Mixture-of-Top-k Attention: Efficient Attention via Scalable Fast Weights

Qishuai Wen, Zhiyuan Huang, Xianghan Meng, Wei He, and Chun-Guang Li
School of Artificial Intelligence, Beijing University of Posts and Telecommunications, Beijing 100876, P.R. China
{wqs, huangzhiyuan, mengxianhan, wei.he, lichunguang}@bupt.edu.cn

Abstract

The vanilla self-attention in Transformers can be viewed as a two-layer fast-weight MLP, where the weights are dynamically induced by inputs and the hidden dimension is equal to the sequence length N. As the context length grows, the expressive capacity of such an N-width MLP increases, but it becomes unscalable for extremely long sequences. Recently, this fast-weight perspective has motivated the Mixture-of-Experts (MoE) attention mechanism, which partitions the sequence into rigid blocks, treats them as fast-weight experts, and sparsely routes the tokens to these experts. In this paper, we elevate this perspective into a unifying framework for efficient attention, interpreting existing methods as making fast weights scalable via routing and/or compression, and organizing them into a five-dimensional taxonomy. Then, we propose **Mi**xture-of-**T**op-k **A**ttention (**MiTA**), which uses a small set of landmark queries to collect top-k attended key-value pairs as query-aware and deformable routed experts, while compressing the N-width MLP into a narrower shared expert. Consequently, MiTA improves the flexibility of prior MoE attention (from rigid to deformable fast-weight experts) and the scalability of prior top-k attention (from query-specific sets to reusable top-k sets). Extensive experiments on vision tasks demonstrate that MiTA achieves superior effectiveness and efficiency, and reveal intriguing properties such as an emergent token-pruning effect and strong generalization from standard attention. Code is available at https://github.com/QishuaiWen/MiTA.

1 Introduction

Attention is the core operation of Transformers, which underpins today's success and wide application of deep learning. Intuitively, attention learns to store the context as key-value associations and retrieve this short-term memory via queries [1, 2]. However, such an all-to-all lookup paradigm incurs quadratic computation and memory complexity in the sequence length, thereby hindering its scaling to long sequences. To this end, a plethora of efficient attention methods have been explored, albeit still lacking a unifying principle [3].

In recent years, several lines of work have converged on a promising viewpoint: the parameters of a two-layer MLP can be viewed as key-value pairs [4], and the key-value pairs in full attention can be viewed as the fast (i.e., input-dependent) weights of an MLP [5–7]. Therefore, pursuing efficient attention can be framed as a *fast-weight scaling* problem and draw inspiration from slow-weight (i.e., parameter) scaling approaches, such as weight tying [8], model pruning [9], and conditional computation [10].

Recent work along this direction has demonstrated that sparse attention inspired by Mixture-of-Experts (MoE) can scale up the effective sequence length, deliver wall-clock speedups, and accommodate large-scale pretraining [11, 12]. In contrast to traditional MoE, whose experts are static and

40th Conference on Neural Information Processing Systems (NeurIPS 2026).

Table 1: A five-dimensional taxonomy for efficient attention methods from the fast-weight scaling perspective. We present expert type and expert count together in the table (due to space limitation). A detailed analysis of this taxonomy and how MiTA fits into it can be found in Appendix B.

Method	Scaling strategy	Expert type and count	Expert construction	Routing topology
Linear Attention [14] MHLA [15]	Compression	One shared linear layer m linear layers	Kernelization Local prior & kernelization	All-to-one m times $\frac{N}{m}$-to-one
Linformer [16] PVT [17]	Compression	One shared MLP	Learnable projection	All-to-one
Agent Attention [18]	Compression	One shared MLP	Landmark probing	All-to-one
TTT [19] ViT^3 [6]	Compression	One shared module	Test-time training	All-to-one
Deformable DETR [20] DAT [21]	Routing	N MLPs One shared MLP	Offset prediction	N times one-to-one N-to-one
BRA [22] MoBA [11] NSA [12]	Routing Routing Compression & Routing	$\frac{N}{n}$ MLPs	Locality prior	$\frac{N}{n}$-to-$\frac{N}{n}$ N-to-$\frac{N}{n}$
Spark Attention [23] DSA [24]	Routing	N MLPs	Lightweight skim	N times one-to-one
MiTA (ours)	Compression & Routing	m MLPs	Landmark probing	N-to-m

structured model parameters (i.e., slow weights), their experts are input-dependent fast weights [13], more precisely, subsets of key-value pairs. By routing queries to the fast-weight experts sparsely, MoE attention reduces the complexity from quadratic to linear in sequence length N.

Roughly speaking, the central challenge in scaling the fast weights via MoE is to construct fast-weight experts from unstructured key-value pairs. For example, given the spatial and temporal locality priors in many modalities, a straightforward, hardware-friendly, albeit naive, approach is to partition the sequence into contiguous, non-overlapping, fixed-size blocks, and regard these blocks as fast-weight experts (i.e., a mixture of blocks or chunks) [11, 12, 25, 26]. After that, routing vectors are obtained by aggregating each block into a single vector, e.g., via average pooling or parameterized modules.

Since splitting the N-width fast-weight MLP into a mixture of such rigid blocks is coarse and suboptimal, subsequent work [27] has sought to improve upon the splitting scheme. Notably, we argue that top-k attention [23, 24] can be interpreted as spawning N k-width sub-MLPs from the N-width MLP, via gathering top-k key-value pairs attended by each query. Therefore, the rigid (fixed-shape) experts in prior MoE attention methods are replaced by top-k attention with top-k key-value pairs. However, instead of the width of the fast-weight MLP in full attention, it is the number of such sets of top-k key-value pairs that always equals N in their top-k attention, which still limits its scalability.

On the other hand, unlike the methods mentioned above that scale fast weights by accessing a subset of them, linear attention [14, 28, 29] and test-time training (TTT) [19] compress the N-width MLP into one (or several [15]) lightweight module(s), analogous to model compression [30] and knowledge distillation [31]. Therefore, from a fast-weight scaling perspective, we divide the scaling approaches of existing efficient attention methods discussed in this paper into two general categories: a) scaling by routing, and b) scaling by compression (see Fig. 1 for an illustration). Scaling by compression alone sacrifices precise access to the original key-value pairs, whereas scaling by routing alone lacks a global summary of the full context. Although these two approaches are not mutually exclusive, most existing methods typically adopt only one of them.

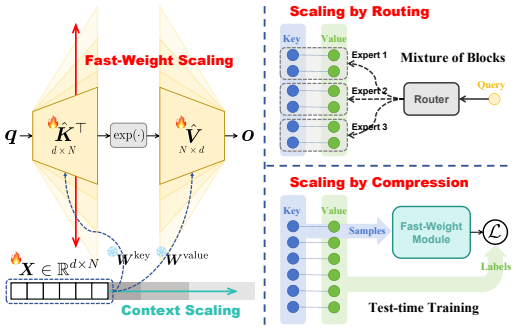


Figure 1: As the context length grows, the width of the two-layer fast-weight MLP induced by full attention increases accordingly. We categorize efficient fast-weight scaling approaches into two strategies: a) scaling by routing and b) scaling by compression, and illustrate each of them with a representative method: MoBA [11] and TTT [19].

In this paper, we first elevate the fast-weight scaling perspective into a five-dimensional taxonomy that accommodates a broad spectrum of prior efficient attention methods (see Tab. 1). While such a

taxonomy is by no means complete, it paves the way towards a unified framework and serves as a useful design principle. Then, we propose **Mi**xture-of-**T**op-k **A**ttention (**MiTA**), which employs a small set of landmark queries to gather top-k attended key-value pairs as query-aware and deformable routed experts, while compressing the N-width MLP into a narrower shared expert. Consequently, our MiTA improves the flexibility of prior MoE attention by moving from rigid to query-aware, deformable experts, and improves the scalability of prior top-k attention by replacing N private (i.e., query-specific) top-k sets with a tunable number m of reusable sets. It further combines the two scaling strategies, i.e., routing and compression, thereby addressing the limitations identified within the taxonomy. Specifically, our MiTA queries original key-value pairs using landmark queries to obtain corresponding landmark values. The resulting landmark query-value pairs (i.e., compressed key-value pairs) form the shared experts, where landmark queries act as keys and landmark values as values. Meanwhile, MiTA reorganizes the original key-value pairs into routed experts by gathering the top-k key-value pairs attended by each landmark query. As shown in Fig. 2, for each query, MiTA concatenates these compressed key-value pairs with a routed subset of the original key-value pairs.

Paper Contributions. The main contributions of the paper are highlighted as follows.

1. **Conceptually**, we introduce a unified perspective that views efficient attention mechanisms as scaling fast weights, systematically organizing existing methods into a five-dimensional taxonomy and naturally revealing their limitations: the rigidity of experts, the non-reusability of top-k sets, and the need to combine routing and compression.
2. **Technically**, we propose a novel efficient attention mechanism, MiTA, which constructs a tunable number of query-aware, deformable experts (equivalently, reusable and routable top-k sets) and achieves scalable fast weights through both routing and compression, thereby addressing these limitations in a principled manner.
3. **Empirically**, we validate the effectiveness and efficiency of MiTA on vision tasks, demonstrating that our MiTA surpasses strong baselines by a large margin while using less wall-clock time, efficiently approaching the expressive capacity of full attention to a remarkable extent.

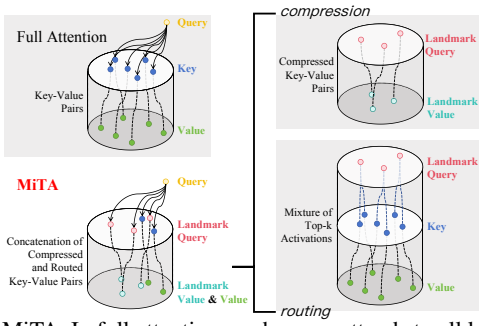


Figure 2: Illustration for our MiTA. In full attention, each query attends to all key-value pairs. In our MiTA, it attends to the concatenation of a small number (i.e., 3 as shown above) of the compressed key-value pairs and a routed subset of the original key-value pairs (i.e., 2 as shown above).

2 Related Work

2.1 Fast-Weight Scaling

In contrast to slow weights (i.e., model parameters that remain unchanged after training and encode persistent knowledge), fast weights are input-conditioned and act as temporary parameters. They can be viewed as short-term memory [32], playing an important role in meta-learning [33]. While most scaling efforts have focused on expanding slow weights (e.g., increasing model width and depth) [34, 35], extending Transformers' sequence length turns out to implicitly scale fast weights.

For instance, test-time training (TTT) [19] effectively compresses the N-width two-layer fast-weight MLP in full attention into a smaller fast-weight module. Similarly, linear attention [14] compresses it into a linear layer [6], while Linformer [16] compresses it into a narrower MLP. Rather than scaling by compressing the full fast-weight MLP, MoE attention factorizes it into experts and accesses them sparsely via routing.

Our MiTA combines the two strategies mentioned above: it compresses full key-value pairs into a smaller set (a narrower MLP), which offers a coarse yet global summary, and then leverages the top-k activations of the landmark queries to identify underlying experts, which enables precise retrieval.

2.2 MoE-Inspired Sparse Attention

Previous sparse attention methods typically focus on the design of fixed sparse patterns (e.g., the local window, axial stripe [36], and the vertical-slash pattern [37]) or learnable ones (e.g., hash buckets [38] and k-means clusters [39]). To maintain global connectivity, shared memory (i.e., tokens that attend to and are attended by all tokens) is also introduced [40]. However, these patterns are either task-dependent [41] or too burdensome to achieve wall-clock speedup [42].

Recently, sparse attention inspired by Mixture-of-Experts (MoE) has been applied to pre-training large language models, yielding extended effective context with reduced overhead [11, 12]. Its key advantage is the routing mechanism, which enables query-aware selection within the key-value cache. Consequently, despite of coarse selection granularity (e.g., evenly split, non-overlapping blocks), the sparse pattern is highly flexible and can vary across queries.

In this paper, we argue that, beyond routing, the experts can also be query-aware and deformable. Specifically, for an arbitrarily long sequence, we can build a fixed, yet configurable number of experts by gathering semantically related key-value pairs. A similarly motivated line of research is to replace the fixed, non-overlapping rectangular patchification in ViTs [43] with a deformable, content-adaptive tokenization scheme [44].

2.3 Deformable Attention

More broadly, conditional computation, the idea underlying MoE, has been explored through the deformable convolution [45] in convolutional neural networks. And deformable attention was introduced for object detection by Zhu et al. [20] and later generalized to Vision Transformers by Xia et al. [21]. Given a query, while being blind to keys, deformable attention predicts offsets relative to a small set of default positions of the keys, thereby inducing a query-aware, deformable sparse attention pattern (i.e., a fast-weight expert in our context).

Recent top-k attention methods, e.g., Spark Attention [23] and DeepSeek Sparse Attention [24], which have been applied to the training of Gemma 3n and DeepSeek-V3.2, respectively, can also be viewed as improved deformable attention. Instead of predicting the spatial positions solely from the query as in prior deformable attention, these methods locate the top-k key-value pairs by allowing each query to take a lightweight skim of the full keys, e.g., using partial features [23] or low precision (FP8) [24].

Our MiTA retains the key advantage of top-k attention by constructing deformable experts (i.e., sparse patterns) that depend on both queries and keys. However, unlike the methods discussed above, we replace the per-query offset prediction [20, 21] or lightweight skim [23, 24] with more efficient per-query routing to reusable sparse patterns, thereby further improving scalability.

3 Methods

3.1 Efficient Attention as Scaling Fast Weights

This subsection will first present the mathematical formulation of the fast-weight MLP equivalent to full attention, and then reinterpret efficient attention as a fast-weight scaling problem. From this perspective, a taxonomy of previous efficient attention methods in terms of how they scale fast weights will be given.

Fast-weight MLP. Formally, full attention (i.e., scaled dot-product attention [46]) can be written as:

$$\mathrm{Atten}(\boldsymbol{q}, \boldsymbol{K}, \boldsymbol{V}) = \boldsymbol{V}\,\mathrm{softmax}\left(\boldsymbol{K}^\top \boldsymbol{q}/\sqrt{d}\right), \tag{1}$$

where $\boldsymbol{q} \in \mathbb{R}^d$ is a query, and the columns of $\boldsymbol{K}, \boldsymbol{V} \in \mathbb{R}^{d\times N}$ are the keys and values, respectively. By contrast, a D-width two-layer MLP, i.e., the feed-forward network (FFN) in Transformers, is:

$$\mathrm{MLP}_\sigma(\boldsymbol{x} \mid \boldsymbol{W}_1; \boldsymbol{W}_2) = \boldsymbol{W}_2\sigma\left(\boldsymbol{W}_1^\top \boldsymbol{x} + \boldsymbol{b}_1\right) + \boldsymbol{b}_2, \tag{2}$$

where $\boldsymbol{x} \in \mathbb{R}^d$ is the input vector, $\boldsymbol{W}_1, \boldsymbol{W}_2 \in \mathbb{R}^{d\times D}$ are weights, $\boldsymbol{b}_1 \in \mathbb{R}^D$ and $\boldsymbol{b}_2 \in \mathbb{R}^d$ are biases[1], and $\sigma(\cdot)$ is an element-wise activation function (e.g., ReLU). Note that full attention in Eq. (1) is equivalent to the following MLP:

$$\mathrm{MLP}_{\exp}(\boldsymbol{q} \mid \hat{\boldsymbol{K}}; \hat{\boldsymbol{V}}) = \hat{\boldsymbol{V}} \exp\left(\hat{\boldsymbol{K}}^\top \boldsymbol{q}\right), \tag{3}$$

$$\hat{\boldsymbol{K}} = \boldsymbol{K}/\sqrt{d},\ \hat{\boldsymbol{V}} = \boldsymbol{V}/\left(\exp(\hat{\boldsymbol{K}}^\top \boldsymbol{q})^\top \mathbf{1}_N\right), \tag{4}$$

where the weights are given by the scaled keys and values, the biases are all zero, and the activation function is the exponential function $\exp(\cdot)$. Thereby, we have demonstrated that full attention in Eq. (1) is equivalent to the *N-width, fast-weight, two-layer MLP* in Eq. (3).

Unbounded fast-weight scaling. Because the hidden dimension of the fast-weight MLP in Eq. (3) always equals the sequence length N, the per-query overhead grows linearly with N rather than remaining fixed as in the slow-weight MLP in Eq. (2). Thus, under the all-to-all lookup paradigm, processing N queries incurs an overall $\mathcal{O}(N^2)$ overhead. This unbounded, rapid growth prevents scaling the fast weights to arbitrarily long sequences. While this challenge is commonly described as the quadratic complexity of full attention in the literature on efficient attention, we refer to it as the *unbounded fast-weight scaling* issue.

Fast-weight scaling taxonomy. Although we have discussed relevant efficient-attention methods in Sec. 2, they have not been systematically organized into a unified fast-weight scaling perspective. We therefore propose a five-dimensional taxonomy: a) *scaling strategy*, i.e., scaling the N-width fast-weight MLP either by compressing it into a lightweight module or by routing each query to a subset of it; b) *expert count*, i.e., how many fast-weight experts are constructed (in particular, scaling by compression typically yields a single shared expert); c) *expert type*, i.e., what module each expert takes (e.g., an MLP or a linear layer); d) *expert construction*, i.e., how experts are formed from the key-value pairs; and e) *routing topology*, i.e., the query-expert assignment pattern. The mapping of existing methods to this taxonomy is summarized in Tab. 1. We defer the in-depth discussion until Appendix B.

3.2 Mixture-of-Top-k Attention (MiTA)

Motivated by the limitations identified through our fast-weight scaling perspective, our method is designed with two goals: a) moving from rigid experts to query-aware, deformable experts and from private (i.e., query-specific) top-k key-value sets to reusable ones; and b) combining compression and routing. We begin with the second goal and show that it naturally leads to the first goal.

Note that both scaling by compression and scaling by routing have inherent limitations. Compressing the full fast-weight MLP inevitably discards information, although the information loss can be mitigated by adopting milder compression schemes (e.g., piecewise compression [15]) or by compressing into a more expressive module (e.g., via test-time training [19]). In contrast, scaling by routing is faithful and lossless, but it often lacks a global view of the context.

To combine the strengths of both strategies, i.e., to retain a compact global summary while enabling precise, token-level retrieval, we introduce a small set of *landmark queries* $\tilde{\boldsymbol{Q}} \in \mathbb{R}^{d\times m}$ with $m \ll N$. These landmark queries probe the full key-value cache and then, via cross-attention, compress it into a global fast-weight module, while simultaneously forming deformable fast-weight experts by gathering the top-k activated (i.e., attended) key-value pairs for each landmark query.

This design is motivated by two observations: a) register tokens can attend to distinct semantic regions of an image [47]; and b) class embeddings [48] (or object queries [49] and mask embeddings [50]) provide a compact global summary for dense prediction tasks [51] and even image generation [52]. The landmark queries can be

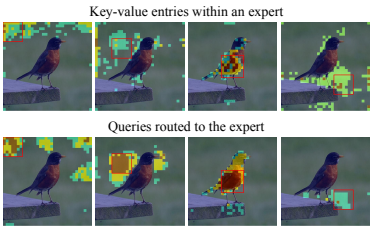


Figure 3: Visualization of key-value pairs gathered by experts and routed queries. The red box marks the local window from which the landmark query is obtained via average pooling. The attention heatmap (averaged over heads) indicates key-value pairs within each expert (top row) and the queries routed to it (bottom row). Notably, neither the expert's key–value pairs nor the routed queries are confined to the local window.

[1]For notational simplicity, biases are omitted from the argument.

Algorithm 1 Mixture-of-Top-k Attention (MiTA)

Require: Query, key, value matrices $\boldsymbol{Q}, \boldsymbol{K}, \boldsymbol{V} \in \mathbb{R}^{d\times N}$; the number of landmark queries m; the width of each fast-weight expert k; each query is routed to the shared expert and to $s = 1$ additional expert.

// Obtain landmark queries
$\tilde{\boldsymbol{Q}} = \mathrm{AdaptiveAvgPool}(\boldsymbol{Q}, \text{output_size}=m)$ // [d, m]
// Lookup key-value pairs via landmark queries
$\boldsymbol{S}^{\mathrm{kv}} = \boldsymbol{K}^\top \tilde{\boldsymbol{Q}}/\sqrt{d}$ // [N, m]
// Gather the top-k activated key-value pairs
$\mathcal{I}^{\mathrm{kv}} = \mathrm{Flatten}(\mathrm{TopK}(\boldsymbol{S}^{\mathrm{kv}\top}, k, \text{dim}=1))$ // [m*k]
$\boldsymbol{K}^{\mathrm{expt}}, \boldsymbol{V}^{\mathrm{expt}} = \boldsymbol{K}[:, \mathcal{I}^{\mathrm{kv}}], \boldsymbol{V}[:, \mathcal{I}^{\mathrm{kv}}]$ // [d, m*k]
// Obtain landmark values (construct the shared expert)
$\tilde{\boldsymbol{V}} = \boldsymbol{V}\,\mathrm{softmax}(\mathbf{S}^{\mathrm{kv}})$ // [d, m]
// Always route queries to the shared expert
$\boldsymbol{O}^{\mathrm{share}} = \mathrm{FlashAttention}(\boldsymbol{Q}, \tilde{\boldsymbol{Q}}, \tilde{\boldsymbol{V}})$
// Sparsely route queries to other experts
$\mathcal{I}^{\mathrm{expt}} = \mathrm{ArgSort}(\mathrm{ArgMax}(\tilde{\boldsymbol{Q}}^\top \boldsymbol{Q}, \text{dim}=0))$ // [N]
$\boldsymbol{O}^{\mathrm{expt}} = \mathrm{FlashAttention}(\boldsymbol{Q}[:, \mathcal{I}^{\mathrm{expt}}], \boldsymbol{K}^{\mathrm{expt}}, \boldsymbol{V}^{\mathrm{expt}},$
cu_seqlens_q= $\mathrm{CumSum}(\mathrm{BinCount}(\mathcal{I}^{\mathrm{expt}}))$, cu_seqlens_k=$[0, k, 2k, \ldots, mk]$)
// Combine results via online softmax
$\boldsymbol{O} = \mathrm{Combine}(\boldsymbol{O}^{\mathrm{share}}, \boldsymbol{O}^{\mathrm{expt}})$ // [d, N]
return $\boldsymbol{O}$

obtained in various ways, e.g., by assigning a set of learnable slow weights or by downsampling the sequence using a convolutional module. In this work, we simply apply average pooling over uniformly spaced, equal-sized windows, as shown in Fig. 3.

Specifically, the i-th landmark query $\tilde{\boldsymbol{q}}_i$ (among m in total) defines the i-th expert $\mathcal{E}_i$ as follows:

$$\mathcal{E}_i(\boldsymbol{q}) = \mathrm{Atten}(\boldsymbol{q}, \boldsymbol{K}^{(i)}, \boldsymbol{V}^{(i)}),\ i \in \{1, \ldots, m\}, \tag{5}$$

$$\boldsymbol{K}^{(i)} = \boldsymbol{K}_{:,\mathcal{I}_i},\ \boldsymbol{V}^{(i)} = \boldsymbol{V}_{:,\mathcal{I}_i} \in \mathbb{R}^{d\times k}, \tag{6}$$

$$\mathcal{I}_i = \mathrm{Top}_k\left(\boldsymbol{K}^\top \tilde{\boldsymbol{q}}_i\right) \in \{1, \ldots, N\}^k, \tag{7}$$

where $\boldsymbol{q}$ is a query routed to expert $\mathcal{E}_i$, and $\boldsymbol{K}^{(i)}, \boldsymbol{V}^{(i)}$ are the top-k key-value pairs activated by $\tilde{\boldsymbol{q}}_i$. This approach effectively reorganizes the full key-value pairs into a mixture of top-k key-value pairs (i.e., a mixture of deformable experts). We directly use $\tilde{\boldsymbol{q}}_i$ as the routing vector for expert $\mathcal{E}_i$. Then the routing logits from the N queries $\boldsymbol{Q} \in \mathbb{R}^{d\times N}$ to the m experts are $\boldsymbol{Q}^\top \tilde{\boldsymbol{Q}} \in \mathbb{R}^{N\times m}$. We denote the index of the rank-j expert that a query $\boldsymbol{q}$ is routed to under these logits as $e_j(\boldsymbol{q}) \in \{1, \ldots, m\}$.

Moreover, regarding the compressed global module, which can be viewed as a shared expert [53], we extract a set of landmark values $\tilde{\boldsymbol{V}} \in \mathbb{R}^{d\times m}$ via cross-attention using the landmark queries. The i-th landmark value corresponding to the i-th landmark query is given by:

$$\tilde{\boldsymbol{v}}_i = \mathrm{Atten}(\tilde{\boldsymbol{q}}_i, \boldsymbol{K}, \boldsymbol{V}). \tag{8}$$

Then, the shared expert $\tilde{\mathcal{E}}$ is defined as:

$$\tilde{\mathcal{E}}(\boldsymbol{q}) = \mathrm{Atten}(\boldsymbol{q}, \tilde{\boldsymbol{Q}}, \tilde{\boldsymbol{V}}), \tag{9}$$

where landmark queries act as the keys in standard attention. Note that the computations in Eqs. (7) and (8) can be largely shared, where landmark queries are both querying the keys. Likewise, Eq. (9) can reuse the routing logits. Therefore, the computations of the two scaling strategies are tightly coupled in our MiTA.

In this paper, rather than a straightforward MoE implementation, which isolates the computation within each expert and aggregates their outputs via a weighted sum, we concatenate the experts as a

single standard attention, as shown in Fig. 2. Therefore, our **MiTA** can be written as:

$$\mathrm{MiTA}(\boldsymbol{q}) = \boldsymbol{V}^\star \,\mathrm{softmax}(\boldsymbol{K}^{\star\top}\boldsymbol{q}/\sqrt{d}), \tag{10}$$

$$\boldsymbol{K}^\star = [\tilde{\boldsymbol{Q}}, \boldsymbol{K}^{(e_1(\boldsymbol{q}))}, \ldots, \boldsymbol{K}^{(e_s(\boldsymbol{q}))}], \tag{11}$$

$$\boldsymbol{V}^\star = [\tilde{\boldsymbol{V}}, \boldsymbol{V}^{(e_1(\boldsymbol{q}))}, \ldots, \boldsymbol{V}^{(e_s(\boldsymbol{q}))}], \tag{12}$$

where $\boldsymbol{K}^\star, \boldsymbol{V}^\star \in \mathbb{R}^{d\times(m+ks)}$ are the key-value pairs that a query $\boldsymbol{q}$ attends to in our MiTA, and s is the number of routed experts (not counting the shared expert) per query. Note that, in practice, the attention can still be computed expert-wise and then combined, thanks to the online softmax [54], as in the implementation [11] of FlashAttention [55].

Implementations. Our MiTA is conceptually simple yet effective, but technically non-trivial to realize with actual wall-clock speedups. In particular, we implement our MiTA in Eq. (10) with $s = 1$ (as in Llama 4), i.e., each query is fed to the shared expert and routed to exactly a single additional fast-weight expert. The pseudocode of our MiTA is given in Algorithm 1, in which queries are matched to experts by sorting queries according to their expert assignments $e_1(\boldsymbol{q})$. The more general case with $s > 1$ can be implemented via MoBA [11] at the cost of reduced speed.

Complexity analysis. The computational complexity of Eq. (10) alone is $\mathcal{O}(N(m+ks))$; whereas the full attention incurs $\mathcal{O}(N^2)$, where in practice $N \gg m+ks$. Moreover, the cross-attention operations in Eqs. (8) and (10) can be accelerated by the FlashAttention family [42, 55, 56]. The primary bottleneck comes from the gathering operation in Eq. (6): it entails irregular (random) memory access and is therefore likely to make the overall pipeline memory-bound. Nonetheless, DeepSeek Sparse Attention suggests that such a limitation can be mitigated via its optimized implementation [24, Fig. 3]. Moreover, compared with prior top-k attention [23, 24], which instantiates a private fast-weight expert per query (where N experts in total) without routing, our MiTA uses a fixed (yet tunable) number (i.e., m) of fast-weight experts, which are more hardware-friendly, at the cost of an N-to-m routing step (as explored in MoBA [11]). Since that the hardware operations required by MiTA have been validated by prior engineering-centric work, in this paper, we focus on the construction of a tunable number of query-aware, deformable fast-weight experts, and the combination of compression and routing.

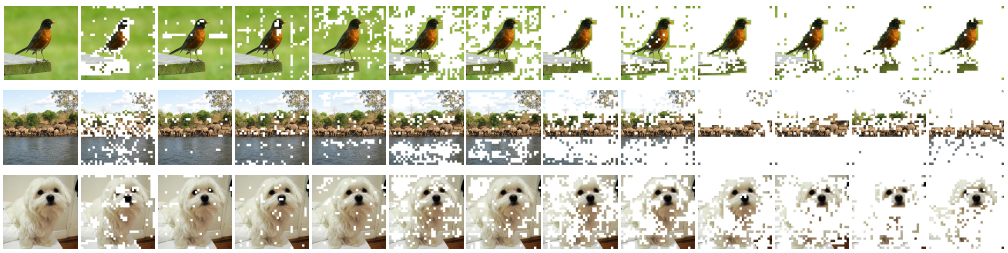


Figure 4: The token pruning effect of MiTA. Each row visualizes, for each layer, the positions of key-value pairs (aggregated over heads) selected as experts; the leftmost image shows the original input. In later layers, most tokens are effectively "pruned" (i.e., not selected as experts), and attention concentrates on class-relevant regions. The examples are taken from the ImageNet-1K training set.

4 Experiments

To verify the efficacy of our MiTA, we conduct image classification experiments on ImageNet-1K [57] and semantic segmentation experiments on ADE20K [58]. Moreover, we assess the long sequence modeling capability of MiTA on the Long Range Arena benchmark [59], and report its wall-clock training/inference speed against full attention on extremely long sequences. A detailed ablation study is provided in Appendix A. Furthermore, we study the robustness (or generalization) of MiTA under varying expert count (m) and width (k) with a fixed trained model, and examine the broader algorithmic generalization of Transformers across attention mechanisms; see Appendix C for details.

Baselines. Note that not all methods listed in Tab. 1 are included in our experiments for the following reasons: a) although representative, many have become outdated; and b) some are designed for language tasks and are therefore not directly comparable in our vision setting. Nonetheless, we include comparisons with their vision counterparts that are based on similar ideas.

Table 2: Results on ImageNet-1K training from scratch. ^{Bias}: using the agent bias [18]; ^{DWC}: using depth-wise convolution; ^{Gate}: adding the attention output to the residual with data-dependent gating. PolaFormer [60], MHLA [15], and FLatten [28] are variants of linear attention. BRA [22] and Agent Attention [18] can be viewed as degenerate cases of MiTA, scaling by routing only and compression only, respectively.

Methods	# Params	FLOPs	Acc. (%)
DeiT-T	5.7M	1.2G	72.2
Linear-DeiT-T	5.7M	1.1G	68.0
PolaFormer-DeiT-T	5.7M	1.2G	68.1
MHLA-DeiT-T	5.7M	1.1G	69.2
FLatten-DeiT-T	6.1M	1.1G	70.2
BRA-DeiT-T	5.7M	1.1G	70.2
Agent-DeiT-T	5.7M	1.1G	70.3
Agent-DeiT-T^{Bias}	6.0M	1.2G	71.1
MiTA-DeiT-T	5.7M	1.1G	71.1 (-1.1)
MiTA-DeiT-T^{DWC}	5.7M	1.1G	73.4 (+1.2)
DeiT-S	22M	4.6G	79.9
Agent-DeiT-S	22M	4.4G	78.5
Agent-DeiT-S^{Bias}	23M	4.4G	79.1
MiTA-DeiT-S	22M	4.4G	79.8 (-0.1)
MiTA-DeiT-S^{DWC}	22M	4.4G	80.6 (+0.7)
MiTA-DeiT-S$^{\text{DWC,Gate}}$	22M	4.7G	81.2 (+1.3)

Table 3: Comparison with SOTA and efficient ViTs on ImageNet-1K. We compare against a recent state-of-the-art ViT variant, ViT-5 [61], as well as strong baselines such as DeiT and DeiT-III [62]. We also include efficient ViTs, including sparse models (SViTE [63] and Sparsifiner [64]) and scaling-by-compression models (InLine [65] and Agent Attention [18]).

Methods	# Params	FLOPs	Acc. (%)
ViT-5-S	22M	4.7G	82.2
DeiT-S	22M	4.6G	79.9
DeiT-III-S	22M	4.6G	81.4
SViTE-S	11M	2.5G	79.7
Sparsifiner-S	23M	4.5G	79.9
InLine-DeiT-S	17M	5.0G	80.2
Agent-DeiT-S$^{\text{Bias,DWC}}$	23M	4.4G	80.5
MHLA-DeiT-S^{DWC}	22M	4.2G	81.0
MiTA-ViT-5-S	22M	4.5G	81.3 (-0.9)
MiTA-ViT-5-S^{DWC}	22M	4.5G	81.7 (-0.5)

4.1 Image Classification

Current advances in efficient attention on ImageNet-1K are typically accompanied by extra components such as depth-wise convolutions, which obscure the expressiveness gap between the proposed efficient mechanisms and full attention. To this end, we report a fair comparison in Tab. 2. Specifically, we train all models using the DeiT [67] recipe and vary only the attention mechanism. Meanwhile, in Tab. 3, we compare our best-performing ViT variant based on MiTA against state-of-the-art and efficient ViTs.

Implementation details. We use an image size of 224 with a patch size of 16, and set $s = 1$, $m = k = 25$ for our MiTA.[2] Under this setting, each query attends to $m + ks = 50$ key-value pairs.

Results. As indicated in Tab. 2, without extra components, our MiTA outperforms other efficient attention by a substantial margin (at least 0.8% and up to 3.1%). Although Agent Attention [18] can narrow the gap by using agent bias[3], it still under-performs our MiTA on small-sized models by 0.7%. Moreover, a severe limitation of this trick is that it is currently not supported by FlashAttention [55]. As shown in Tab. 3, when equipped with the architectural modifications explored in ViT-5 [61], our MiTA outperforms a range of strong ViT baselines and approaches state-of-the-art performance within a small margin while using fewer FLOPs.

[2]We set $s = 1$ throughout this work for a preliminary study. As analyzed in Appendix B, this choice does not fully exploit the expressiveness of MiTA.

[3]An agent bias $\boldsymbol{B} \in \mathbb{R}^{N \times m}$ is added to attention scores as $\mathrm{softmax}(\boldsymbol{K}^\top \tilde{\boldsymbol{Q}} + \boldsymbol{B})$.

Table 4: ImageNet finetuning results. We finetune ImageNet-21K pretrained ViTs [66] on ImageNet-1K, replacing the standard attention with Agent Attention [18] or our MiTA. †: the number of agent tokens (i.e., landmark queries) is increased from 49 (default) to 64.

Methods	Tiny	Small	Base	Large
Standard Attention	76.9	81.2	84.4	85.9
Linear Attention	73.2	78.6	79.8	80.7
Agent Attention	74.6	79.7	81.4	83.5
Agent Attention†	74.7	79.9	81.5	83.8
MiTA	75.6 (-1.3)	80.9 (-0.3)	82.8 (-1.6)	85.5 (-0.4)

Table 5: Results on ADE20K semantic segmentation. $^{\triangledown}$: the attention mechanism is directly replaced, rather than being natively pretrained. Mask Transformer is the decoder introduced in the pioneering work Segmenter [48] for Transformer-based segmentation, while DEPICT [51] and CBT [70] are two principled improvements.

Backbones	Seg. head	Resolution	FLOPs	mIoU (%)
ViT-T	Mask Trans.	512^2	13G	38.1
ViT-T	DEPICT	512^2	13G	39.3
ViT-T	CBT	512^2	12G	39.1
MiTA-ViT-T$^{\triangledown}$	CBT	512^2	7G (↓42%)	36.5 (-2.6)
ViT-S	Mask Trans.	512^2	38G	45.3
ViT-S	DEPICT	512^2	35G	46.7
ViT-S	CBT	512^2	33G	45.8
MiTA-ViT-S$^{\triangledown}$	CBT	512^2	25G (↓24%)	44.5 (-1.3)
ViT-B	Mask Trans.	512^2	128G	48.5
ViT-B	DEPICT	512^2	111G	49.2
ViT-B	CBT	512^2	111G	49.3
MiTA-ViT-B$^{\triangledown}$	CBT	512^2	95G (↓14%)	48.1 (-1.2)
ViT-L	Mask Trans.	640^2	667G	51.8
ViT-L	DEPICT	640^2	628G	52.9
ViT-L	CBT	640^2	622G	53.3
MiTA-ViT-L$^{\triangledown}$	CBT	640^2	508G (↓18%)	50.5 (-2.8)

Attention adaptation. Given that many vision foundation models [43, 68, 69] are pretrained with standard attention, we investigate whether they can be adapted to MiTA via lightweight finetuning. As shown in Tab. 4, parameters pretrained with standard attention transfer more easily to MiTA, while Agent Attention [18] struggles to narrow this gap even when increasing the number of agent tokens.

4.2 Semantic Segmentation

ADE20K [58] is a scene-centric dataset for semantic segmentation, which requires higher image resolutions to preserve fine-grained details such as object boundaries. In Tab. 5, we apply our MiTA to the backbone for efficient visual encoding and compare against methods with ViT backbones and Transformer-based segmentation heads.

Implementation details. The ViT backbones [66] are pretrained on ImageNet-21K with an image size of 384 and a patch size of 16. We use a consistent patch size of 16 when training on ADE20K, and set $s = 1$, $m = k = 49$ for our MiTA. Under this setting, each query attends to $m + ks = 98$ key-value pairs, while it is $1{,}024$ (for 512^2 image resolution) or $1{,}600$ (640^2 resolution) in full attention.

Results. As depicted in Tab. 5, our MiTA remarkably reduces the FLOPs (by up to 42%) while achieving comparable segmentation performance. Note that our MiTA is not fully exploited here because the backbone is not natively pretrained with it.

4.3 Evaluation on Long Sequences

As a preliminary validation for long-sequence modeling, we evaluate our MiTA on the Long Range Arena (LRA) benchmark [59] and compare against several classical efficient attention methods [38, 16, 71, 72]. We report the accuracy and training throughput (and wall-clock time) on LRA in Tab. 6, while the inference throughput improvement of MiTA over standard attention is shown in Fig. 5.

Table 6: Results on LRA benchmark. For each task, we report the accuracy and training throughput (steps/s), along with the average accuracy across tasks and the total training wall-clock time (hours). ‡: a route-only variant (i.e., the compression or local window branch is removed), where the number of entries attended per query is kept unchanged by increasing k.

Methods	ListOps (2K)	Text (4K)	Retrieval (4K)	Image (1K)	Pathfinder (1K)	Avg. / Tot.
Standard Atten	37.35 / 12.7	63.63 / 3.8	79.90 / 1.9	40.38 / 5.7	69.68 / 5.7	58.19 / 10.6
Reformer	18.55 / 34.1	65.08 / 16.8	78.77 / 8.5	44.04 / 8.3	69.04 / 8.5	55.10 / 4.5
Linformer	38.05 / 35.0	56.69 / 24.6	78.41 / 13.7	39.56 / 11.9	67.13 / 11.9	55.97 / 3.1
Performer	17.84 / 28.0	65.13 / 16.5	77.18 / 8.9	36.64 / 8.1	70.20 / 8.0	53.40 / 4.7
Nyströmformer	29.69 / 22.0	65.84 / 16.7	79.68 / 8.8	36.83 / 4.5	72.50 / 4.5	56.91 / 7.3
Agent Atten	36.59 / 40.1	63.86 / 38.9	79.98 / 24.3	38.92 / 19.5	71.77 / 19.1	58.22 / 1.9
MoBA‡	37.10 / 30.4	62.35 / 15.8	76.64 / 8.2	37.92 / 8.6	69.67 / 8.4	56.74 / 4.6
MiTA‡	37.20 / 25.4	64.09 / 22.2	79.92 / 13.1	40.17 / 5.3	72.66 / 6.3	58.81 / 5.5
MiTA	37.20 / 52.5 (×4.1)	63.47 / 37.7 (×9.9)	79.88 / 19.6 (×10.3)	40.72 / 13.6 (×2.4)	73.26 / 15.3 (×2.7)	58.91 / 2.4 (↓77%)

Implementation details. The training configuration follows Xiong et al. [72] and the hyperparameters are kept consistent across all methods for a fair comparison. Thus, the reported accuracies are not tuned to fully exploit the best performance of each approach. We set $s = 1$ and $m = k = 128$ for MiTA. Throughput and wall-clock time reported in Tab. 6 and Fig. 5 are measured on a 24GB RTX 4090. Standard attention is implemented with PyTorch's fused kernel (i.e., FlashAttention [55]). More specifically, results in Fig. 5 are measured on a three-layer Transformer with an embedding dimension of 128, and the batch size is tuned for each run to maximize throughput.

Results. As shown in Tab. 6, our MiTA achieves accuracy comparable to standard attention, while providing a substantial speedup that reduces total training time by 77%. Notably, the route-only variant of MiTA also performs well but is slower, since gathering more key-value pairs incurs higher overhead. This suggests that using a shared, compressed set of key-value pairs can effectively represent thousands of routed original pairs, thereby reducing the overhead.

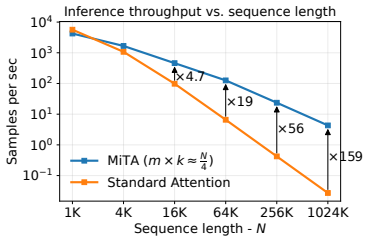


Figure 5: Inference throughput.

Table 7: Long-context classification performance.

Method	16K	32K	64K
Standard Attention	69.8	89.7	97.0
Linear Attention	67.9	87.3	96.4
Linformer	64.1	87.5	97.4
Performer	68.9	86.5	96.6
MiTA	88.0 (+18.2)	92.9 (+3.2)	98.7 (+1.7)

Long-document classification. We further evaluate our MiTA on arXiv Classification [73], which requires both holistic document understanding and sensitivity to fine-grained cues distributed throughout long documents to identify an article's category (e.g., computer vision or machine learning). Tab. 7 compares our results with the results reported by Joshi et al. [74].

5 Conclusion

We adopted fast-weight scaling as a unifying perspective for efficient attention and introduced a five-dimensional taxonomy to identify the limitations of prior methods. Moreover, we proposed a novel efficient attention method, Mixture-of-Top-k Attention (MiTA), which constructs a tunable number of query-aware, deformable fast-weight experts and bridges the two scaling strategies (i.e., compression and routine) but getting rid of their inherent limitations. We provided a detailed analysis of both the taxonomy and our MiTA, suggesting a principled and promising path for developing efficient attention from the fast-weight scaling perspective.

Acknowledgments and Disclosure of Funding

This work is supported by the National Natural Science Foundation of China under Grant 62576048. C.-G. Li is the corresponding author.

References

[1] Alberto Bietti, Vivien Cabannes, Diane Bouchacourt, Hervé Jégou, and Léon Bottou. Birth of a transformer: A memory viewpoint. In *NeurIPS*, 2023.

[2] Jiecheng Lu and Shihao Yang. Free energy mixer. In *ICLR*, 2026.

[3] Jintao Zhang, Rundong Su, Chunyu Liu, Jia Wei, Ziteng Wang, Pengle Zhang, Haoxu Wang, Huiqiang Jiang, Haofeng Huang, Chendong Xiang, Haocheng Xi, Shuo Yang, Xingyang Li, Yuezhou Hu, Tianyu Fu, Tianchen Zhao, Yicheng Zhang, Youhe Jiang, Chang Chen, Kai Jiang, Huayu Chen, Min Zhao, Xiaoming Xu, Jun Zhu, and Jianfei Chen. A survey of efficient attention methods: Hardware-efficient, sparse, compact, and linear attention. 2025.

[4] Mor Geva, Roei Schuster, Jonathan Berant, and Omer Levy. Transformer feed-forward layers are key-value memories. In *EMNLP*, 2021.

[5] Imanol Schlag, Kazuki Irie, and Jürgen Schmidhuber. Linear transformers are secretly fast weight programmers. In *ICML*, 2021.

[6] Dongchen Han, Yining Li, Tianyu Li, Zixuan Cao, Ziming Wang, Jun Song, Yu Cheng, Bo Zheng, and Gao Huang. ViT^3: Unlocking test-time training in vision. *arXiv preprint arXiv:2512.01643*, 2025.

[7] Jiecheng Lu and Shihao Yang. HyperMLP: An integrated perspective for sequence modeling. In *ICML*, 2026.

[8] Bohong Wu, Mengzhao Chen, Xiang Luo, Shen Yan, Qifan Yu, Fan Xia, Tianqi Zhang, Hongrui Zhan, Zheng Zhong, Xun Zhou, et al. Parallel loop transformer for efficient test-time computation scaling. *arXiv preprint arXiv:2510.24824*, 2025.

[9] Hongrong Cheng, Miao Zhang, and Javen Qinfeng Shi. A survey on deep neural network pruning: Taxonomy, comparison, analysis, and recommendations. *TPAMI*, 46(12):10558–10578, 2024.

[10] 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.

[11] Enzhe Lu, Zhejun Jiang, Jingyuan Liu, Yulun Du, Tao Jiang, Chao Hong, Shaowei Liu, Weiran He, Enming Yuan, Yuzhi Wang, et al. MoBA: Mixture of block attention for long-context LLMs. *arXiv preprint arXiv:2502.13189*, 2025.

[12] Jingyang Yuan, Huazuo Gao, Damai Dai, Junyu Luo, Liang Zhao, Zhengyan Zhang, Zhenda Xie, Yuxing Wei, Lean Wang, Zhiping Xiao, et al. Native sparse attention: Hardware-aligned and natively trainable sparse attention. In *ACL*, 2025.

[13] Jürgen Schmidhuber. Learning to control fast-weight memories: An alternative to dynamic recurrent networks. *Neural Computation*, 4(1):131–139, 1992.

[14] Angelos Katharopoulos, Apoorv Vyas, Nikolaos Pappas, and François Fleuret. Transformers are RNNs: Fast autoregressive transformers with linear attention. In *ICML*, 2020.

[15] Kewei Zhang, Ye Huang, Yufan Deng, Jincheng Yu, Junsong Chen, Huan Ling, Enze Xie, and Daquan Zhou. MHLA: Restoring expressivity of linear attention via token-level multi-head. *arXiv preprint arXiv:2601.07832*, 2026.

[16] Sinong Wang, Belinda Z Li, Madian Khabsa, Han Fang, and Hao Ma. Linformer: Self-attention with linear complexity. *arXiv preprint arXiv:2006.04768*, 2020.

[17] 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.

[18] Dongchen Han, Tianzhu Ye, Yizeng Han, Zhuofan Xia, Siyuan Pan, Pengfei Wan, Shiji Song, and Gao Huang. Agent attention: On the integration of softmax and linear attention. In *ECCV*, 2024.

[19] Yu Sun, Xinhao Li, Karan Dalal, Jiarui Xu, Arjun Vikram, Genghan Zhang, Yann Dubois, Xinlei Chen, Xiaolong Wang, Sanmi Koyejo, Tatsunori Hashimoto, and Carlos Guestrin. Learning to (learn at test time): RNNs with expressive hidden states. In *ICML*, 2025.

[20] Xizhou Zhu, Weijie Su, Lewei Lu, Bin Li, Xiaogang Wang, and Jifeng Dai. Deformable DETR: Deformable transformers for end-to-end object detection. In *ICLR*, 2021.

[21] Zhuofan Xia, Xuran Pan, Shiji Song, Li Erran Li, and Gao Huang. Vision transformer with deformable attention. In *CVPR*, 2022.

[22] Lei Zhu, Xinjiang Wang, Zhanghan Ke, Wayne Zhang, and Rynson W. H. Lau. BiFormer: Vision transformer with bi-level routing attention. In *CVPR*, 2023.

[23] Chong You, Kan Wu, Zhipeng Jia, Lin Chen, Srinadh Bhojanapalli, Jiaxian Guo, Utku Evci, Jan Wassenberg, Praneeth Netrapalli, Jeremiah J Willcock, et al. Spark transformer: Reactivating sparsity in FFN and attention. In *NeurIPS*, 2025.

[24] Aixin Liu, Aoxue Mei, Bangcai Lin, Bing Xue, Bingxuan Wang, Bingzheng Xu, Bochao Wu, Bowei Zhang, Chaofan Lin, Chen Dong, et al. DeepSeek-V3.2: Pushing the frontier of open large language models. *arXiv preprint arXiv:2512.02556*, 2025.

[25] Jianzong Wu, Liang Hou, Haotian Yang, Xin Tao, Ye Tian, Pengfei Wan, Di Zhang, and Yunhai Tong. VMoBA: Mixture-of-block attention for video diffusion models. *arXiv preprint arXiv:2506.23858*, 2025.

[26] Shengqu Cai, Ceyuan Yang, Lvmin Zhang, Yuwei Guo, Junfei Xiao, Ziyan Yang, Yinghao Xu, Zhenheng Yang, Alan Yuille, Leonidas Guibas, et al. Mixture of contexts for long video generation. *arXiv preprint arXiv:2508.21058*, 2025.

[27] Weinan Jia, Yuning Lu, Mengqi Huang, Hualiang Wang, Binyuan Huang, Nan Chen, Mu Liu, Jidong Jiang, and Zhendong Mao. MoGA: Mixture-of-groups attention for end-to-end long video generation. *arXiv preprint arXiv:2510.18692*, 2025.

[28] Dongchen Han, Xuran Pan, Yizeng Han, Shiji Song, and Gao Huang. FLatten transformer: Vision transformer using focused linear attention. In *ICCV*, 2023.

[29] Jiecheng Lu, Xu Han, Yan Sun, Viresh Pati, Yubin Kim, Siddhartha Somani, and Shihao Yang. ZeroS: Zero-sum linear attention for efficient transformers. In *NeurIPS*, 2025.

[30] Jimmy Ba and Rich Caruana. Do deep nets really need to be deep? In *NeurIPS*, 2014.

[31] Geoffrey Hinton, Oriol Vinyals, and Jeff Dean. Distilling the knowledge in a neural network. *arXiv preprint arXiv:1503.02531*, 2015.

[32] Ali Behrouz, Peilin Zhong, and Vahab Mirrokni. Titans: Learning to memorize at test time. *arXiv preprint arXiv:2501.00663*, 2024.

[33] Louis Kirsch, James Harrison, Jascha Sohl-Dickstein, and Luke Metz. General-purpose in-context learning by meta-learning transformers. *arXiv preprint arXiv:2212.04458*, 2022.

[34] Jared Kaplan, Sam McCandlish, Tom Henighan, Tom B Brown, Benjamin Chess, Rewon Child, Scott Gray, Alec Radford, Jeffrey Wu, and Dario Amodei. Scaling laws for neural language models. *arXiv preprint arXiv:2001.08361*, 2020.

[35] Mingxing Tan and Quoc Le. EfficientNet: Rethinking model scaling for convolutional neural networks. In *ICML*, 2019.

[36] Xiaoyi Dong, Jianmin Bao, Dongdong Chen, Weiming Zhang, Nenghai Yu, Lu Yuan, Dong Chen, and Baining Guo. CSWin transformer: A general vision transformer backbone with cross-shaped windows. In *CVPR*, 2022.

[37] Huiqiang Jiang, Yucheng Li, Chengruidong Zhang, Qianhui Wu, Xufang Luo, Surin Ahn, Zhenhua Han, Amir H. Abdi, Dongsheng Li, Chin-Yew Lin, Yuqing Yang, and Lili Qiu. MInference 1.0: Accelerating pre-filling for long-context LLMs via dynamic sparse attention. In *NeurIPS*, 2024.

[38] Nikita Kitaev, Lukasz Kaiser, and Anselm Levskaya. Reformer: The efficient transformer. In *ICLR*, 2020.

[39] Aurko Roy, Mohammad Saffar, Ashish Vaswani, and David Grangier. Efficient content-based sparse attention with routing transformers. *TACL*, 9:53–68, 2021.

[40] Iz Beltagy, Matthew E Peters, and Arman Cohan. Longformer: The long-document transformer. *arXiv preprint arXiv:2004.05150*, 2020.

[41] Xunhao Lai, Jianqiao Lu, Yao Luo, Yiyuan Ma, and Xun Zhou. FlexPrefill: A context-aware sparse attention mechanism for efficient long-sequence inference. In *ICLR*, 2025.

[42] Tri Dao, Dan Fu, Stefano Ermon, Atri Rudra, and Christopher Ré. FlashAttention: Fast and memory-efficient exact attention with IO-awareness. *NeurIPS*, 2022.

[43] 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.

[44] Zhiyang Chen, Yousong Zhu, Chaoyang Zhao, Guosheng Hu, Wei Zeng, Jinqiao Wang, and Ming Tang. DPT: Deformable patch-based transformer for visual recognition. In *ACMMM*, 2021.

[45] Jifeng Dai, Haozhi Qi, Yuwen Xiong, Yi Li, Guodong Zhang, Han Hu, and Yichen Wei. Deformable convolutional networks. In *ICCV*, 2017.

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

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

[48] Robin Strudel, Ricardo Garcia, Ivan Laptev, and Cordelia Schmid. Segmenter: Transformer for semantic segmentation. In *ICCV*, 2021.

[49] Nicolas Carion, Francisco Massa, Gabriel Synnaeve, Nicolas Usunier, Alexander Kirillov, and Sergey Zagoruyko. End-to-end object detection with transformers. In *ECCV*, 2020.

[50] Bowen Cheng, Alex Schwing, and Alexander Kirillov. Per-pixel classification is not all you need for semantic segmentation. In *NeurIPS*, 2021.

[51] Qishuai Wen and Chun-Guang Li. Rethinking decoders for transformer-based semantic segmentation: A compression perspective. In *NeurIPS*, 2024.

[52] Qihang Yu, Mark Weber, Xueqing Deng, Xiaohui Shen, Daniel Cremers, and Liang-Chieh Chen. An image is worth 32 tokens for reconstruction and generation. In *NeurIPS*, 2024.

[53] Damai Dai, Chengqi Deng, Chenggang Zhao, R. X. Xu, Huazuo Gao, Deli Chen, Jiashi Li, Wangding Zeng, Xingkai Yu, Y. Wu, Zhenda Xie, Y. K. Li, Panpan Huang, Fuli Luo, Chong Ruan, Zhifang Sui, and Wenfeng Liang. DeepSeekMoE: Towards ultimate expert specialization in mixture-of-experts language models. In *ACL*, 2024.

[54] Maxim Milakov and Natalia Gimelshein. Online normalizer calculation for softmax. *arXiv preprint arXiv:1805.02867*, 2018.

[55] Tri Dao. FlashAttention-2: Faster attention with better parallelism and work partitioning. In *ICLR*, 2024.

[56] Jay Shah, Ganesh Bikshandi, Ying Zhang, Vijay Thakkar, Pradeep Ramani, and Tri Dao. FlashAttention-3: Fast and accurate attention with asynchrony and low-precision. In *NeurIPS*, 2024.

[57] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. ImageNet: A large-scale hierarchical image database. In *CVPR*, 2009.

[58] Bolei Zhou, Hang Zhao, Xavier Puig, Tete Xiao, Sanja Fidler, Adela Barriuso, and Antonio Torralba. Semantic understanding of scenes through the ADE20K dataset. *IJCV*, 127(3):302–321, 2019.

[59] Yi Tay, Mostafa Dehghani, Samira Abnar, Yikang Shen, Dara Bahri, Philip Pham, Jinfeng Rao, Liu Yang, Sebastian Ruder, and Donald Metzler. Long range arena : A benchmark for efficient transformers. In *ICLR*, 2021.

[60] Weikang Meng, Yadan Luo, Xin Li, Dongmei Jiang, and Zheng Zhang. PolaFormer: Polarity-aware linear attention for vision transformers. In *ICLR*, 2025.

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

[62] Hugo Touvron, Matthieu Cord, and Hervé Jégou. DeiT III: Revenge of the ViT. In *ECCV*, 2022.

[63] Tianlong Chen, Yu Cheng, Zhe Gan, Lu Yuan, Lei Zhang, and Zhangyang Wang. Chasing sparsity in vision transformers: An end-to-end exploration. In *NeurIPS*, 2021.

[64] Cong Wei, Brendan Duke, Ruowei Jiang, Parham Aarabi, Graham W Taylor, and Florian Shkurti. Sparsifiner: Learning sparse instance-dependent attention for efficient vision transformers. In *CVPR*, 2023.

[65] Dongchen Han, Yifan Pu, Zhuofan Xia, Yizeng Han, Xuran Pan, Xiu Li, Jiwen Lu, Shiji Song, and Gao Huang. Bridging the divide: Reconsidering softmax and linear attention. In *NeurIPS*, 2024.

[66] Andreas Peter Steiner, Alexander Kolesnikov, Xiaohua Zhai, Ross Wightman, Jakob Uszkoreit, and Lucas Beyer. How to train your ViT? Data, augmentation, and regularization in vision transformers. *TMLR*, 2022.

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

[68] Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, et al. Learning transferable visual models from natural language supervision. In *ICML*, 2021.

[69] Oriane Siméoni, Huy V Vo, Maximilian Seitzer, Federico Baldassarre, Maxime Oquab, Cijo Jose, Vasil Khalidov, Marc Szafraniec, Seungeun Yi, Michaël Ramamonjisoa, et al. DINOv3. *TMLR*, 2026.

[70] Qishuai Wen, Zhiyuan Huang, and Chun-Guang Li. Towards interpretable and efficient attention: Compressing all by contracting a few. In *NeurIPS*, 2025.

[71] Krzysztof Marcin Choromanski, Valerii Likhosherstov, David Dohan, Xingyou Song, Andreea Gane, Tamás Sarlós, Peter Hawkins, Jared Quincy Davis, Afroz Mohiuddin, Lukasz Kaiser, David Benjamin Belanger, Lucy J. Colwell, and Adrian Weller. Rethinking attention with performers. In *ICLR*, 2021.

[72] Yunyang Xiong, Zhanpeng Zeng, Rudrasis Chakraborty, Mingxing Tan, Glenn Fung, Yin Li, and Vikas Singh. Nyströmformer: A nyström-based algorithm for approximating self-attention. In *AAAI*, 2021.

[73] Jun He, Liqun Wang, Liu Liu, Jiao Feng, and Hao Wu. Long document classification from local word glimpses via recurrent attention learning. *IEEE Access*, 2019.

[74] Sahil Joshi, Agniva Chowdhury, Amar Kanakamedala, Ekam Singh, Evan Tu, and Anshumali Shrivastava. RACE attention: A strictly linear-time attention layer for training on outrageously large contexts. In *ICLR*, 2026.

[75] Daniel Bolya, Cheng-Yang Fu, Xiaoliang Dai, Peizhao Zhang, Christoph Feichtenhofer, and Judy Hoffman. Token merging: Your ViT but faster. In *ICLR*, 2023.

[76] Liang Zhao, Xiachong Feng, Xiaocheng Feng, Weihong Zhong, Dongliang Xu, Qing Yang, Hongtao Liu, Bing Qin, and Ting Liu. Length extrapolation of transformers: A survey from the perspective of positional encoding. In *EMNLP*, 2024.

Appendix

A Ablation Study

In Tab. 8, we ablate three key design choices in MiTA: a) the landmark extraction strategy; b) the number of experts (m) and the number of key-value pairs per expert (k); and c) the standalone effects of routing and compression. In the following paragraphs, we analyze these results and their implications.

Table 8: Ablation study under the same setting as Tab. 2.

Setting	Acc.	Δ
Landmark Extraction		
Random Selection	70.6	-0.5
Learnable Parameters	66.3	-4.8
1D Average Pooling	70.4	-0.7
2D Average Pooling	71.1	Default
Convolution	67.0	-4.1
Depth-Wise Convolution	68.8	-2.3
Token Merging [75]	**71.3**	**+0.2**
$m \times k$		
16×16	70.0	-1.1
16×25	71.0	-0.1
25×16	70.5	-0.6
25×25	71.1	Default
25×36	71.7	+0.6
36×25	71.2	+0.1
36×36	71.7	+0.6
Compression & Routing		
Compress-and-route	71.1	Default
Compress-only	70.3	-0.8
Route-only	70.8	-0.3

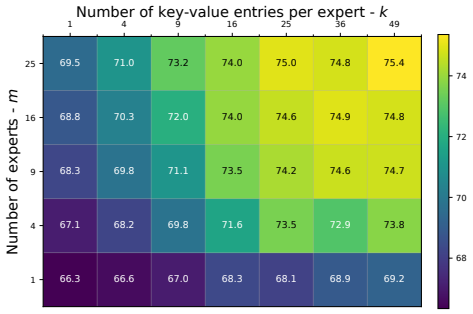


Figure 6: Ablation study of m and k on CIFAR-100. Experiments with other (m, k) configurations were not conducted due to out-of-memory (OOM) errors.

Landmark extraction. Our simple default choice—average pooling over evenly split, non-overlapping rectangular regions—even outperforms parameterized alternatives. This may stem from the fact that landmark queries serve the dual roles of compressed keys and routers, potentially leading to gradient conflicts and suggesting a direction for further improving MiTA. We also observe that more advanced methods, such as ToMe [75], can improve MiTA. However, ToMe must be applied repeatedly to achieve aggressive compression, making it a less lightweight and less elegant solution.

Hyperparameters m and k. Performance improves as either m or k increases, which is expected since MiTA more closely approximates full attention with larger m or k, and recovers full attention when $m = k = N$. More importantly, we observe that performance is more sensitive to k than to m; that is, increasing k is typically more beneficial than increasing m. This trend is further validated in

Fig. 6. This finding supports our motivation that gathering top-k activations independently for each query (e.g., as in Spark Attention [23] and DeepSeek Sparse Attention [24]) is redundant, and that many such activations can instead be shared through routing. For practical use, we provide a simple rule of thumb: first choose a fixed ratio $\frac{m \times k}{N}$; then start from $m = k$ and explore $k > m$ during subsequent tuning.

Compression and routing. Routing is more critical than compression. Nevertheless, introducing compression further improves the performance while also providing a notable acceleration, as shown in Tab. 6.

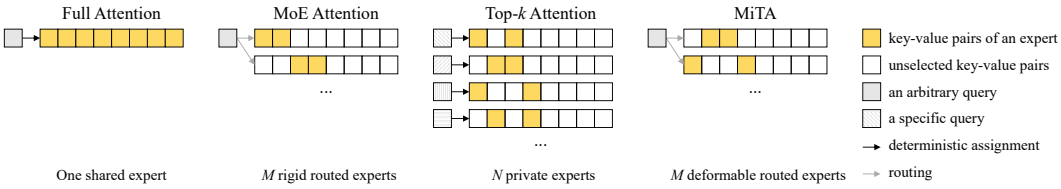


Figure 7: Visual comparison of relevant attention methods.

B Fitting MiTA into the Fast-Weight Scaling Taxonomy

While MiTA is introduced in Sec. 3.2 from the perspective of combining the two complementary scaling strategies, we find that it also enriches the remaining dimensions of the fast-weight scaling taxonomy proposed in Sec. 3.1. This section will briefly discuss each of them.

Expert construction. This core dimension determines the expert type and count, and hence the routing topology. An important consideration is what the construction conditions on: the query only (e.g., DAT [21]), the key-value pairs only (e.g., MHLA [15]), or both. When using key-value pairs, the construction may be content-dependent and thus deformable (e.g., DSA [24]) or merely position-driven (e.g., MoBA [11]). While most efficient attention methods condition on one of the aspects mentioned above, our MiTA conditions on all of them by extracting landmark queries from queries and then probing the full key–value pool. We illustrate the resultant benefit of this design in Fig. 4.

Expert type and count. Expert types can be categorized into three classes: a) linear layers (e.g., linear attention), b) MLPs (e.g., sparse attention and PVT [17]), c) arbitrary modules (e.g., test-time training). The scaling-by-routing branch of MiTA is restricted to MLP experts, as in sparse attention. However, the scaling-by-compression branch can be implemented through test-time training, thereby generalizing to other modules. As for the expert count, MiTA controls it via the hyperparameter m. One benefit of this design has been discussed in the complexity analysis. Another benefit is that it enables expert composition through the routing mechanism.

Routing topology. In MiTA, there is a fixed number of base sparse patterns (i.e., fast-weight experts), which is a substantial improvement over efficient attention with only a single expert, yet is still far fewer than prior top-k attention [23, 24], which has N experts. Crucially, to compensate for this limitation, MiTA can resort to the routing mechanism to compose these m base experts: when routing each query to s experts, the number of effective sparse patterns (i.e., the combinatorial number of experts) is $\binom{m}{s}$. In particular, when $s = 1$, routing may degenerate into clustering, restricting information flow within each cluster. We thus quantify the positional overlap between the gathered key–value pairs by an expert and the queries routed to it. As shown in Fig. 8, the overlap remains consistently modest across layers, suggesting that MiTA with $s = 1$ performs routing rather than hard clustering.

C Algorithmic Generalization

Broadly, generalization typically refers to performance beyond the training data, e.g., from the training set to the validation set, length extrapolation [76] (from the training resolution to a new one). Motivated by the unifying perspective for efficient attention, this work investigates a new dimension of generalization: *algorithmic generalization*, i.e., performance beyond the training algorithm (e.g., attention mechanism).

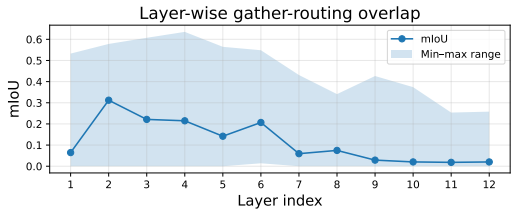
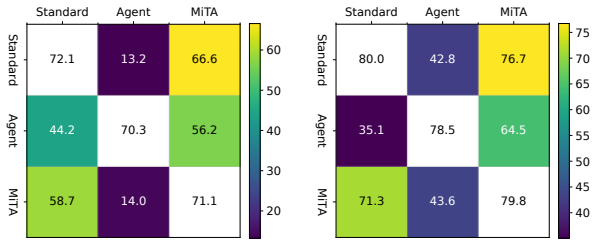


Figure 8: The layer-wise positional overlap (between the key–value pairs gathered by an expert and the queries routed to it) is quantified by mIoU, averaged over experts and attention heads.



(a) DeiT-T models (b) DeiT-S models

Figure 9: Evaluation with a different inference attention. The horizontal axis indexes the training attention, while the vertical axis indexes the inference attention. We omit the diagonal entries from the heatmap since they are not of interest. Currently, only standard attention, Agent Attention, and MiTA are included.

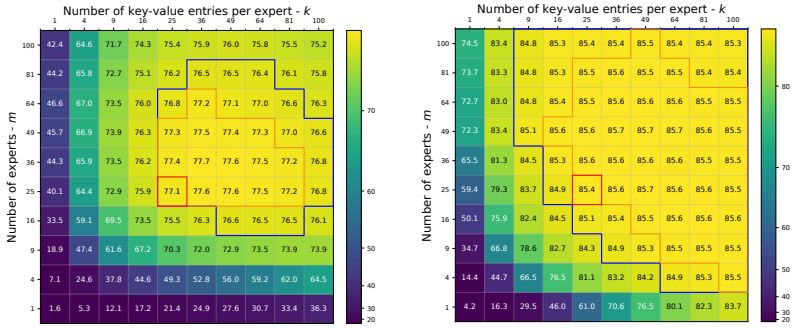
In Fig. 10, we investigate MiTA's generalization across hyperparameters: the expert count m and expert width k. Note that changing the hyperparameters of an algorithm is analogous to changing the input resolution. Moreover, in Fig. 9, we study how an attention mechanism generalizes to others. Specifically, given a model trained with one attention mechanism (*training attention*), we replace it with a different mechanism (*inference attention*) at test time. Beyond the generalization setting, where model parameters are fixed, we also finetune models pretrained on a large-scale dataset with a different *finetuning attention* in Tab. 4.

Implementation details. ViTs [66] in Tab. 4 are pretrained on ImageNet-21K with an image size of 224 and a patch size of 16. We finetune them for 50 epochs with the AdamW optimizer, using the listed attention mechanisms (including standard attention). The model in Fig. 10(a) is similarly finetuned, but starts from a ViT pretrained with an image size of 384. Models in Fig. 9 are the same as those in Tab. 2.

Results. Fig. 10 suggests that MiTA generalizes well when scaling up the expert count and width: many configurations exceed the original result. This indicates a promising strategy: training MiTA with smaller m and k for efficiency, then increasing m or k at inference to obtain gains. In Fig. 9, we observe that standard attention and MiTA generalize to each other better than other pairs, especially when training with standard attention and testing with MiTA, which retains over 95% of the original performance with linear complexity.

D Limitations

Our experiments primarily focus on the vision domain, and our current implementation does not address autoregressive decoding in LLMs or the KV cache management. Nevertheless, our motivation and unified perspective are modality-agnostic, and we discuss their potential applicability beyond vision. Extending MiTA to LLMs remains an important direction for future work. We emphasize that the central contribution of this work is a unified perspective on scalable fast weights that enables principled architectural improvements, rather than a vision-specific implementation.



(a) MiTA-ViT-T @512 (b) MiTA-ViT-L @224

Figure 10: MiTA's generalization across m and k. The red box marks the training (or finetuning) (m, k) and the corresponding baseline accuracy on ImageNet-1K. The orange boundary marks the inference (m, k) configurations that exceed the baseline accuracy, and the blue boundary marks those that achieve 99% of the baseline accuracy. Best viewed when zoomed in.

NeurIPS Paper Checklist

1. **Claims**

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

   Answer: [Yes]

   Justification: The main claims are well supported by evidence from other sections.

   Guidelines:

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

2. **Limitations**

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

   Answer: [Yes]

   Justification: We discuss the limitations of this work in Appendix D.

   Guidelines:

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

3. **Theory assumptions and proofs**

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

   Answer: [N/A]

Justification: The paper does not include theoretical results.

Guidelines:

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

4. **Experimental result reproducibility**

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

   Answer: [Yes]

   Justification: We provide detailed implementation settings for all experiments.

   Guidelines:

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

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

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

Answer: [Yes]

Justification: Our code and models are available at https://github.com/QishuaiWen/MiTA.

Guidelines:

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

6. **Experimental setting/details**

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

   Answer: [Yes]

   Justification: We provide detailed implementation settings for all experiments.

   Guidelines:

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

7. **Experiment statistical significance**

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

   Answer: [No]

   Justification: Error bars are not reported because it would be too computationally expensive.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - The authors should answer [Yes] if the results are accompanied by error bars, confidence intervals, or statistical significance tests, at least for the experiments that support the main claims of the paper.
   - The factors of variability that the error bars are capturing should be clearly stated (for example, train/test split, initialization, random drawing of some parameter, or overall run with given experimental conditions).
   - The method for calculating the error bars should be explained (closed form formula, call to a library function, bootstrap, etc.)
   - The assumptions made should be given (e.g., Normally distributed errors).
   - It should be clear whether the error bar is the standard deviation or the standard error of the mean.

- It is OK to report 1-sigma error bars, but one should state it. The authors should preferably report a 2-sigma error bar than state that they have a 96% CI, if the hypothesis of Normality of errors is not verified.
- For asymmetric distributions, the authors should be careful not to show in tables or figures symmetric error bars that would yield results that are out of range (e.g., negative error rates).
- If error bars are reported in tables or plots, the authors should explain in the text how they were calculated and reference the corresponding figures or tables in the text.

8. **Experiments compute resources**

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

   Answer: [Yes]

   Justification: We provide detailed implementation settings for all experiments.

   Guidelines:

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

9. **Code of ethics**

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

   Answer: [Yes]

   Justification: The research conducted in the paper conforms, in every respect, with the NeurIPS Code of Ethics.

   Guidelines:

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

10. **Broader impacts**

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

    Answer: [N/A]

    Justification: This work focuses on general deep learning methodology, and we do not identify immediate broader societal impacts.

    Guidelines:

    - The answer [N/A] means that there is no societal impact of the work performed.
    - If the authors answer [N/A] or [No], they should explain why their work has no societal impact or why the paper does not address societal impact.
    - Examples of negative societal impacts include potential malicious or unintended uses (e.g., disinformation, generating fake profiles, surveillance), fairness considerations (e.g., deployment of technologies that could make decisions that unfairly impact specific groups), privacy considerations, and security considerations.

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

11. **Safeguards**

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

    Answer: [N/A]

    Justification: All experiments are conducted on publicly available datasets.

    Guidelines:

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

12. **Licenses for existing assets**

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

    Answer: [Yes]

    Justification: All experiments are conducted on publicly available datasets.

    Guidelines:

    - The answer [N/A] means that the paper does not use existing assets.
    - The authors should cite the original paper that produced the code package or dataset.
    - The authors should state which version of the asset is used and, if possible, include a URL.
    - The name of the license (e.g., CC-BY 4.0) should be included for each asset.
    - For scraped data from a particular source (e.g., website), the copyright and terms of service of that source should be provided.
    - If assets are released, the license, copyright information, and terms of use in the package should be provided. For popular datasets, `paperswithcode.com/datasets` has curated licenses for some datasets. Their licensing guide can help determine the license of a dataset.
    - For existing datasets that are re-packaged, both the original license and the license of the derived asset (if it has changed) should be provided.

- If this information is not available online, the authors are encouraged to reach out to the asset's creators.

13. **New assets**

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

    Answer: [Yes]

    Justification: Our code and models are available at https://github.com/QishuaiWen/MiTA.

    Guidelines:

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

14. **Crowdsourcing and research with human subjects**

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

    Answer: [N/A]

    Justification: The paper does not involve crowdsourcing or research with human subjects.

    Guidelines:

    - The answer [N/A] means that the paper does not involve crowdsourcing or research with human subjects.
    - Including this information in the supplemental material is fine, but if the main contribution of the paper involves human subjects, then as much detail as possible should be included in the main paper.
    - According to the NeurIPS Code of Ethics, workers involved in data collection, curation, or other labor should be paid at least the minimum wage in the country of the data collector.

15. **Institutional review board (IRB) approvals or equivalent for research with human subjects**

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

    Answer: [N/A]

    Justification: The paper does not involve crowdsourcing or research with human subjects.

    Guidelines:

    - The answer [N/A] means that the paper does not involve crowdsourcing or research with human subjects.
    - Depending on the country in which research is conducted, IRB approval (or equivalent) may be required for any human subjects research. If you obtained IRB approval, you should clearly state this in the paper.
    - We recognize that the procedures for this may vary significantly between institutions and locations, and we expect authors to adhere to the NeurIPS Code of Ethics and the guidelines for their institution.
    - For initial submissions, do not include any information that would break anonymity (if applicable), such as the institution conducting the review.

16. **Declaration of LLM usage**

Question: Does the paper describe the usage of LLMs if it is an important, original, or non-standard component of the core methods in this research? Note that if the LLM is used only for writing, editing, or formatting purposes and does *not* impact the core methodology, scientific rigor, or originality of the research, declaration is not required.

Answer: [N/A]

Justification: We used LLMs solely for writing assistance and language refinement.

Guidelines:

- The answer [N/A] means that the core method development in this research does not involve LLMs as any important, original, or non-standard components.
- Please refer to our LLM policy in the NeurIPS handbook for what should or should not be described.