT	48.8	276	**45.9** ↑3.2	**41.6** ↑2.3
Swin-S	69.3	354	44.8	40.9
+ REMEVIT	70.9	368	**47.3** ↑2.5	**42.7** ↑1.8

4.2 PERFORMANCE ON STANDARD VISION BENCHMARKS

Image Classification. Table 1 reports the classification results on ImageNet-1K. With DeiT-Small, the memory-augmented version achieves **81.8%** top-1 accuracy, compared with 79.8% for the original model, while introducing 0.2M additional parameters. For Swin-Small, with REMEVIT, the corresponding accuracies are **83.8%** and 82.9%, respectively. Across the evaluated model scales, visual pattern memory can consistently improve the performance. We additionally report FLOPs and throughput, where the memory pathway introduces negligible computational cost.

Semantic Segmentation. Table 2 summarizes the semantic segmentation results on ADE20K (Zhou et al., 2017). With the same segmentation decoder, Swin-Small equipped with visual pattern memory reaches **49.4%** mIoU, compared with 47.6% for its counterpart. The corresponding improvements for DeiT-Small is **2.8%** mIoU. These gains indicate that the benefits of visual pattern memory extend beyond image-level classification to pixel-level semantic prediction.

Object Detection & Instance Segmentation. Table 3 presents the results on MS COCO 2017 (Lin et al., 2014) using Mask R-CNN (He et al., 2017). Augmented with visual pattern memory, Swin-Small obtains **47.3** box AP and **42.7** mask AP, compared with 44.8 and 40.9 for the baseline. Similar trends can be observed on larger backbones. These comparisons assess the effectiveness of transferring memory augmented representations from image-level recognition to object-level prediction.

4.3 EVALUATING VISUAL PATTERN MEMORY

Beyond the standard comparisons, we examine visual pattern memory from three perspectives: i) whether learned memory remains useful for new tasks; ii) whether it benefits recognition from incomplete visual evidence; and iii) whether pattern memory can be learned without class label.

Cross-Task Memory Reuse. We investigate whether visual pattern memory learned on ImageNet-1K can be reused on DTD, Flowers102, and EuroSAT. We compare four configurations, which are no memory, randomly initialized memory trained on the target task, and ImageNet-initialized memory that is either frozen or fine-tuned. All variants use the same ImageNet-pretrained backbone. As shown in Table 4, fine-tuned transferred memory achieves the best performance. Notably, the frozen

Table 4: Quantitative results for cross task memory reuse under the VTAB-1k setup.

Configuration	Trainability	DTD	Flowers	EuroSAT	Mean↑
No Memory	-	70.21	92.96	96.63	86.60
Random	✓	70.74	93.43	96.80	**86.99** ↑0.39
Transferred	×	71.81	93.89	96.95	**87.55** ↑0.95
Transferred	✓	72.12	94.07	96.92	**87.70** ↑1.10

Table 5: Quantitative results for recognition under occlusion on Occluded-PASCAL3D+ `val`.

Method	L0	L1	L2	L3	Mean↑
Faster R-CNN	83.2	67.9	60.7	42.9	63.7
Faster R-CNN with reg.	85.4	73.1	64.8	47.4	67.7
Faster R-CNN with occ.	86.9	77.8	67.2	52.3	71.1
CA-CompNet	89.3	82.9	79.1	66.7	79.5
+ REMEVIT	90.1	84.2	80.4	68.3	**80.8** ↑1.3

Table 6: Quantitative results for self-supervised learning on ImageNet-1K `val`.

Framework	params (M)	FLOPs (G)	Top-1
LeJEPA-S	21.7	4.6	64.0
+ REMEVIT	21.9	4.7	**65.2** ↑1.2
DINO (ViT-S)	21.5	4.6	74.5
+ REMEVIT	21.7	4.7	**75.2** ↑0.7

Table 7: Analysis of different memory instantiations.

#	Memory Config.	Top-1↑	mIoU↑	AP^{box} ↑
1	Baseline	72.2	33.5	31.4
2	Matched Param.	72.2	33.5	31.4
3	Conv-only	72.3	33.7	31.8
4	KV Memory	73.2	34.8	34.2
5	$\mathcal{M} = \{\boldsymbol{R}, \boldsymbol{D}, \boldsymbol{V}\}$	**74.9**	**37.4**	**37.9**

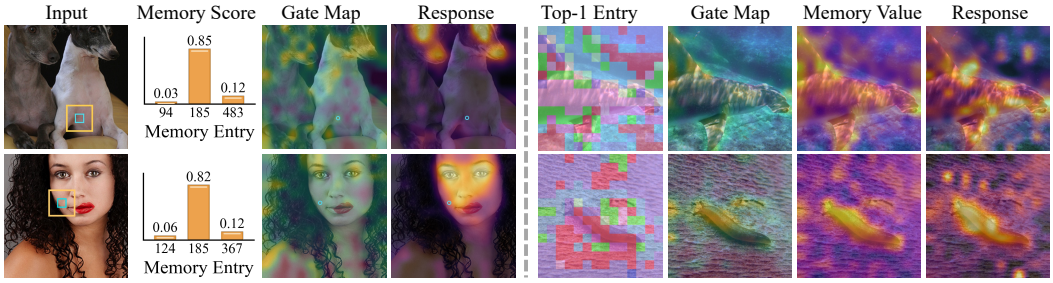

Figure 4: Left: patches sharing the same dominant memory entry yield different gates and responses. Right: spatial visualizations of top-1 entry assignments, gates, recalled values, and responses.

transferred bank outperforms a random bank learned on the target task. This suggests that patterns accumulated during pretraining remain useful beyond the task on which they were learned.

Recognition under Occlusion. We next examine whether REMEVIT supports recognition from incomplete visual evidence in Table 5, by integrating it into Faster R-CNN and evaluating on Occluded-PASCAL3D+ (Wang et al., 2020). The memory-augmented detector outperforms CA-CompNet (Wang et al., 2020) across different occlusion levels without test-time adaptation. The observed improvement is consistent with memory acting as a cue-conditioned prior, which allows partial observations to draw on learned regularities beyond the evidence directly visible in the input.

Self-Supervised Learning. We further investigate whether visual pattern memory can benefit representation learning without category supervision. After self-supervised training, we freeze each model and evaluate its learned representations on the ImageNet-1K validation set using the k-NN protocol. As shown in Table 6, incorporating memory consistently improves DINO (Caron et al., 2021) and LeJEPA (Balestriero & LeCun, 2025) over memory-free counterparts. This suggests that useful pattern content can be learned without category labels and selectively recalled to support current representations, which provides a complementary mechanism for self-supervised learning.

4.4 QUALITATIVE ANALYSIS

Fig. 4 illustrates context-dependent memory utilization. In the left panel, the highlighted patches favor the same memory entry (*i.e.*, 185), with retrieval weights of 0.85 and 0.82, yet exhibit different gates and responses. The right panel shows how gating modulates recalled values into responses emphasizing object regions. These examples illustrate that memory provides reusable pattern content rather than fixed feature responses, with its contribution adapted to the current visual context.

4.5 DIAGNOSTIC EXPERIMENTS

We examine the core designs through ablation studies on ImageNet-1K, ADE20K, and MS COCO, reporting top-1 accuracy, mIoU, and AP^{box}, respectively, with Swin-Min as the baseline by default.

Memory Instantiation. We first examine the memory instantiations in Table 7. The parameter-matched baseline (#2) does not improve over the original backbone (#1), while the convolution-only

Table 8: Analysis of the recontextualization mechanism that contributes to shared memory across blocks.

#	Memory	# Entry	Recontextualize	Top-1↑	mIoU↑
1	Block-wise	M	$\boldsymbol{g} = \sigma(\boldsymbol{W}_x\boldsymbol{x})$	73.4	35.1
2	Block-wise	$M \times B$	$\boldsymbol{g} = \sigma(\boldsymbol{W}_x\boldsymbol{x})$	73.7	35.5
3	Shared	M	$\boldsymbol{g} = \sigma(\boldsymbol{W}_x\boldsymbol{x})$	73.2	35.0
4	Shared	M	$\boldsymbol{g} = \sigma(\boldsymbol{W}_x\boldsymbol{x} + \boldsymbol{W}_d\widetilde{\boldsymbol{d}})$	**74.9**	**37.4**

Table 9: Analysis of context-aware gating in Eq. 8.

#	Config	Top-1↑	mIoU↑	AP^{box} ↑
1	No gate	72.8	34.5	33.5
2	$\boldsymbol{g} = \sigma(\boldsymbol{W}_g)$	73.6	35.4	35.1
3	$\boldsymbol{g} = \sigma(\boldsymbol{W}_x\boldsymbol{x})$	73.4	35.1	34.6
4	$\boldsymbol{g} = \sigma(\boldsymbol{W}_d\widetilde{\boldsymbol{d}})$	73.9	36.2	35.8
5	$\boldsymbol{g} = \sigma(\boldsymbol{W}_x\boldsymbol{x} + \boldsymbol{W}_d\widetilde{\boldsymbol{d}})$	**74.9**	**37.4**	**37.9**

Table 10: Analysis of neighbor feature aggregation.

DWConv (Eq. 4)	DWConv (Eq. 9)	Top-1↑	mIoU↑	AP^{box} ↑
		74.1	36.5	36.9
✓		74.6	37.0	37.4
	✓	74.7	37.1	37.5
✓	✓	**74.9**	**37.4**	**37.9**

Table 11: Analysis of the entry size M in memory.

# M	Top-1↑	mIoU↑	AP^{box} ↑
256	74.1	37.1	37.7
512	74.4	37.3	37.9
1024	**74.9**	**37.4**	**37.9**
2048	75.1	37.5	37.8
4096	75.5	37.4	37.8

variant (#3), which uses DWConv to generate the residual, yields only a modest gain. Introducing key–value memory without recontextualization (#4) provides a larger improvement, suggesting the benefit of explicitly recalling stored pattern content. Our full formulation, $\mathcal{M} = \{\boldsymbol{R}, \boldsymbol{D}, \boldsymbol{V}\}$ (#5), which separately parameterizes the retrieval codes, descriptors, and memory values for addressing, contextual modulation, and content storage, achieves the best performance. These results reveal that the improvements cannot be explained solely by increased parameter count or local convolution, and its advantage over the key–value counterpart supports separating memory retrieval from utilization.

Cross-Depth Memory Sharing. Table 8 examines the role of recontextualization in sharing memory across blocks. Simply sharing a memory bank with patch-only gating underperforms block-wise memory (#3 *v.s.* #1). This indicates that shared storage alone is insufficient to match block-specific banks. After incorporating memory descriptors into the gate, each block can determine how shared patterns contribute based on its current state. This leads to the best performance.

Context-Aware Gating. Table 9 examines the conditioning signals used in the channel-wise gate (Eq. 8). We compare five variants, which are no gating, an input-independent learnable gate, a patch-conditioned gate, a descriptor-conditioned gate, and our joint gate. The two single-input variants use only the current patch state (*i.e.*, $\boldsymbol{x}_i$) or the retrieved descriptor (*i.e.*, $\tilde{\boldsymbol{d}}_i$), respectively. Joint conditioning outperforms both single-input variants, revealing that the current patch state and the recalled descriptor provide complementary conditioning signals for memory utilization.

Spatial Aggregation. Table 10 examines the two depthwise convolutions used for local query construction (Eq. 4) and memory-response refinement (Eq. 9). The former incorporates neighboring patch features into memory addressing, while the latter combines neighboring memory responses before residual fusion. Compared with the baseline, enabling either one individually improves performance, indicating that spatial aggregation is beneficial at both stages. Combining the two yields the best results, which suggests that their contributions are complementary.

Memory Capacity. Table 11 examines the number of memory entries M. On ImageNet-1K, performance benefits from increasing entries and reaches its best at $M = 1024$, with no remarkable improvement from further expansion. In contrast, $M = 256$ is sufficient for the smaller-scale datasets like MS COCO and ADE20K. These results support the scalability of our design, that visual pattern memory can exploit additional storage capacity to support larger-scale visual learning.

5 Conclusion

In this work, we bring dynamic computation and conditional recall together in visual representation learning. Our visual pattern memory retains recurring local patterns as persistent content that can be selectively recalled to inform current perception. Separating stored content from its contextual use also formulates a shared memory bank to support different responses across network depths. Experiments demonstrate consistent improvements across stand visual tasks such as image classification and dense prediction. Beyond these gains, the learned memory can also benefit new tasks, improve recognition under occlusion, and enhance self-supervised representation learning. Together, these findings support persistent visual memory as a source of reusable perceptual priors, and highlight the contextual reuse of experience as a complement to the design of modern neural architectures.

6 AI USE STATEMENT

In this work, we used generative AI tools solely for language editing, including sentence-level polishing and grammar checking. Generative AI was not used to formulate the research questions, develop the method, or generate scientific claims. All AI-assisted revisions were manually reviewed to ensure that they preserved the intended meaning. We take full responsibility for the final content of this work, including all text, claims, and artifacts produced with the aid of generative AI.

7 REPRODUCIBILITY

Our method is implemented in PyTorch(Paszke et al., 2019), with image classification experiments built on `timm`(Wightman, 2019) and downstream experiments conducted using MMDetection and MMSegmentation. All experiments use four NVIDIA 4090 GPUs and BF16 mixed precision. Code will be released after acceptance. The training and testing details are provided in §A.1.

REFERENCES

Alaaeldin Ali, Hugo Touvron, Mathilde Caron, Piotr Bojanowski, Matthijs Douze, Armand Joulin, Ivan Laptev, Natalia Neverova, Gabriel Synnaeve, Jakob Verbeek, et al. Xcit: Cross-covariance image transformers. In *NeurIPS*, 2021.

Randall Balestriero and Yann LeCun. Lejepa: Provable and scalable self-supervised learning without the heuristics. *arXiv preprint arXiv:2511.08544*, 2025.

Sudong Cai, Shuai Yuan, Bingzhi Chen, Rui Mao, and Bing Wang. Selection-as-nonlinearity: bridging attention and activation via a joint game-decision lens for interpretable, discriminative visual representations. In *CVPR*, 2026.

Mathilde Caron, Ishan Misra, Julien Mairal, Priya Goyal, Piotr Bojanowski, and Armand Joulin. Unsupervised learning of visual features by contrasting cluster assignments. In *NeurIPS*, 2020.

Mathilde Caron, Hugo Touvron, Ishan Misra, Hervé Jégou, Julien Mairal, Piotr Bojanowski, and Armand Joulin. Emerging properties in self-supervised vision transformers. In *ICCV*, 2021.

Yinpeng Chen, Xiyang Dai, Mengchen Liu, Dongdong Chen, Lu Yuan, and Zicheng Liu. Dynamic convolution: Attention over convolution kernels. In *CVPR*, 2020.

Xin Cheng, Wangding Zeng, Damai Dai, Qinyu Chen, Bingxuan Wang, Zhenda Xie, Kezhao Huang, Xingkai Yu, Zhewen Hao, Han Zhang, et al. Conditional memory via scalable lookup: A new axis of sparsity for large language models. In *ACL*, 2026.

Mircea Cimpoi, Subhransu Maji, Iasonas Kokkinos, Sammy Mohamed, and Andrea Vedaldi. Describing textures in the wild. In *CVPR*, 2014.

Timothée Darcet, Maxime Oquab, Julien Mairal, and Piotr Bojanowski. Vision transformers need registers. In *ICLR*, 2024.

Stéphane D'Ascoli, Hugo Touvron, Matthew L. Leavitt, Ari S. Morcos, Giulio Biroli, and Levent Sagun. ConViT: Improving vision transformers with soft convolutional inductive biases. In *ICML*, 2021.

Bert De Brabandere, Xu Jia, Tinne Tuytelaars, and Luc Van Gool. Dynamic filter networks. In *NeurIPS*, 2016.

Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, Jakob Uszkoreit, and Neil Houlsby. An image is worth 16x16 words: Transformers for image recognition at scale. In *ICLR*, 2021.

Haoqi Fan, Bo Xiong, Karttikeya Mangalam, Yanghao Li, Zhicheng Yan, Jitendra Malik, and Christoph Feichtenhofer. Multiscale vision transformers. In *ICCV*, 2021.

William Fedus, Barret Zoph, and Noam Shazeer. Switch transformers: Scaling to trillion parameter models with simple and efficient sparsity. *Journal of Machine Learning Research*, 23(120):1–39, 2022.

Dong Gong, Lingqiao Liu, Vuong Le, Budhaditya Saha, Moussa Reda Mansour, Svetha Venkatesh, and Anton van den Hengel. Memorizing normality to detect anomaly: Memory-augmented deep autoencoder for unsupervised anomaly detection. In *ICCV*, 2019.

Alex Graves, Greg Wayne, and Ivo Danihelka. Neural turing machines. *arXiv preprint arXiv:1410.5401*, 2014.

Alex Graves, Greg Wayne, Malcolm Reynolds, Tim Harley, Ivo Danihelka, Agnieszka Grabska-Barwińska, Sergio Gómez Colmenarejo, Edward Grefenstette, Tiago Ramalho, John Agapiou, Adrià Puigdomènech Badia, Karl Moritz Hermann, Yori Zwols, Georg Ostrovski, Adam Cain, Helen King, Christopher Summerfield, Phil Blunsom, Koray Kavukcuoglu, and Demis Hassabis. Hybrid computing using a neural network with dynamic external memory. *Nature*, 538(7626): 471–476, 2016.

Meng-Hao Guo, Zheng-Ning Liu, Tai-Jiang Mu, and Shi-Min Hu. Beyond self-attention: External attention using two linear layers for visual tasks. *IEEE TPAMI*, 45(5):5436–5447, 2022.

Ali Hassani, Steven Walton, Jiachen Li, Shen Li, and Humphrey Shi. Neighborhood attention transformer. In *CVPR*, 2023.

Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *CVPR*, 2016.

Kaiming He, Georgia Gkioxari, Piotr Dollár, and Ross Girshick. Mask R-CNN. In *ICCV*, 2017.

Kaiming He, Haoqi Fan, Yuxin Wu, Saining Xie, and Ross Girshick. Momentum contrast for unsupervised visual representation learning. In *CVPR*, 2020.

Patrick Helber, Benjamin Bischke, Andreas Dengel, and Damian Borth. Eurosat: A novel dataset and deep learning benchmark for land use and land cover classification. *IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing*, 12(7):2217–2226, 2019.

Nicholas C. Hindy, Felicia Y. Ng, and Nicholas B. Turk-Browne. Linking pattern completion in the hippocampus to predictive coding in visual cortex. *Nature Neuroscience*, 19(5):665–667, 2016.

Andrew G Howard, Menglong Zhu, Bo Chen, Dmitry Kalenichenko, Weijun Wang, Tobias Weyand, Marco Andreetto, and Hartwig Adam. Mobilenets: Efficient convolutional neural networks for mobile vision applications. *arXiv preprint arXiv:1704.04861*, 2017.

Gao Huang, Yu Sun, Zhuang Liu, Daniel Sedra, and Kilian Q Weinberger. Deep networks with stochastic depth. In *ECCV*, 2016.

Purvish Jajal, Nick Eliopoulos, Benjamin Shiue-Hal Chou, George K. Thiruvathukal, James C. Davis, and Yung-Hsiang Lu. Token turing machines are efficient vision models. In *WACV*, 2025.

Zhenchao Jin, Tao Gong, Dongdong Yu, Qi Chu, Jian Wang, Changhu Wang, and Jie Shao. Mining contextual information beyond image for semantic segmentation. In *ICCV*, 2021.

Peter Kok, Janneke F. M. Jehee, and Floris P. de Lange. Less is more: Expectation sharpens representations in the primary visual cortex. *Neuron*, 75(2):265–270, 2012.

Alex Krizhevsky, Ilya Sutskever, and Geoffrey E. Hinton. ImageNet classification with deep convolutional neural networks. In *NeurIPS*, 2012.

Guillaume Lample, Alexandre Sablayrolles, Marc'Aurelio Ranzato, Ludovic Denoyer, and Hervé Jégou. Large memory layers with product keys. In *NeurIPS*, 2019.

Tsung-Yi Lin, Michael Maire, Serge Belongie, James Hays, Pietro Perona, Deva Ramanan, Piotr Dollár, and C. Lawrence Zitnick. Microsoft COCO: Common objects in context. In *ECCV*, 2014.

Yue Liu, Yunjie Tian, Yuzhong Zhao, Hongtian Yu, Lingxi Xie, Yaowei Wang, Qixiang Ye, Jianbin Jiao, and Yunfan Liu. VMamba: Visual state space model. In *NeurIPS*, 2024.

Ze Liu, Yutong Lin, Yue Cao, Han Hu, Yixuan Wei, Zheng Zhang, Stephen Lin, and Baining Guo. Swin transformer: Hierarchical vision transformer using shifted windows. In *ICCV*, 2021.

Zhuang Liu, Hanzi Mao, Chao-Yuan Wu, Christoph Feichtenhofer, Trevor Darrell, and Saining Xie. A convnet for the 2020s. In *CVPR*, 2022.

David G. Lowe. Distinctive image features from scale-invariant keypoints. *IJCV*, 60(2):91–110, 2004.

Alexander Miller, Adam Fisch, Jesse Dodge, Amir-Hossein Karimi, Antoine Bordes, and Jason Weston. Key-value memory networks for directly reading documents. In *EMNLP*, 2016.

Maria-Elena Nilsback and Andrew Zisserman. Automated flower classification over a large number of classes. In *Indian conference on computer vision, graphics & image processing*, 2008.

Seoung Wug Oh, Joon-Young Lee, Ning Xu, and Seon Joo Kim. Video object segmentation using space-time memory networks. In *ICCV*, 2019.

Bruno A. Olshausen and David J. Field. Emergence of simple-cell receptive field properties by learning a sparse code for natural images. *Nature*, 381(6583):607–609, 1996.

Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, et al. Pytorch: An imperative style, high-performance deep learning library. In *NeurIPS*, 2019.

Carlos Riquelme, Joan Puigcerver, Basil Mustafa, Maxim Neumann, Rodolphe Jenatton, André Susano Pinto, Daniel Keysers, and Neil Houlsby. Scaling vision with sparse mixture of experts. In *NeurIPS*, 2021.

Olga Russakovsky, Jia Deng, Hao Su, Jonathan Krause, Sanjeev Satheesh, Sean Ma, Zhiheng Huang, Andrej Karpathy, Aditya Khosla, Michael Bernstein, Alexander C. Berg, and Li Fei-Fei. Imagenet large scale visual recognition challenge. *International Journal of Computer Vision*, 115(3):211–252, 2015.

Michael S. Ryoo, Keerthana Gopalakrishnan, Kumara Kahatapitiya, Ted Xiao, Kanishka Rao, Austin Stone, Yao Lu, Julian Ibarz, and Anurag Arnab. Token turing machines. In *CVPR*, 2023.

Mark Sandler, Andrey Zhmoginov, Max Vladymyrov, and Andrew Jackson. Fine-tuning image transformers using learnable memory. In *CVPR*, 2022a.

Mark Sandler, Andrey Zhmoginov, Max Vladymyrov, and Andrew Jackson. Fine-tuning image transformers using learnable memory. In *CVPR*, 2022b.

Noam Shazeer, Azalia Mirhoseini, Krzysztof Maziarz, Andy Davis, Quoc V. Le, Geoffrey E. Hinton, and Jeff Dean. Outrageously large neural networks: The sparsely-gated mixture-of-experts layer. In *ICLR*, 2017.

Sainbayar Sukhbaatar, Arthur Szlam, Jason Weston, and Rob Fergus. End-to-end memory networks. In *NeurIPS*, 2015.

Yuwei Sun, Hideya Ochiai, Zhirong Wu, Stephen Lin, and Ryota Kanai. Associative transformer. In *CVPR*, 2025.

Ilya O Tolstikhin, Neil Houlsby, Alexander Kolesnikov, Lucas Beyer, Xiaohua Zhai, Thomas Unterthiner, Jessica Yung, Andreas Steiner, Daniel Keysers, Jakob Uszkoreit, et al. Mlp-mixer: An all-mlp architecture for vision. In *NeurIPS*, 2021.

Hugo Touvron, Matthieu Cord, Matthijs Douze, Francisco Massa, Alexandre Sablayrolles, and Hervé Jégou. Training data-efficient image transformers & distillation through attention. In *ICML*, 2021a.

Hugo Touvron, Matthieu Cord, Alexandre Sablayrolles, Gabriel Synnaeve, and Hervé Jégou. Going deeper with image transformers. In *ICCV*, 2021b.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. In *NeurIPS*, 2017.

Angtian Wang, Yihong Sun, Adam Kortylewski, and Alan L Yuille. Robust object detection under occlusion with context-aware compositionalnets. In *CVPR*, 2020.

Jianfeng Wang, Amine M'Charrak, Luk Koska, Xiangtao Wang, Daniel Petriceanu, Ruizhi Wang, Michael Bumbar, Luca Pinchetti, and Thomas Lukasiewicz. Vision hopfield memory networks for image recognition. *arXiv preprint arXiv:2603.25157*, 2026.

Wenhai Wang, Enze Xie, Xiang Li, Deng-Ping Fan, Kaitao Song, Ding Liang, Tong Lu, Ping Luo, and Ling Shao. Pyramid vision transformer: A versatile backbone for dense prediction without convolutions. In *ICCV*, 2021.

Xin Wang, Fisher Yu, Zi-Yi Dou, Trevor Darrell, and Joseph E Gonzalez. Skipnet: Learning dynamic routing in convolutional networks. In *ECCV*, 2018.

Jason Weston, Sumit Chopra, and Antoine Bordes. Memory networks. *arXiv preprint arXiv:1410.3916*, 2014.

Mark E. Wheeler, Steven E. Petersen, and Randy L. Buckner. Memory's echo: Vivid remembering reactivates sensory-specific cortex. *Proceedings of the National Academy of Sciences*, 97(20): 11125–11129, 2000.

Ross Wightman. PyTorch Image Models. `https://github.com/rwightman/pytorch-image-models`, 2019.

Haiping Wu, Bin Xiao, Noel Codella, Mengchen Liu, Xiyang Dai, Lu Yuan, and Lei Zhang. CvT: Introducing convolutions to vision transformers. In *ICCV*, 2021.

Zhirong Wu, Yuanjun Xiong, Stella X. Yu, and Dahua Lin. Unsupervised feature learning via non-parametric instance discrimination. In *CVPR*, 2018a.

Zuxuan Wu, Tushar Nagarajan, Abhishek Kumar, Steven Rennie, Larry S Davis, Kristen Grauman, and Rogerio Feris. Blockdrop: Dynamic inference paths in residual networks. In *CVPR*, 2018b.

Tete Xiao, Yingcheng Liu, Bolei Zhou, Yuning Jiang, and Jian Sun. Unified perceptual parsing for scene understanding. In *ECCV*, 2018.

Brandon Yang, Gabriel Bender, Quoc V Le, and Jiquan Ngiam. Condconv: Conditionally parameterized convolutions for efficient inference. In *NeurIPS*, 2019.

Xiaohua Zhai, Joan Puigcerver, Alexander Kolesnikov, Pierre Ruyssen, Carlos Riquelme, Mario Lucic, Josip Djolonga, Andre Susano Pinto, Maxim Neumann, Alexey Dosovitskiy, et al. A large-scale study of representation learning with the visual task adaptation benchmark. *arXiv preprint arXiv:1910.04867*, 2019.

Xiaosong Zhang, Yunjie Tian, Lingxi Xie, Wei Huang, Qi Dai, Qixiang Ye, and Qi Tian. Hivit: A simpler and more efficient design of hierarchical vision transformer. In *ICLR*, 2023.

Bolei Zhou, Hang Zhao, Xavier Puig, Sanja Fidler, Adela Barriuso, and Antonio Torralba. Scene parsing through ADE20K dataset. In *CVPR*, 2017.

Tianfei Zhou, Wenguan Wang, Ender Konukoglu, and Luc Van Gool. Rethinking semantic segmentation: A prototype view. In *CVPR*, 2022.

Lianghui Zhu, Bencheng Liao, Qian Zhang, Xinlong Wang, Wenyu Liu, and Xinggang Wang. Vision mamba: Efficient visual representation learning with bidirectional state space model. In *ICML*, 2024.

A APPENDIX

A.1 IMPLEMENTATION DETAILS

Training. For ImageNet-1K, models are trained from scratch at a resolution of 224×224 for 300 epochs. We use AdamW with an initial learning rate of 2×10^{-3}, a weight decay of 0.05, and an effective batch size of 2,048. The learning rate is warmed up for 20 epochs and then decayed using a cosine schedule. The training pipeline includes repeated augmentation, Mixup, CutMix, RandAugment, random erasing, and label smoothing. Memory parameters use twice the base learning rate, while the layer-specific access parameters use the base learning rate. Weight decay is disabled for the memory and access components. For downstream tasks, backbones are initialized from the ImageNet-1K checkpoint. Analyses on MS COCO use Mask R-CNN with FPN (He et al., 2017) and the $1\times$ schedule for 12 epochs. We use AdamW with a learning rate of 10^{-4}, a weight decay of 0.05, and a global batch size of 16. Experiments on ADE20K use UPerNet (Xiao et al., 2018) with 512×512 crops and a 160K-iteration schedule. The optimizer is AdamW with a learning rate of 6×10^{-5}, a weight decay of 0.01, and a global batch size of 16. For cross-task memory reuse, all variants are initialized from the same ImageNet-1K pretrained backbone. In the transferred-memory setting, the memory tables remain frozen. The random-memory setting uses identically sized tables initialized at random and optimized on the target task. Both variants follow the same initialization procedure and adaptation budget. For self-supervised learning, we train models from scratch on ImageNet-1K following the adopted self-supervised pipeline. The backbone, memory tables, and recall and recontextualization interfaces are learned jointly under the self-supervised objective.

Testing. We use single-crop evaluation on ImageNet-1K and single-scale inference on COCO and ADE20K. The three selected transfer datasets follow the VTAB-1k splits. Occlusion experiments evaluate the same trained model on clean inputs and at different occlusion levels, without test-time adaptation. Memory contents remain fixed during inference.

A.2 ABLATION CONFIGURATION DETAILS

We clarify the configurations used in Tables 7-9. Unless otherwise specified, All these ablation experiments adopt Swin-Min(Cai et al., 2026) as the backbone.

Memory instantiation. Table 7 compares five configurations. Baseline denotes the original backbone without a memory pathway. Matched Param serves as a parameter-count control without memory retrieval. Conv-only generates an additional residual through DWConv rather than recalling content from a persistent memory bank. KV Memory reads stored values without recontextualization. The full configuration maintains $\mathcal{M} = \{\boldsymbol{R}, \boldsymbol{D}, \boldsymbol{V}\}$, with retrieval codes for addressing, descriptors for contextual gating, and values for storing pattern content.

Cross-depth memory sharing. For Table 8, let $B = |\mathcal{S}|$ denote the number of memory access layers. The "# Entry" column specifies the number of entries per bank. Block-wise memory assigns an independent bank to each access layer. Using M/B entries per bank gives M independently stored entries in total, whereas using M entries per bank gives $B{\times}M$ entries. Shared memory instead provides all access layers with the same bank of M entries. The recall and recontextualization interfaces remain layer-specific. These entry counts describe the storage allocation rather than the complete model parameter count. The first three configurations use patch-conditioned gating, $\boldsymbol{g} = \sigma(\boldsymbol{W}_x\boldsymbol{x})$. The final configuration additionally uses the retrieved descriptor, $\boldsymbol{g} = \sigma(\boldsymbol{W}_x\boldsymbol{x} + \boldsymbol{W}_d\widetilde{\boldsymbol{d}})$. Thus, the last two rows compare patch-only and joint conditioning within shared memory.

Context-aware gating. Table 9 varies the conditioning signals used to modulate the projected memory value in Eq. 8. No gate removes multiplicative modulation, which is equivalent to setting $\boldsymbol{g} = \mathbf{1}$. The input-independent variant uses a learnable channel-wise gate, $\boldsymbol{g} = \sigma(\boldsymbol{W}_g)$, that depends on neither the current patch nor the recalled descriptor. The patch-conditioned and descriptor-conditioned variants use $\boldsymbol{g} = \sigma(\boldsymbol{W}_x\boldsymbol{x})$ and $\boldsymbol{g} = \sigma(\boldsymbol{W}_d\widetilde{\boldsymbol{d}})$, respectively. The joint variant combines both signals. Although descriptor-only gating excludes direct conditioning on the current patch state, it remains input-dependent since $\widetilde{\boldsymbol{d}}$ is obtained through conditional recall. Following the shorthand in Tables 8 and 9, these expressions omit the normalization operators in Eq. 8.

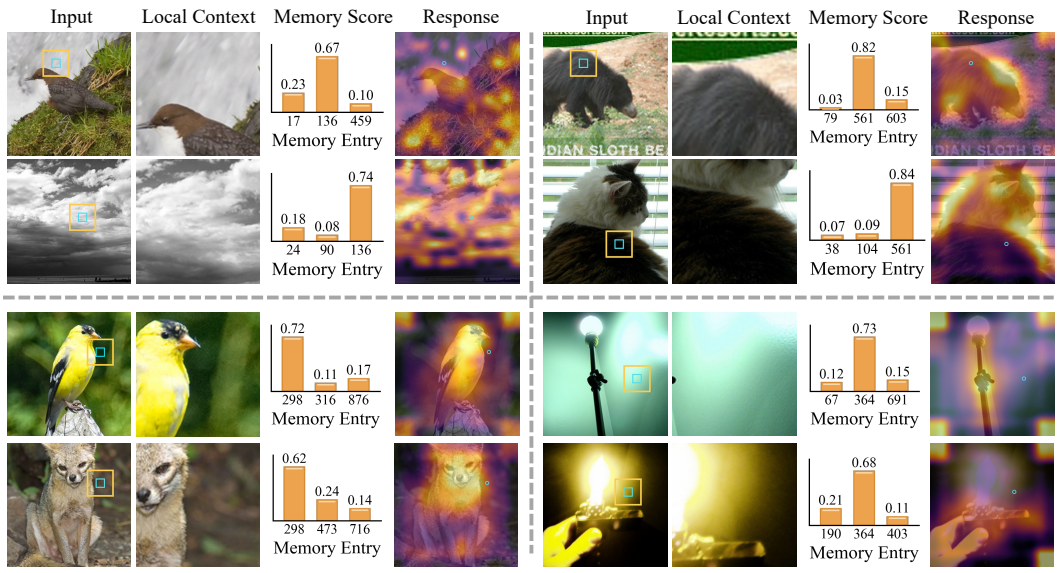

Figure 5: **Cross-image recall and context-dependent memory responses.** Each pair shows patches from different images that share the same dominant memory entry. Despite favoring the same entry, paired examples exhibit distinct response patterns in their respective visual contexts.

A.3 Limitations

While ReMeViT enables the conditional reuse of learned visual patterns, the current formulation has two limitations concerning online memory updating and retrieval scalability. **First**, the memory tables in ReMeViT are learned during training and remain fixed at inference. Recall and recontextualization select and adapt existing content, but do not write new entries or update stored patterns from test inputs. **Second**, although top-k selection restricts the entries used in the readout, the current implementation computes similarity scores against all M retrieval codes before selection. For N patch queries of dimension d, this matching step requires $O(N \times M \times d)$ computation. The matching cost increases with both memory capacity and the number of patch queries. These limitations motivate two directions for future work, *i.e.*, online memory updates to incorporate newly observed patterns, and more efficient retrieval mechanisms to support larger-scale addressing.

A.4 Visualization for Cross-Image Memory Recontextualization

Fig. 5 illustrates how visual pattern memory is recalled and utilized across different observations. The four pairs share dominant memory entries 136, 561, 298, and 364, respectively, despite differences in image content. For example, the highlighted patches in the bear and cat images favor entry 561 with weights of 0.82 and 0.84, while regions near the lamp and flame favor entry 364 with weights of 0.73 and 0.68. These examples suggest that memory entries capture recurring local regularities shared across different objects and scenes. Meanwhile, the response maps exhibit distinct spatial patterns across paired inputs. This behavior is consistent with our recall–recontextualization formulation, in which the current patch state and retrieved descriptors jointly modulate recalled values, adapting reusable pattern content to the current observation.

A.5 Visualization for Visual Pattern Memory

In Fig. 6, we further visualize the spatial behavior of memory recall and recontextualization. The top-1 entry maps exhibit relatively coarse assignments that do not necessarily align with object boundaries, while the recalled value maps often contain diffuse activations across both objects and surrounding regions. After recontextualization, the final responses become more spatially selective, emphasizing structures such as animal faces, the turtle shell, and the segmented body of the centipede, with less prominent responses in the surrounding background. These observations illustrate the distinction between recalling pattern content and determining its contribution: retrieval makes stored values available, while the current patch state and retrieved descriptors jointly condition their

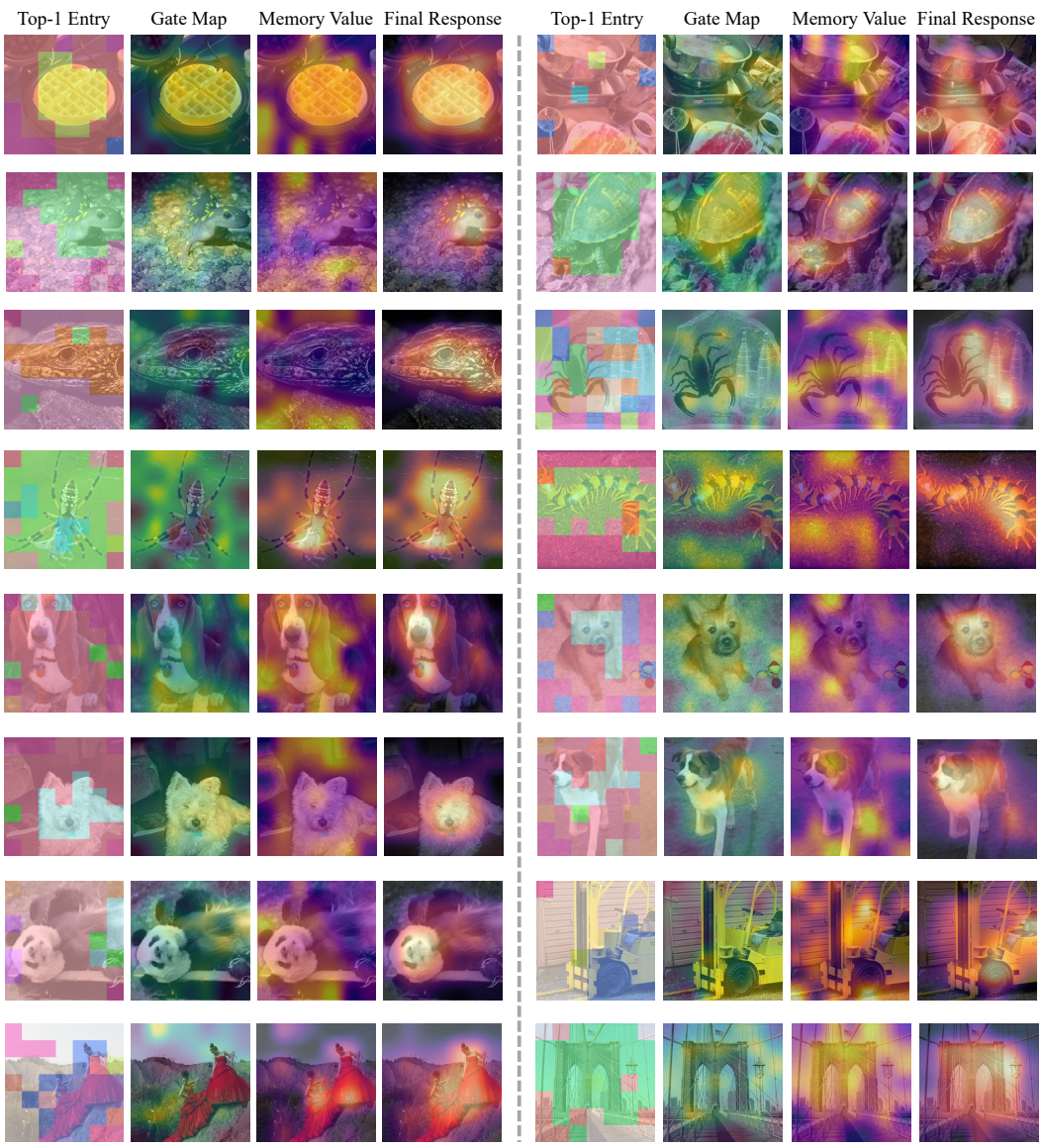

Figure 6: **Spatial visualization of memory recall and recontextualization.** Different colors in the entry maps denote different memory entries. Compared with the recalled value maps, the final responses tend to concentrate on object regions and local structures, illustrating how recontextualization adapts retrieved pattern content to the current observation.

utilization. Rather than treating recalled values as ready-made feature responses, recontextualization adapts their contribution to the current visual context.