MORPHING INTO HYBRID ATTENTION MODELS

Anonymous authors
Paper under double-blind review

ABSTRACT

Hybrid attention models improve long-context efficiency by retaining only a subset of full-attention layers and replacing the remaining layers with linear attention. However, the effectiveness of Transformer-to-hybrid conversion critically depends on which layers preserve full attention. Existing hybrid layer selection methods typically rely on heuristic strategies such as fixed placement patterns or layerwise scoring, implicitly treating layer importance as isolated and overlooking the interdependent layer effect under a global hybrid configuration. In this work, we formulate hybrid layer selection as a budget-constrained subset optimization problem. We further propose **FlashMorph** (**F**ast **LA**yer **S**election for **H**ybrid **MORPH**ing), an effective, efficient and scalable layer selection method for Transformer-to-hybrid conversion. FlashMorph first constructs a morphable model by equipping each full-attention layer with a converted linear-attention branch. It then freezes all model weights and jointly optimizes layerwise gates on synthetic long-context retrieval data, with a linearization regularization that encourages the model to rely on linear attention for efficiency. The learned gates are discretized under a preset full-attention budget to instantiate the hybrid architecture, followed by standard logits distillation and long-context finetuning. Extensive experiments show that FlashMorph discovers more effective hybrid configurations, preserves strong long-context recall and general benchmark performance while substantially reducing layer selection cost compared with existing layer selection methods, demonstrating its effectiveness, efficiency, and scalability.

1 INTRODUCTION

The Transformer architecture (Vaswani et al., 2017) has become the dominant backbone of modern large language models (LLMs), driving substantial progress in sequence modeling and complex reasoning (Grattafiori et al., 2024; Cai et al., 2024; Liu et al., 2024; Guo et al., 2025; Yang et al., 2025). Nevertheless, its reliance on softmax attention introduces a fundamental efficiency bottleneck: increasing sequence length leads to quadratic growth in attention computation and linear growth in the Key-Value (KV) cache required for autoregressive inference (Kwon et al., 2023; Sun et al., 2025a). These limitations have motivated the development of more efficient sequence mixers, such as linear attention (Katharopoulos et al., 2020; Yang et al., 2023; Qin et al., 2024a;b; Yang et al., 2024b;a) and state-space models (Gu & Dao, 2023; Dao & Gu, 2024; Lahoti et al., 2026; Hu et al., 2025), which reduce the computational cost of long-sequence modeling and eliminate the need for a growing KV cache by maintaining fixed-size recurrent states. However, purely linear recurrent architectures are generally less effective than Transformer-based LLMs on long-context and recall-sensitive tasks (Arora et al., 2024; Wen et al., 2025). This performance gap has motivated hybrid attention architectures, which retain full attention in a subset of layers while replacing the remaining layers with efficient linear sequence mixers, achieving a more favorable trade-off between model quality and efficiency (Chen et al., 2025; Team et al., 2025; Merrill et al., 2026).

However, training high-quality hybrid LLMs from scratch remains prohibitively expensive. A more practical alternative is **Transformer-to-hybrid Conversion**, which starts from a pretrained Transformer-based LLM, retains only a small subset of its full-attention layers, and replaces the remaining layers with efficient linear attention through model parameter transfer, distillation and continued finetuning (Kasai et al., 2021; Mercat et al., 2024; Wang et al., 2024; Bick et al., 2024; Zhang et al., 2024a; Lan et al., 2025). By reusing the weights of a strong Transformer model, this paradigm avoids the significant cost of training a hybrid architecture from scratch while preserving

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.

much of the original model capability. Nevertheless, its effectiveness critically depends on how the limited budget of retained full attention is allocated across layers.

In principle, given an L-layer model and a budget of K retained full-attention layers, identifying the optimal hybrid configuration requires evaluating all $\binom{L}{K}$ possible subsets, which quickly becomes intractable (Li et al., 2025). Existing hybrid layer selection methods can therefore be understood as heuristic, tractable approximations to this combinatorial subset-selection problem. Uniform interleaving avoids search altogether by imposing fixed attention placement rules, but it ignores the pretrained model and the heterogeneous functional roles of different layers. Search-based methods explore layer placements through auxiliary architecture search or supernet training, but they introduce substantial optimization and evaluation overhead (Gu et al., 2025). Layerwise methods estimate the marginal utility of each layer by perturbing, replacing, or restoring one layer at a time, and then select layers according to the resulting scores (Yang et al., 2026; Li et al., 2025; Chen et al., 2026). More fundamentally, these heuristic approximations reduce hybrid layer selection to either fixed attention placement rules or isolated layer scoring, implicitly assuming that the effect of a multi-layer hybrid configuration can be inferred from individual layer importance. This assumption overlooks interdependent layer effects and fails to capture the joint effects that emerge when multiple layers are converted together. Moreover, layerwise methods require repeated one-layer perturbation or restoration to estimate layer importance, incurring substantial selection overhead and making them costly to scale. This motivates the central question studied in this work:

> *Can hybrid layer selection be formulated as a budget-constrained joint optimization problem that globally accounts for interdependent layer effect, rather than relying on heuristic approximations such as fixed attention placement rules or isolated layerwise scoring?*

We argue that hybrid layer selection should be treated as a subset selection problem under a global hybrid configuration. In this setting, the contribution of each layer depends on which other layers are retained or converted: jointly converting layers that support related functions may amplify degradation, whereas retaining layers with overlapping roles may yield diminishing returns. The objective is to identify a subset of full-attention layers whose collective effect best balances model quality and efficiency, while accounting for complementarity and redundancy across layers.

Motivated by this, we propose **FlashMorph** (**F**ast **LA**yer **S**election for **H**ybrid **MORPH**ing), an effective, efficient, and scalable layer selection method for Transformer-to-hybrid conversion. FlashMorph first constructs a morphable model, in which each pretrained full-attention layer is equipped with a converted linear-attention branch and can be continuously transformed between full and linear attention by introducing layerwise learnable gates. During layer selection, both the pretrained backbone and the converted linear-attention branches are kept frozen, while only the layerwise gates are optimized on synthetic retrieval data. By jointly optimizing all gates, FlashMorph estimates the relative necessity of retaining full attention under a global hybrid configuration, rather than relying on isolated layer scoring. The optimized gates are then discretized according to a prescribed full-attention budget to instantiate a hybrid architecture, which is subsequently trained with standard distillation and long-context fine-tuning. As summarized in Table 1, FlashMorph performs non-isolated layer selection that captures inter-layer complementarity and redundancy through joint optimization, and requires substantially fewer selection tokens than prior methods, thereby yielding a stronger quality-efficiency trade-off for Transformer-to-hybrid conversion.

Table 1: **Comparison of Hybrid Layer Selection (LS) Methods.** LS cost refers to the required tokens for layer selection. ✓, ✗, and △indicate fully supported, not supported, and partially supported, respectively.

LS method	LS cost ↓	Non-isolated	Optimization-based
Uniform	N/A	✗	✗
PostNAS	50B	△	✓
KL-LS	20B	✗	✓
HALO	234M	✗	✗
FlashMorph (ours)	**20M**	✓	✓

To summarize, our contributions are as follows:

- We formulate hybrid layer selection as a budget-constrained joint optimization problem that explicitly captures the interdependence and complementarity among retained full-attention layers.
- We propose FlashMorph, an effective, efficient, and scalable layer selection method that recasts hybrid layer selection as budget-constrained joint optimization, enabling globally coordinated allocation of full-attention layers.
- We conduct comprehensive experiments across Qwen3 backbones and multiple linear-attention variants, demonstrating that FlashMorph preserves strong long-context retrieval, commonsense reasoning and recall performance while substantially reducing layer-selection cost.

2 PRELIMINARIES

Notation. Let $\mathbf{X} = [\mathbf{x}_1; \mathbf{x}_2; \ldots; \mathbf{x}_T] \in \mathbb{R}^{T\times d}$ be an input sequence with the length of T, where $\mathbf{x}_t \in \mathbb{R}^{1\times d}$ is the token representation at position t with the dimension of d. For simplicity, we omit multi-head notation and write all equations for a single attention head. The query, key, and value vectors are computed as

$$\mathbf{q}_t = \mathbf{x}_t\mathbf{W}_Q, \quad \mathbf{k}_t = \mathbf{x}_t\mathbf{W}_K, \quad \mathbf{v}_t = \mathbf{x}_t\mathbf{W}_V, \tag{1}$$

where $\mathbf{q}_t, \mathbf{k}_t, \mathbf{v}_t \in \mathbb{R}^{1\times d}$, and $\mathbf{W}_Q, \mathbf{W}_K, \mathbf{W}_V \in \mathbb{R}^{d\times d}$ are learnable projection matrices.

Full Attention. Full (softmax) attention computes the output at each position by comparing the current query with all previous keys and taking a weighted sum over the corresponding values. Under causal masking, the attention output at position t is

$$\mathbf{o}_t = \sum_{i=1}^{t} \alpha_{t,i}\mathbf{v}_i\mathbf{W}_O, \quad \alpha_{t,i} = \frac{\exp\left(\mathbf{q}_t\mathbf{k}_i^\top/\sqrt{d}\right)}{\sum_{j=1}^{t}\exp\left(\mathbf{q}_t\mathbf{k}_j^\top/\sqrt{d}\right)}, \tag{2}$$

where $\mathbf{W}_O \in \mathbb{R}^{d\times d}$ is the output projection. Because each query attends to all previous keys, full attention explicitly models pairwise token interactions. This gives it strong capacity for precise matching and long-range dependency modeling, but also leads to $\mathcal{O}(T^2)$ computation over the sequence length and KV cache whose memory grows linearly with the length during AR inference.

Linear Attention. Linear attention replaces the softmax kernel with a formulation that can be accumulated recurrently. Let $\phi(\cdot)$ be a feature map applied to queries and keys, a generic kernelized linear attention output can be written as

$$\mathbf{o}_t = \frac{\phi(\mathbf{q}_t)\sum_{i=1}^{t}\phi(\mathbf{k}_i)^\top\mathbf{v}_i}{\phi(\mathbf{q}_t)\sum_{i=1}^{t}\phi(\mathbf{k}_i)^\top}\mathbf{W}_O. \tag{3}$$

In many modern linear attention variants, the normalization term is omitted, and the key-value statistics are maintained as a recurrent memory state. After absorbing the feature map into $\mathbf{q}_t$ and $\mathbf{k}_t$, a broad class of linear attention sequence mixers can be expressed as RNN style

$$\mathbf{S}_t = \mathbf{A}_t\mathbf{S}_{t-1} + \mathbf{k}_t^\top\mathbf{v}_t, \quad \mathbf{o}_t = \mathbf{q}_t\mathbf{S}_t\mathbf{W}_O, \tag{4}$$

where $\mathbf{S}_t \in \mathbb{R}^{d\times d}$ is the recurrent memory state and $\mathbf{A}_t \in \mathbb{R}^{d\times d}$ is a state-transition or decay matrix. Unlike full attention, the recurrent formulation avoids storing all previous keys and values, enabling $\mathcal{O}(T)$ linear-time sequence processing and constant-size state caching during AR decoding.

Hybrid Attention. While linear attention is efficient, purely linear models may lose part of the long-range dependency modeling and retrieval ability of full attention. Hybrid attention mitigates this trade-off by retaining full attention in a subset of layers and replacing the remaining layers with linear attention. For an L-layer model, let $\mathcal{I}_{\text{full}} \subseteq \{1, \ldots, L\}$ denote the set of layers that use full attention. The sequence mixer in layer l is defined as

$$\text{Mixer}^{(l)} = \begin{cases} \text{FullAttn}^{(l)} = A_{\text{full}}^{(l)}, & l \in \mathcal{I}_{\text{full}}, \\ \text{LinearAttn}^{(l)} = A_{\text{lin}}^{(l)}, & l \notin \mathcal{I}_{\text{full}}. \end{cases} \tag{5}$$

Therefore, a hybrid Transformer block can then be written as

$$\begin{aligned} \mathbf{H}^{(l)} &= \mathbf{X}^{(l)} + \text{Mixer}^{(l)}\left(\text{LN}(\mathbf{X}^{(l)})\right), \\ \mathbf{X}^{(l+1)} &= \mathbf{H}^{(l)} + \text{FFN}^{(l)}\left(\text{LN}(\mathbf{H}^{(l)})\right). \end{aligned} \tag{6}$$

Here, the retained set $\mathcal{I}_{\text{full}}$ determines the core efficiency–effectiveness trade-off of the hybrid attention model. Keeping more full-attention layers preserves the retrieval and precise matching ability of the original Transformer, but also increases the computational and memory cost. Conversely, converting more layers to linear attention improves efficiency, but may degrade performance if critical layers are replaced. Therefore, under a fixed full-attention budget $K = |\mathcal{I}_{\text{full}}|$, the central question is not only how to train the linear-attention replacements, but also which layers should retain full attention, which is the focus of the next section.

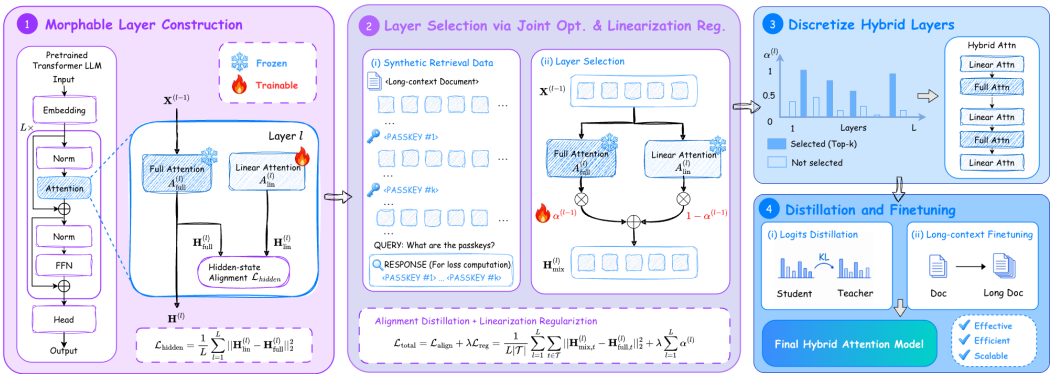


Figure 1: **Overview of FlashMorph.** FlashMorph constructs morphable layers through hidden-state alignment, then jointly optimizes layerwise gates with linearization regularization, and discretizes the learned gates into a hybrid architecture for subsequent distillation and long-context finetuning.

3 Morphing into Hybrid Attention Models

In this section, we introduce **FlashMorph** (**F**ast **LA**yer **S**election for **H**ybrid **MORPH**ing), an effective, efficient, and scalable layer selection method for Transformer-to-hybrid conversion. As discussed in Sec. 3.1, we formulate hybrid layer selection as a budget-constrained subset optimization problem, which motivates us to move beyond fixed allocation patterns and isolated layerwise scoring. As illustrated in Fig. 1, FlashMorph first constructs morphable layers by distilling an all-linear model from the original Transformer through hidden-state alignment, thereby equipping each full-attention layer with a trained linear-attention replacement, as described in Sec. 3.2. It then performs layer selection by jointly optimizing layerwise gate values on synthetic long-context retrieval data and discretizing the resulting gates into a hybrid attention model, as detailed in Sec. 3.3. In Sec. 3.4 we finally apply logits distillation and long-context finetuning to recover the quality of the hybrid model.

3.1 Rethinking Layer Selection in Hybrid Attention Model

Given a pretrained L-layer Transformer model, Transformer-to-hybrid conversion keeps only a subset of layers as full attention and converts the remaining layers to linear attention. Let $[L] = \{1, \dots, L\}$, and let $\mathcal{I}_{\text{full}} \subseteq [L]$ denote the retained full-attention layers. The corresponding hybrid model is denoted by $\mathcal{M}(\mathcal{I}_{\text{full}})$. Under a full-attention budget K, the ideal layer selection objective is

$$\mathcal{I}^{\star}_{\text{full}} = \arg \max_{\substack{\mathcal{I}_{\text{full}} \subseteq [L] \\ |\mathcal{I}_{\text{full}}| = K}} \text{Score}\left(\mathcal{M}(\mathcal{I}_{\text{full}})\right), \tag{7}$$

where $\text{Score}(\cdot)$ denotes a higher-is-better performance metric. This formulation shows that hybrid layer selection is inherently a subset optimization problem. Solving Eq. 7 exactly requires evaluating $\binom{L}{K}$ possible subsets, which is computationally infeasible for modern LLMs.

Existing methods therefore rely on tractable approximations. Uniform interleaving (Wang et al., 2024; Paliotta et al., 2025; Lan et al., 2025; Goldstein et al., 2025) imposes a fixed handcrafted pattern, while layerwise estimation (Yang et al., 2026; Li et al., 2025; Chen et al., 2026) follows a cumbersome layer-by-layer procedure, where each layer is independently replaced, restored, or scored before the top-ranked layers are selected under the budget. This isolated evaluation is not only inefficient, but also implicitly treats layer importance as an isolated property while ignoring inter-layer dependencies and redundancy, causing the selected layers to be suboptimal when deployed as a joint hybrid configuration. We next introduce our proposed method FlashMorph for joint optimization-based hybrid layer selection.

3.2 Morphable Layers Construction

We first construct a morphable model by equipping each full-attention layer with a converted linear-attention branch. To obtain these replacement branches, we follow prior Transformer-to-hybrid conversion pipelines (Goldstein et al., 2025; Li et al., 2025; Chen et al., 2026), where the pretrained full-attention model $\mathcal{M}_{\text{full}}$ serves as the teacher and the linear-attention branches are trained to imitate its layerwise representations.

During this stage, all parameters of the original full-attention model are frozen, and only the parameters of the linear-attention branches are updated. Let $\mathbf{H}_{\text{full}}^{(l)}$ and $\mathbf{H}_{\text{lin}}^{(l)}$ denote the output hidden states produced by the full-attention branch and the linear-attention branch at layer l, respectively. We optimize the linear-attention branches using a layerwise hidden-state alignment loss

$$\mathcal{L}_{\text{hidden}} = \frac{1}{L}\sum_{l=1}^{L}\mathcal{L}_{\text{hidden}}^{(l)} = \frac{1}{L}\sum_{l=1}^{L}\left\|\mathbf{H}_{\text{lin}}^{(l)} - \mathbf{H}_{\text{full}}^{(l)}\right\|_2^2. \tag{8}$$

This stage yields a trained all-linear student $\mathcal{M}_{\text{all-linear}}$, which provides a linear-attention replacement for each full-attention layer in the original model. We then construct a morphable model by pairing these trained linear-attention replacements with the frozen full-attention layers of the pretrained model, enabling arbitrary full/linear layer configurations for subsequent layer selection stage.

3.3 Layer Selection via Joint Optimization and Linearization Regularization

Optimization-based layer selection. Given the morphable model constructed above, layer selection aims to identify a subset of layers that should retain full attention under a fixed budget. This is a combinatorial subset-selection problem: for an L-layer model with a budget of K retained full-attention layers, one must choose K layers from L candidates, resulting in $\binom{L}{K}$ possible hybrid configurations. Exhaustively evaluating these configurations is intractable. To avoid isolated layerwise scoring, FlashMorph relaxes the discrete subset-selection problem into a continuous optimization problem over layerwise gates, allowing all layers to be optimized jointly under a global hybrid configuration. Specifically, for each layer l, we introduce a scalar gate $\alpha^{(l)} \in [0, 1]$ that interpolates between the full-attention branch and the linear-attention branch:

$$\mathbf{H}_{\text{mix}}^{(l)} = \alpha^{(l)}\mathbf{H}_{\text{full}}^{(l)} + (1 - \alpha^{(l)})\mathbf{H}_{\text{lin}}^{(l)}. \tag{9}$$

A larger $\alpha^{(l)}$ indicates stronger reliance on the full-attention branch, whereas a smaller $\alpha^{(l)}$ suggests that the layer can be more safely linearized. We initialize all gates as $\alpha^{(l)} = 1$, corresponding to the original full-attention model. During layer selection stage, both the full-attention backbone and the trained linear-attention branches are frozen, and only the gate values $\boldsymbol{\alpha} = \{\alpha^{(l)}\}_{l=1}^{L}$ are optimized. This keeps the number of trainable parameters extremely small and prevents the selection stage from adapting model weights to compensate for a poor architectural choice.

To preserve the behavior of the original full-attention model, we optimize the gates by aligning the hidden states of the morphed model with those of the full-attention teacher. Let $\{\mathbf{H}_{\text{full}}^{(l)}\}_{l=1}^{L}$ denote the hidden states of the original full-attention teacher, and let $\{\mathbf{H}_{\text{mix}}^{(l)}\}_{l=1}^{L}$ denote the corresponding hidden states of the morphed model. We compute the alignment loss at the answer-token positions:

$$\mathcal{L}_{\text{align}} = \frac{1}{L|\mathcal{T}(x)|}\sum_{l=1}^{L}\sum_{t\in\mathcal{T}(x)}\left\|\mathbf{H}_{\text{mix},t}^{(l)} - \mathbf{H}_{\text{full},t}^{(l)}\right\|_2^2, \tag{10}$$

where $\mathcal{T}(x)$ denotes the set of answer-token positions. To encourage the model to rely on linear attention whenever possible for efficiency, we further introduce linearization regularization:

$$\mathcal{L}_{\text{reg}} = \sum_{l=1}^{L}\alpha^{(l)}. \tag{11}$$

The final optimization objective for layer selection is therefore defined as

$$\mathcal{L}_{\text{total}} = \mathcal{L}_{\text{align}} + \lambda\mathcal{L}_{\text{reg}}, \tag{12}$$

where λ is a hyperparameter (set to 0.1 in our default implementation) controlling the strength of linearization regularization. The alignment term preserves the behavior of the full-attention teacher, while the regularization term penalizes reliance on full attention and pushes layers toward their linear-attention branches whenever this does not significantly change the teacher output. Since all gate values are optimized simultaneously, this objective enables FlashMorph to perform layer selection via joint optimization under a global hybrid configuration rather than scoring each layer in isolation. The resulting gates therefore reflect the relative necessity of retaining full attention while accounting for inter-layer dependency, redundancy, and complementarity.

Synthetic retrieval data. Generic language modeling objectives are often dominated by local dependencies and may provide weak supervision for identifying layers that are critical for long-context retrieval. We therefore perform layer selection on synthetic long-context retrieval examples. In each example, randomly generated passkeys are inserted at different depths of a long-context document, and the model is required to recover the corresponding passkeys at the end of the sequence. Similar synthetic retrieval signals have been used to identify long-context-critical attention components (Xiao et al., 2025; Lin et al., 2026; Bick et al., 2026). Implementation details are in Appendix. B

In our setting, the synthetic retrieval data provides a targeted selection signal for measuring whether replacing full attention with linear attention disrupts long-range information access. Importantly, this data is used only to optimize the layer-wise gates rather than the model weights. As a result, the learned gates reflect how much each layer needs to retain full attention when all layers are optimized jointly, thereby capturing inter-layer dependency, redundancy, and complementarity that are overlooked by isolated layerwise estimation.

Discretizing hybrid layers. After optimization, the learned gate values indicate the relative necessity of preserving full attention in each layer. Given a target budget K of full-attention layers, FlashMorph selects the layers as full attention with the largest gate values

$$\mathcal{I}_{\text{full}}^{\text{Hybrid}} = \text{TopK}\left(\{\alpha_l\}_{l=1}^{L}, K\right). \tag{13}$$

The selected layers $\mathcal{I}_{\text{full}}^{\text{Hybrid}}$ keep their original full-attention branches, while the remaining layers $\mathcal{I}_{\text{lin}}^{\text{Hybrid}} = [L] \setminus \mathcal{I}_{\text{full}}^{\text{Hybrid}}$ are instantiated with their trained linear-attention replacements. After this discretization step, the gate values are discarded, and the resulting architecture $\mathcal{M}(\mathcal{I}_{\text{full}}^{\text{Hybrid}})$ becomes a hybrid attention model with no additional overhead at inference time.

3.4 Distillation and Long-context Finetuning

After layer selection stage, we obtain a hybrid attention model $\mathcal{M}_{\text{Hybrid}} = \mathcal{M}(\mathcal{I}_{\text{full}}^{\text{Hybrid}})$. Following prior Transformer-to-hybrid conversion pipelines (Goldstein et al., 2025; Li et al., 2025; Chen et al., 2026), we further apply logits distillation and long-context finetuning to recover the quality of the selected hybrid attention model.

Logits distillation. We distill the hybrid attention model from the original full-attention teacher. Let $p_{\text{T}}(\cdot)$ and $p_{\text{H}}(\cdot)$ denote the output logits generated by the full-attention teacher and the hybrid attention model, respectively. The distillation objective optimized by the Kullback-Leibler (KL) divergence D_{KL} is defined as

$$\mathcal{L}_{\text{KD}} = D_{\text{KL}}(p_{\text{T}}(\mathbf{X}) \parallel p_{\text{H}}(\mathbf{X})), \tag{14}$$

Long-context finetuning. We then finetune the hybrid model on long-context sequences with the standard language modeling objective

$$\mathcal{L}_{\text{FT}} = -\sum \log p_{\text{H}}(x_t \mid x_{<t}). \tag{15}$$

This completes the FlashMorph Transformer-to-hybrid conversion pipeline. The full procedure is summarized in Algorithm 1.

4 Experiments

4.1 Experiment Setup

Baselines. We compare FlashMorph with representative hybrid layer selection methods, including uniform interleaving, PostNAS (Gu et al., 2025), KL-LS (Li et al., 2025), and HALO (Chen et al., 2026). Uniform interleaving retains full-attention layers according to a fixed periodic pattern, while PostNAS, KL-LS, and HALO represent data-driven selection strategies based on supernet training and searching, KL-guided scoring, or layerwise replacement evaluation.

Training. We use Qwen3-0.6B and Qwen3-1.7B as the pretrained full-attention Transformer backbones. Throughout our experiments, we follow the HALO Transformer-to-hybrid conversion pipeline (Chen et al., 2026). Specifically, the pipeline first trains linear-attention replacement branches through hidden-state alignment, then applies layer selection to determine which layers retain full attention, and finally recovers the selected hybrid model through logits distillation and long-context finetuning. To isolate the effect of layer selection, we keep the model architecture,

Table 2: **NIAH Performance across 0.6B and 1.7B Backbones from 32K to 256K Context Lengths.** * indicates selected layers taken from (Chen et al., 2026). The best results are in **bold**, and the second-best results are underlined.

Model	LS. Tokens	NIAH-Single-1				NIAH-Single-2				NIAH-Single-3			
		32K	64K	128K	256K	32K	64K	128K	256K	32K	64K	128K	256K
0.6B backbone													
Qwen3	-	100	100	0	0	100	99.0	0	0	99.8	82.8	0	0
Qwen3+YaRN	-	30.2	29.2	20.2	31.0	0.6	0	0	0	0.8	0	0	0
Uniform	N/A	99.4	98.6	97.8	98.0	68.4	49.2	41.2	12.8	57.6	36.0	28.8	12.2
KL-LS	20B	98.2	97.6	96.4	94.8	69.4	60.6	50.2	**24.0**	32.2	17.2	8.6	3.6
HALO	234M	**99.6**	**99.8**	**99.6**	**99.2**	86.0	80.6	**62.4**	21.0	68.6	67.4	**57.2**	**32.4**
FlashMorph	**20M**	99.0	99.0	99.0	**99.2**	**92.2**	**82.4**	45.8	11.0	**81.2**	**73.6**	45.6	28.4
1.7B backbone													
Qwen3	-	100	100	0	0	100	98.8	0	0	99.8	96.4	0	0
Qwen3+YaRN	-	63.8	17.0	3.8	2.8	31.8	8.8	1.8	10.2	7.8	1.0	0.8	0.2
Uniform	N/A	99.8	99.6	99.6	**100**	71.8	86.8	28.4	19.2	58.6	59.4	16.4	27.8
PostNAS*	50B	99.2	99.4	99.8	99.2	96.8	95.6	78.0	73.8	56.4	51.2	57.2	57.6
KL-LS	20B	98.6	98.6	98.0	94.4	62.2	68.6	47.4	34.6	22.8	4.0	9.4	3.8
HALO	234M	99.8	**100**	**100**	**100**	**99.6**	98.6	95.0	**95.2**	86.4	90.8	67.4	52.8
FlashMorph	**20M**	**100**	**100**	**100**	**100**	**99.6**	**100**	**98.2**	88.2	**96.6**	**95.4**	**94.4**	**73.2**

training data and pipeline unchanged across all methods, and replace only the layer selection strategy with FlashMorph or the corresponding baseline methods. All the experiments are conducted on 8× GPUs with BFloat16 precision. Unless otherwise specified, all main comparisons are conducted under the 3:1 hybrid ratio. Our detailed model and training configurations are provided in Appendix A.

Evaluation. We evaluate FlashMorph under two settings. For the Needle-in-a-Haystack (NIAH) task (Hsieh et al., 2024), we follow the HypeNet setting (Chen et al., 2026), which adopts Lightning Attention (Qin et al., 2024a) together with hybrid positional encoding (HyPE). The resulting hybrid model retains the key architectural components used in HypeNet, including attention logits scaling, QK normalization, GQA-to-MHA conversion, and output normalization and gating for linear-attention layers. Unlike the original HypeNet configuration, however, we do not apply gated attention (Qiu et al., 2026) to the retained full-attention layers; instead, these layers remain standard full-attention blocks. For commonsense reasoning and recall-intensive tasks, we adopt the standard RoPE-based setting for all layers. Under this setting, we evaluate three linear-attention backbones: Lightning Attention (Qin et al., 2024a), Gated Linear Attention (GLA) (Yang et al., 2023), and Gated DeltaNet (GDN) (Yang et al., 2024a). Our attention implementations are based on the `flash-linear-attention` library (Yang & Zhang, 2024), and downstream evaluations are conducted using `lm-evaluation-harness` (Gao et al., 2024).

4.2 Main Results

Needle-in-a-Haystack. Table 2 reports the NIAH results on the 0.6B and 1.7B backbones across three retrieval variants and context lengths from 32K to 256K. The original Qwen3 models perform well at short context lengths but quickly collapse as the context is extended, while directly applying YaRN (Peng et al., 2024) fails to consistently restore long-context retrieval ability. Hybrid conversion substantially improves retrieval over extended contexts, but its effectiveness depends strongly on which layers are retained as full attention. On the 0.6B backbone, FlashMorph achieves near-perfect accuracy on NIAH-Single-1 and delivers strong performance on the more challenging NIAH-Single-2 and NIAH-Single-3 settings, particularly at short and medium context lengths. On the 1.7B backbone, the advantage becomes more pronounced: FlashMorph maintains perfect accuracy on NIAH-Single-1 across all context lengths, and substantially improves NIAH-Single-2 and NIAH-Single-3, where accurate retrieval over long contexts is more challenging. Notably, FlashMorph obtains these results using only 20M layer selection tokens, compared with substantially larger selection budgets required by prior methods, showing that FlashMorph can identify full-attention layers effectively and efficiently while preserving strong long-context retrieval performance.

Commonsense Reasoning and Recall-intensive Tasks. We evaluate zero-shot commonsense reasoning on ARC-Easy/Challenge (Clark et al., 2018), PIQA (Bisk et al., 2020), HellaSwag (Zellers et al., 2019), and WinoGrande (Sakaguchi et al., 2021), as well as recall-intensive tasks on SQuAD (Rajpurkar et al., 2018), FDA (Arora et al., 2023), and SWDE (Lockard et al., 2019). Table 3 shows that FlashMorph remains close to the strongest baselines for commonsense reasoning across attention backbones and model scales. This indicates that the selected hybrid architectures preserve the

Table 3: **Zero-shot Performance on Commonsense Reasoning and Long-context Recall-intensive Tasks across Attention Backbones and Model Scales.** * indicates that the selected layers are derived from (Chen et al., 2026). The best results are in **bold**, and the second-best results are underlined.

Attn.	Method	LS. Tokens	ARC-e acc	ARC-c acc_n	PIQA acc	Hella. acc_n	Wino. acc	Avg.	SQuAD acc	FDA acc	SWDE acc	Avg.
0.6B backbone												
Full	Qwen3	-	60.6	34.1	67.6	47.3	55.7	53.1	44.1	82.1	80.5	68.9
Lightning	Uniform	N/A	62.3	32.9	66.9	46.1	55.6	52.8	30.3	51.3	71.8	51.1
	KL-LS	20B	62.1	**33.0**	**67.5**	46.2	54.9	52.7	29.3	58.5	70.5	52.8
	HALO	234M	**63.5**	32.4	67.4	**46.4**	**56.1**	**53.2**	34.7	60.4	72.6	55.9
	FlashMorph	**20M**	62.8	32.0	67.3	**46.4**	55.2	52.7	**41.7**	**62.4**	**76.2**	**60.1**
GLA	Uniform	N/A	63.1	33.1	67.3	46.8	**55.3**	**53.1**	31.1	55.6	75.0	53.9
	KL-LS	20B	61.8	**33.8**	67.2	**46.9**	54.9	52.9	32.7	61.4	75.2	56.4
	HALO	234M	62.9	32.6	67.4	**46.9**	55.1	53.0	**36.4**	68.5	75.4	60.1
	FlashMorph	**20M**	**63.9**	32.9	**67.5**	**46.9**	54.1	**53.1**	35.4	**70.7**	**76.0**	**60.7**
GDN	Uniform	N/A	59.6	32.3	67.1	**47.6**	55.3	52.4	30.3	55.5	72.2	52.7
	KL-LS	20B	61.1	**35.3**	**67.9**	47.4	**56.7**	**53.7**	33.5	**72.8**	75.6	60.6
	HALO	234M	60.1	34.1	67.7	47.5	55.3	53.0	26.5	62.0	71.2	53.2
	FlashMorph	**20M**	**63.1**	33.5	67.8	47.5	56.4	**53.7**	**38.4**	71.3	**76.7**	**62.1**
1.7B backbone												
Full	Qwen3	-	72.4	43.5	72.5	60.4	61.0	62.0	39.8	79.0	85.1	67.9
Lightning	Uniform	N/A	**74.7**	**45.1**	72.3	60.5	**62.2**	**62.9**	39.0	56.6	78.6	58.1
	PostNAS*	50B	**74.7**	43.0	72.9	**60.8**	60.9	62.5	**51.3**	64.7	80.9	65.7
	KL-LS	20B	74.2	42.8	72.3	60.4	59.1	61.8	39.5	49.4	74.8	54.5
	HALO	234M	73.5	42.8	72.6	60.6	61.5	62.2	**51.3**	70.5	**82.5**	**68.1**
	FlashMorph	**20M**	73.2	42.3	**73.2**	60.6	61.0	62.1	51.1	**71.6**	81.6	**68.1**
GLA	Uniform	N/A	74.4	45.5	72.7	60.8	**63.0**	**63.3**	43.6	57.3	81.2	60.7
	PostNAS*	50B	73.4	43.1	73.2	**61.5**	60.1	62.3	**54.2**	69.4	81.9	**68.5**
	KL-LS	20B	**74.6**	**46.3**	72.4	61.0	59.8	62.8	44.1	53.5	78.6	58.7
	HALO	234M	74.4	44.5	72.6	60.9	62.8	63.0	43.7	41.4	74.2	53.1
	FlashMorph	**20M**	74.5	44.7	**73.3**	61.2	62.3	63.2	47.3	**73.7**	**82.3**	67.7
GDN	Uniform	N/A	74.3	44.5	72.3	61.5	62.4	63.0	48.2	59.2	78.9	62.1
	PostNAS*	50B	75.1	45.1	73.1	**61.7**	62.3	63.4	52.8	67.9	**82.4**	67.7
	KL-LS	20B	74.3	**47.6**	72.9	61.6	62.3	63.7	54.1	73.5	81.2	69.6
	HALO	234M	**75.8**	45.7	73.2	61.1	63.1	**63.8**	**54.5**	68.2	80.9	67.9
	FlashMorph	**20M**	74.4	44.1	**73.3**	61.5	**63.9**	63.4	54.3	**74.1**	82.2	**70.2**

general reasoning capabilities of the original Transformer. Its advantage is more pronounced on recall-intensive tasks: FlashMorph achieves the highest 0.6B recall average across all backbones and the best or tied-best 1.7B average with Lightning Attention and GDN. It also remains highly competitive with GLA, reaching 67.7 versus 68.5 for PostNAS while using orders of magnitude fewer layer-selection tokens. Overall, these results show that FlashMorph preserves strong recall while maintaining general capability.

4.3 Efficiency Results

Prefilling and Decoding. We compare FlashMorph (linear:full = 3:1) with purely full-attention Qwen3 on the 1.7B backbone using a single GPU and batch size 1. Figure 2 reports latency and peak GPU memory as sequence length increases. For prefilling, FlashMorph reaches 2.24× and 2.81× speedups at 128K and 256K tokens, respectively, and enables 512K-token prefilling where Qwen3-1.7B runs out of memory. For decoding with a 1K-token prompt, it achieves 1.56× and 2.07× speedups at 256K and 512K tokens and remains executable at 1M tokens, unlike Qwen3-1.7B.

Layer Selection. Table 4 compares selection cost on Qwen3-1.7B. FlashMorph uses only 20M selection tokens, 2.5×10^{17} FLOPs, and 2.1 GPU hours, compared with 234M–50B tokens for the baselines. It reduces GPU hours by 7.3×, 510.4×, and 1219.7× relative to HALO, KL-LS, and PostNAS, respectively. Figure 2 further shows that FlashMorph consistently has the lowest FLOPs and GPU hours across model sizes. Unlike KL-LS and HALO, which rely on repeated per-layer restoration or evaluation, FlashMorph jointly optimizes gates under a global hybrid configuration, yielding a lightweight and scalable selection stage.

Table 4: **Efficiency Comparison of Layer Selection Methods**. *The result is calculated from (Gu et al., 2025).

LS Method	Tokens ↓	FLOPs ↓	GPU hours ↓
PostNAS*	50B	8.0e20	2561.3
KL-LS	20B	2.5e20	1071.8
HALO	234M	6.5e17	15.4
FlashMorph	**20M**	**2.5e17**	**2.1**

4.4 Analysis

Robustness across Hybrid Ratios. On Qwen3-1.7B, we evaluate linear:full ratios of 6:1, 3:1, and 1:1, while keeping the post-selection pipeline fixed to isolate layer selection. As shown in Fig. 3a,

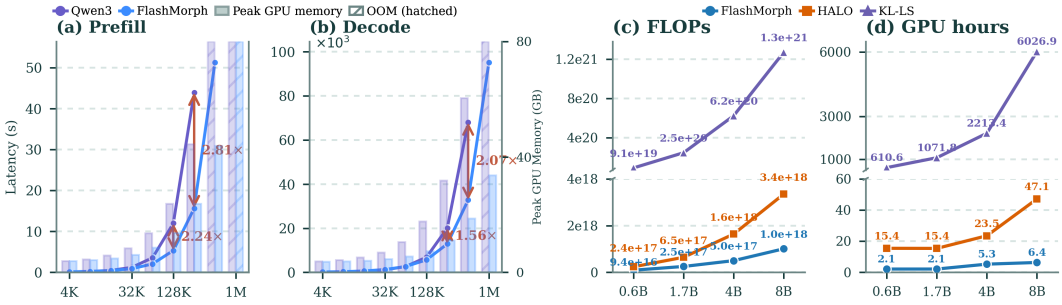
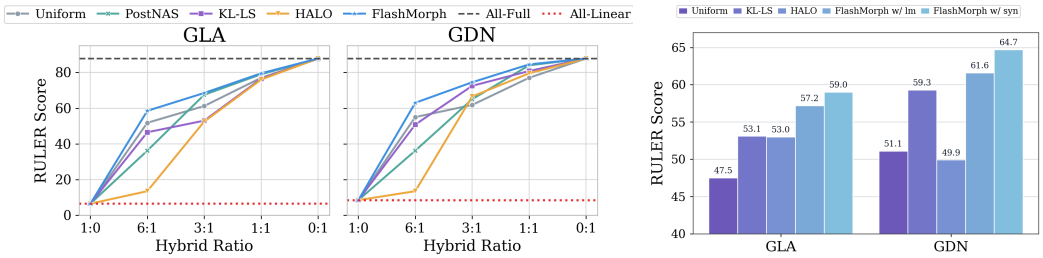


Figure 2: **Efficiency of FlashMorph.** (a) Prefilling and (b) decoding latency (lines, left axes) and peak GPU memory (bars, right axes) of FlashMorph and Qwen3-1.7B under increasing sequence lengths; hatched bars denote out-of-memory (OOM). (c) Layer-selection FLOPs and (d) GPU hours across model sizes. FlashMorph consistently requires substantially less cost than HALO and KL-LS.



(a) RULER performance under different hybrid ratios.

(b) Effect of layer selection supervision.

Figure 3: **Analysis of FlashMorph under Different Hybrid Configurations and Supervision Signals.** (a) FlashMorph consistently achieves strong performance compared with prior layer selection methods across hybrid ratios. (b) FlashMorph with language-modeling supervision already outperforms prior layer selection methods, while synthetic passkey-based supervision further improves performance on both GLA and GDN backbones.

FlashMorph achieves the best or among the strongest RULER (Hsieh et al., 2024) performance on both GLA and GDN across ratios. Its advantage is largest at 6:1, where full-attention layers are scarce. As the budget increases, all methods approach the all-full upper bound, while FlashMorph remains competitive, demonstrating robust selection across efficiency–performance trade-offs.

Comparison of Layer Selection Supervision. On Qwen3-0.6B, we compare generic language-modeling supervision with our synthetic passkey retrieval supervision. Figure 3b shows that FlashMorph with language-modeling supervision already outperforms KL-LS and HALO on both GLA and GDN. This indicates that the proposed joint optimization can identify stronger hybrid configurations even without specialized retrieval-oriented data. Synthetic retrieval supervision further improves RULER from 57.2 to 59.0 on GLA and from 61.6 to 64.7 on GDN, indicating that retrieval-oriented signals better identify recall-critical full-attention layers.

5 Conclusion

In this paper, we presented **FlashMorph**, an effective, efficient, and scalable layer selection method for converting pretrained Transformers into hybrid attention models. Rather than relying on fixed placement rules or isolated layerwise scoring, FlashMorph formulates hybrid layer selection as a budget-constrained joint optimization problem that accounts for inter-layer dependencies, redundancy, and complementarity, and constructs morphable attention layers to optimize lightweight layerwise gates under a global hybrid configuration. Extensive experiments across Qwen3-series backbones and multiple linear-attention variants show that FlashMorph preserves strong long-context retrieval and recall-intensive performance, maintains competitive commonsense reasoning ability, and substantially reduces layer selection overhead, demonstrating the effectiveness, efficiency, and scalability of joint optimization-based Transformer-to-hybrid conversion.

AI USE STATEMENT

In this work, we used generative AI tools for language polishing, typographical error checking, and visualization-related assistance, including preparing existing experimental results for plotting and writing or refining plotting code. We did not use generative AI tools for other applicable tasks requiring disclosure under the ICLR 2027 AI Policy for Authors.

We reviewed all AI-assisted textual edits to ensure that they preserved the intended meaning, inspected the plotting code, and checked that the figures accurately represented the underlying experimental results. We take responsibility for the final content of this work, including all text, claims, code, and figures produced with the assistance of generative AI tools.

ETHICS STATEMENT

This work does not introduce new data collection, user interaction, or application-specific decision-making mechanisms. All experiments are conducted on publicly available academic benchmarks and synthetic retrieval data, without involving human subjects, sensitive personal data, or deployment in real-world decision-making or safety-critical settings. We therefore believe that this work poses no direct ethical risks beyond those associated with the general use of large language models.

REPRODUCIBILITY STATEMENT

We have provided our core algorithm pseudocode in Algorithm 1 and the detailed experimental settings (model configurations, hyperparameters, *etc*.) in Sec.4.1, Appendices A and B to facilitate reproducibility.

REFERENCES

Simran Arora, Brandon Yang, Sabri Eyuboglu, Avanika Narayan, Andrew Hojel, Immanuel Trummer, and Christopher Ré. Language models enable simple systems for generating structured views of heterogeneous data lakes. *arXiv preprint arXiv:2304.09433*, 2023.

Simran Arora, Sabri Eyuboglu, Michael Zhang, Aman Timalsina, Silas Alberti, Dylan Zinsley, James Zou, Atri Rudra, and Christopher Ré. Simple linear attention language models balance the recall-throughput tradeoff. *arXiv preprint arXiv:2402.18668*, 2024.

Aviv Bick, Kevin Y Li, Eric P Xing, J Zico Kolter, and Albert Gu. Transformers to ssms: Distilling quadratic knowledge to subquadratic models. *Advances in neural information processing systems*, 37:31788–31812, 2024.

Aviv Bick, Tobias Katsch, Nimit Sohoni, Arjun Desai, and Albert Gu. Llamba: Scaling distilled recurrent models for efficient language processing. *arXiv preprint arXiv:2502.14458*, 2025.

Aviv Bick, Eric P Xing, and Albert Gu. Retrieval-aware distillation for transformer-ssm hybrids. *arXiv preprint arXiv:2602.11374*, 2026.

Yonatan Bisk, Rowan Zellers, Jianfeng Gao, Yejin Choi, et al. Piqa: Reasoning about physical commonsense in natural language. In *Proceedings of the AAAI conference on artificial intelligence*, volume 34, pp. 7432–7439, 2020.

Aaron Blakeman, Aarti Basant, Abhinav Khattar, Adithya Renduchintala, Akhiad Bercovich, Aleksander Ficek, Alexis Bjorlin, Ali Taghibakhshi, Amala Sanjay Deshmukh, Ameya Sunil Mahabaleshwarkar, et al. Nemotron-h: A family of accurate and efficient hybrid mamba-transformer models. *arXiv preprint arXiv:2504.03624*, 2025.

Zheng Cai, Maosong Cao, Haojiong Chen, Kai Chen, Keyu Chen, Xin Chen, Xun Chen, Zehui Chen, Zhi Chen, Pei Chu, et al. Internlm2 technical report. *arXiv preprint arXiv:2403.17297*, 2024.

Aili Chen, Aonian Li, Bangwei Gong, Binyang Jiang, Bo Fei, Bo Yang, Boji Shan, Changqing Yu, Chao Wang, Cheng Zhu, et al. Minimax-m1: Scaling test-time compute efficiently with lightning attention. *arXiv preprint arXiv:2506.13585*, 2025.

Hanting Chen, Zhicheng Liu, Xutao Wang, Yuchuan Tian, and Yunhe Wang. Dijiang: Efficient large language models through compact kernelization. *arXiv preprint arXiv:2403.19928*, 2024.

Yingfa Chen, Zhen Leng Thai, Zihan Zhou, Zhu Zhang, Xingyu Shen, Shuo Wang, Chaojun Xiao, Xu Han, and Zhiyuan Liu. Hybrid linear attention done right: Efficient distillation and effective architectures for extremely long contexts. *arXiv preprint arXiv:2601.22156*, 2026.

Yuhong Chou, Man Yao, Kexin Wang, Yuqi Pan, Ruijie Zhu, Yiran Zhong, Yu Qiao, Jibin Wu, Bo Xu, and Guoqi Li. Metala: Unified optimal linear approximation to softmax attention map. *Advances in Neural Information Processing Systems*, 37:71034–71067, 2024.

Peter Clark, Isaac Cowhey, Oren Etzioni, Tushar Khot, Ashish Sabharwal, Carissa Schoenick, and Oyvind Tafjord. Think you have solved question answering? try arc, the ai2 reasoning challenge. *arXiv preprint arXiv:1803.05457*, 2018.

Tri Dao and Albert Gu. Transformers are ssms: Generalized models and efficient algorithms through structured state space duality. *arXiv preprint arXiv:2405.21060*, 2024.

Soham De, Samuel L Smith, Anushan Fernando, Aleksandar Botev, George Cristian-Muraru, Albert Gu, Ruba Haroun, Leonard Berrada, Yutian Chen, Srivatsan Srinivasan, et al. Griffin: Mixing gated linear recurrences with local attention for efficient language models. *arXiv preprint arXiv:2402.19427*, 2024.

Jusen Du, Jiaxi Hu, Tao Zhang, Weigao Sun, and Yu Cheng. Native hybrid attention for efficient sequence modeling. *arXiv preprint arXiv:2510.07019*, 2025a.

Jusen Du, Weigao Sun, Disen Lan, Jiaxi Hu, and Yu Cheng. Mom: Linear sequence modeling with mixture-of-memories. *arXiv preprint arXiv:2502.13685*, 2025b.

Leo Gao, Jonathan Tow, Baber Abbasi, Stella Biderman, Sid Black, Anthony DiPofi, Charles Foster, Laurence Golding, Jeffrey Hsu, Alain Le Noac'h, Haonan Li, Kyle McDonell, Niklas Muennighoff, Chris Ociepa, Jason Phang, Laria Reynolds, Hailey Schoelkopf, Aviya Skowron, Lintang Sutawika, Eric Tang, Anish Thite, Ben Wang, Kevin Wang, and Andy Zou. The language model evaluation harness, 07 2024. URL https://zenodo.org/records/12608602.

Paolo Glorioso, Quentin Anthony, Yury Tokpanov, James Whittington, Jonathan Pilault, Adam Ibrahim, and Beren Millidge. Zamba: A compact 7b ssm hybrid model. *arXiv preprint arXiv:2405.16712*, 2024.

Daniel Goldstein, Eric Alcaide, Janna Lu, and Eugene Cheah. Radlads: Rapid attention distillation to linear attention decoders at scale. *arXiv preprint arXiv:2505.03005*, 2025.

Aaron Grattafiori, Abhimanyu Dubey, Abhinav Jauhri, Abhinav Pandey, Abhishek Kadian, Ahmad Al-Dahle, Aiesha Letman, Akhil Mathur, Alan Schelten, Alex Vaughan, et al. The llama 3 herd of models. *arXiv preprint arXiv:2407.21783*, 2024.

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

Yuxian Gu, Qinghao Hu, Shang Yang, Haocheng Xi, Junyu Chen, Song Han, and Han Cai. Jet-nemotron: Efficient language model with post neural architecture search. *arXiv preprint arXiv:2508.15884*, 2025.

Daya Guo, Dejian Yang, Haowei Zhang, Junxiao Song, Peiyi Wang, Qihao Zhu, Runxin Xu, Ruoyu Zhang, Shirong Ma, Xiao Bi, et al. Deepseek-r1: Incentivizing reasoning capability in llms via reinforcement learning. *arXiv preprint arXiv:2501.12948*, 2025.

Cheng-Ping Hsieh, Simeng Sun, Samuel Kriman, Shantanu Acharya, Dima Rekesh, Fei Jia, Yang Zhang, and Boris Ginsburg. Ruler: What's the real context size of your long-context language models? *arXiv preprint arXiv:2404.06654*, 2024.

Jiaxi Hu, Yongqi Pan, Jusen Du, Disen Lan, Xiaqiang Tang, Qingsong Wen, Yuxuan Liang, and Weigao Sun. Comba: Improving bilinear rnns with closed-loop control. *arXiv preprint arXiv:2506.02475*, 2025.

Shengding Hu, Yuge Tu, Xu Han, Chaoqun He, Ganqu Cui, Xiang Long, Zhi Zheng, Yewei Fang, Yuxiang Huang, Weilin Zhao, et al. Minicpm: Unveiling the potential of small language models with scalable training strategies. *arXiv preprint arXiv:2404.06395*, 2024.

Jungo Kasai, Hao Peng, Yizhe Zhang, Dani Yogatama, Gabriel Ilharco, Nikolaos Pappas, Yi Mao, Weizhu Chen, and Noah A Smith. Finetuning pretrained transformers into rnns. In *Proceedings of the 2021 conference on empirical methods in natural language processing*, pp. 10630–10643, 2021.

Angelos Katharopoulos, Apoorv Vyas, Nikolaos Pappas, and François Fleuret. Transformers are rnns: Fast autoregressive transformers with linear attention. In *International conference on machine learning*, pp. 5156–5165. PMLR, 2020.

Woosuk Kwon, Zhuohan Li, Siyuan Zhuang, Ying Sheng, Lianmin Zheng, Cody Hao Yu, Joseph Gonzalez, Hao Zhang, and Ion Stoica. Efficient memory management for large language model serving with pagedattention. In *Proceedings of the 29th symposium on operating systems principles*, pp. 611–626, 2023.

Aakash Lahoti, Kevin Y Li, Berlin Chen, Caitlin Wang, Aviv Bick, J Zico Kolter, Tri Dao, and Albert Gu. Mamba-3: Improved sequence modeling using state space principles. *arXiv preprint arXiv:2603.15569*, 2026.

Disen Lan, Weigao Sun, Jiaxi Hu, Jusen Du, and Yu Cheng. Liger: Linearizing large language models to gated recurrent structures. *arXiv preprint arXiv:2503.01496*, 2025.

Jeffrey Li, Alex Fang, Georgios Smyrnis, Maor Ivgi, Matt Jordan, Samir Gadre, Hritik Bansal, Etash Guha, Sedrick Keh, Kushal Arora, et al. Datacomp-lm: In search of the next generation of training sets for language models. *Advances in Neural Information Processing Systems*, 37:14200–14282, 2024.

Yanhong Li, Songlin Yang, Shawn Tan, Mayank Mishra, Rameswar Panda, Jiawei Zhou, and Yoon Kim. Distilling to hybrid attention models via kl-guided layer selection. *arXiv preprint arXiv:2512.20569*, 2025.

Opher Lieber, Barak Lenz, Hofit Bata, Gal Cohen, Jhonathan Osin, Itay Dalmedigos, Erez Safahi, Shaked Meirom, Yonatan Belinkov, Shai Shalev-Shwartz, et al. Jamba: A hybrid transformer-mamba language model. *arXiv preprint arXiv:2403.19887*, 2024.

Gang Lin, Dongfang Li, Zhuoen Chen, Yukun Shi, Xuhui Chen, Baotian Hu, and Min Zhang. Lycheedecode: Accelerating long-context llm inference via hybrid-head sparse decoding. *arXiv preprint arXiv:2602.04541*, 2026.

Aixin Liu, Bei Feng, Bing Xue, Bingxuan Wang, Bochao Wu, Chengda Lu, Chenggang Zhao, Chengqi Deng, Chenyu Zhang, Chong Ruan, et al. Deepseek-v3 technical report. *arXiv preprint arXiv:2412.19437*, 2024.

Colin Lockard, Prashant Shiralkar, and Xin Luna Dong. Openceres: When open information extraction meets the semi-structured web. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pp. 3047–3056, 2019.

Ilya Loshchilov and Frank Hutter. Decoupled weight decay regularization. *arXiv preprint arXiv:1711.05101*, 2017.

Jean Mercat, Igor Vasiljevic, Sedrick Keh, Kushal Arora, Achal Dave, Adrien Gaidon, and Thomas Kollar. Linearizing large language models. *arXiv preprint arXiv:2405.06640*, 2024.

William Merrill, Yanhong Li, Tyler Romero, Anej Svete, Caia Costello, Pradeep Dasigi, Dirk Groeneveld, David Heineman, Bailey Kuehl, Nathan Lambert, et al. Olmo hybrid: From theory to practice and back. *arXiv preprint arXiv:2604.03444*, 2026.

Daniele Paliotta, Junxiong Wang, Matteo Pagliardini, Kevin Y Li, Aviv Bick, J Zico Kolter, Albert Gu, François Fleuret, and Tri Dao. Thinking slow, fast: Scaling inference compute with distilled reasoners. *arXiv preprint arXiv:2502.20339*, 2025.

Bo Peng, Eric Alcaide, Quentin Anthony, Alon Albalak, Samuel Arcadinho, Stella Biderman, Huanqi Cao, Xin Cheng, Michael Chung, Leon Derczynski, et al. Rwkv: Reinventing rnns for the transformer era. In *Findings of the association for computational linguistics: EMNLP 2023*, pp. 14048–14077, 2023.

Bowen Peng, Jeffrey Quesnelle, Honglu Fan, and Enrico Shippole. Yarn: Efficient context window extension of large language models. In *International Conference on Learning Representations*, volume 2024, pp. 31932–31951, 2024.

Zhen Qin, Songlin Yang, and Yiran Zhong. Hierarchically gated recurrent neural network for sequence modeling. *Advances in Neural Information Processing Systems*, 36:33202–33221, 2023.

Zhen Qin, Weigao Sun, Dong Li, Xuyang Shen, Weixuan Sun, and Yiran Zhong. Lightning attention-2: A free lunch for handling unlimited sequence lengths in large language models. *arXiv preprint arXiv:2401.04658*, 2024a.

Zhen Qin, Songlin Yang, Weixuan Sun, Xuyang Shen, Dong Li, Weigao Sun, and Yiran Zhong. Hgrn2: Gated linear rnns with state expansion. *arXiv preprint arXiv:2404.07904*, 2024b.

Zihan Qiu, Zekun Wang, Bo Zheng, Zeyu Huang, Kaiyue Wen, Songlin Yang, Rui Men, Le Yu, Fei Huang, Suozhi Huang, et al. Gated attention for large language models: Non-linearity, sparsity, and attention-sink-free. *Advances in Neural Information Processing Systems*, 38:100092–100118, 2026.

Qwen Team. Qwen3.5: Towards native multimodal agents, February 2026. URL https://qwen.ai/blog?id=qwen3.5.

Pranav Rajpurkar, Robin Jia, and Percy Liang. Know what you don't know: Unanswerable questions for squad. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pp. 784–789, 2018.

Liliang Ren, Yang Liu, Yadong Lu, Chen Liang, Weizhu Chen, et al. Samba: Simple hybrid state space models for efficient unlimited context language modeling. In *International Conference on Learning Representations*, volume 2025, pp. 53551–53575, 2025.

Keisuke Sakaguchi, Ronan Le Bras, Chandra Bhagavatula, and Yejin Choi. Winogrande: An adversarial winograd schema challenge at scale. *Communications of the ACM*, 64(9):99–106, 2021.

Weigao Sun, Jiaxi Hu, Yucheng Zhou, Jusen Du, Disen Lan, Kexin Wang, Tong Zhu, Xiaoye Qu, Yu Zhang, Xiaoyu Mo, et al. Speed always wins: A survey on efficient architectures for large language models. *arXiv preprint arXiv:2508.09834*, 2025a.

Weigao Sun, Disen Lan, Tong Zhu, Xiaoye Qu, and Yu Cheng. Linear-moe: Linear sequence modeling meets mixture-of-experts. *arXiv preprint arXiv:2503.05447*, 2025b.

Kimi Team, Yu Zhang, Zongyu Lin, Xingcheng Yao, Jiaxi Hu, Fanqing Meng, Chengyin Liu, Xin Men, Songlin Yang, Zhiyuan Li, et al. Kimi linear: An expressive, efficient attention architecture. *arXiv preprint arXiv:2510.26692*, 2025.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. *Advances in neural information processing systems*, 30, 2017.

Dustin Wang, Rui-Jie Zhu, Steven Abreu, Yong Shan, Taylor Kergan, Yuqi Pan, Yuhong Chou, Zheng Li, Ge Zhang, Wenhao Huang, et al. A systematic analysis of hybrid linear attention. *arXiv preprint arXiv:2507.06457*, 2025.

Junxiong Wang, Daniele Paliotta, Avner May, Alexander M Rush, and Tri Dao. The mamba in the llama: Distilling and accelerating hybrid models. *Advances in Neural Information Processing Systems*, 37:62432–62457, 2024.

Kaiyue Wen, Xingyu Dang, and Kaifeng Lyu. Rnns are not transformers (yet): The key bottleneck on in-context retrieval. In *International Conference on Learning Representations*, volume 2025, pp. 48813–48856, 2025.

Guangxuan Xiao, Jiaming Tang, Jingwei Zuo, Junxian Guo, Shang Yang, Haotian Tang, Yao Fu, and Song Han. Duoattention: Efficient long-context llm inference with retrieval and streaming heads. In *International Conference on Learning Representations*, volume 2025, pp. 37228–37253, 2025.

An Yang, Anfeng Li, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chang Gao, Chengen Huang, Chenxu Lv, et al. Qwen3 technical report. *arXiv preprint arXiv:2505.09388*, 2025.

Mingyu Yang, Mehdi Rezagholizadeh, Guihong Li, Vikram Appia, and Emad Barsoum. Zebra-llama: Towards extremely efficient hybrid models. *Advances in Neural Information Processing Systems*, 38:78167–78194, 2026.

Songlin Yang and Yu Zhang. Fla: A triton-based library for hardware-efficient implementations of linear attention mechanism, January 2024. URL https://github.com/fla-org/flash-linear-attention.

Songlin Yang, Bailin Wang, Yikang Shen, Rameswar Panda, and Yoon Kim. Gated linear attention transformers with hardware-efficient training. *arXiv preprint arXiv:2312.06635*, 2023.

Songlin Yang, Jan Kautz, and Ali Hatamizadeh. Gated delta networks: Improving mamba2 with delta rule. *arXiv preprint arXiv:2412.06464*, 2024a.

Songlin Yang, Bailin Wang, Yu Zhang, Yikang Shen, and Yoon Kim. Parallelizing linear transformers with the delta rule over sequence length. *Advances in neural information processing systems*, 37: 115491–115522, 2024b.

Rowan Zellers, Ari Holtzman, Yonatan Bisk, Ali Farhadi, and Yejin Choi. Hellaswag: Can a machine really finish your sentence? In *Proceedings of the 57th annual meeting of the association for computational linguistics*, pp. 4791–4800, 2019.

Michael Zhang, Simran Arora, Rahul Chalamala, Alan Wu, Benjamin Spector, Aaryan Singhal, Krithik Ramesh, and Christopher Ré. Lolcats: On low-rank linearizing of large language models. *arXiv preprint arXiv:2410.10254*, 2024a.

Michael Zhang, Kush Bhatia, Hermann Kumbong, and Christopher Ré. The hedgehog & the porcupine: Expressive linear attentions with softmax mimicry. *arXiv preprint arXiv:2402.04347*, 2024b.

Yu Zhang, Songlin Yang, Ruijie Zhu, Yue Zhang, Leyang Cui, Yiqiao Wang, Bolun Wang, Freda Shi, Bailin Wang, Wei Bi, et al. Gated slot attention for efficient linear-time sequence modeling. *Advances in Neural Information Processing Systems*, 37:116870–116898, 2024c.

Jingwei Zuo, Maksim Velikanov, Ilyas Chahed, Younes Belkada, Dhia Eddine Rhayem, Guillaume Kunsch, Hakim Hacid, Hamza Yous, Brahim Farhat, Ibrahim Khadraoui, et al. Falcon-h1: A family of hybrid-head language models redefining efficiency and performance. *arXiv preprint arXiv:2507.22448*, 2025.

A Model and Training Configuration

We report the complete model and training configuration in Table 5. All training data are randomly sampled from the DCLM corpus (Li et al., 2024); in each stage, we use only a token subset of the specified size to control the data budget across model variants and training stages. We use the AdamW optimizer (Loshchilov & Hutter, 2017) with beta values of $(0.9, 0.95)$ and a weight decay of 0.0 throughout all training stages. Following HALO (Chen et al., 2026), the overall Transformer-to-hybrid conversion training pipeline of FlashMorph consists of four stages: hidden-state alignment for morphable layer construction, layer selection, distillation, and long-context finetuning. Stage-specific hyperparameters are summarized in Table 5. We also provide the full procedure in Algorithm 1.

B Implementation Details

During the layer selection stage, we construct a synthetic long-context retrieval dataset based on the DCLM corpus (Li et al., 2024). For each example, we use DCLM text as the background context and insert ten randomly generated passkey sequences at different depths. Each passkey contains s words sampled from a fixed alphabet, with $s = 32$ in our experiments. The context length is randomly sampled from 50 length intervals ranging from 1K to 16K tokens. For insertion, we discretize the context depth into 1,000 candidate points and randomly sample the insertion positions of the passkeys. The model is then required to recall all ten passkeys at the end of the context. This design provides dense long-range retrieval supervision, enabling us to identify which layers can be replaced by linear attention while preserving the model's ability to recover information from distant positions under long-context scenarios.

Case Sample from Synthetic Retrieval Data

query

<|im_start|> This is a very long story book: <book>

... background DCLM text ...

Remember this sequence of words, it is the first passkey to the vault:
[Passkey 1] xray whiskey lima charlie papa hotel mike india quebec mike victor echo kilo lima charlie india yankee golf golf delta uniform alpha whiskey golf yankee zulu charlie november november charlie charlie golf

... more DCLM text with the second to ninth passkeys inserted at different depths ...

Remember this sequence of words, it is the tenth passkey to the vault:
[Passkey 10] whiskey hotel charlie sierra xray kilo golf uniform alpha bravo alpha lima november quebec november victor xray whiskey tango juliett papa india delta uniform alpha mike golf uniform india delta quebec zulu

</book>.

response

Based on the content of the book, what is the first passkey to the vault?
[Answer 1] Passkey: xray whiskey lima charlie papa hotel mike india quebec mike victor echo kilo lima charlie india yankee golf golf delta uniform alpha whiskey golf yankee zulu charlie november november charlie charlie golf

...

Based on the content of the book, what is the tenth passkey to the vault?
[Answer 10] Passkey: whiskey hotel charlie sierra xray kilo golf uniform alpha bravo alpha lima november quebec november victor xray whiskey tango juliett papa india delta uniform alpha mike golf uniform india delta quebec zulu

Table 5: **Hyperparameter Configurations for FlashMorph Models and Training Pipeline.**

Setting	Hyperparameter	0.6B	1.7B	8B	30B-A3B
	Model Architecture				
Backbone	#layers	28	28	36	48
	hidden size	1024	2048	4096	2048
	FFN width	3072	6144	12288	6144
	#full-attention layers	7	7	9	12
	#linear-attention layers	21	21	27	36
	head dimension	128	128	128	128
	#attention heads	16	16	32	32
	#full-attention KV heads	8	8	8	4
	#linear-attention KV heads	16	16	32	32
	attn. logits scaling α *(if applicable)*	300	500	900	600
	Training Pipeline				
Stage 1 (Hidden-state Alignment)	tokens	320M			
	LR scheduler	cosine			
	learning rate	$1e{-3} \rightarrow 1e{-5}$			
	sequence length	512			
	batch size	32			
	warmup steps	50			
	training steps	20,000			
Stage 2 (Layer Selection)	tokens	20M			
	LR scheduler	WSD (Hu et al., 2024)			
	learning rate	$2e{-2} \rightarrow 2e{-3}$			
	sequence length	$<$ 16K			
	batch size	8			
	warmup steps	50			
	decay steps	50			
	training steps	250			
Stage 3 (Distillation)	tokens	1B			
	LR scheduler	cosine			
	learning rate	$1e{-4} \rightarrow 1e{-5}$			
	sequence length	512			
	batch size	96			
	warmup steps	50			
	training steps	20,000			
Stage 4 (Long-context Finetuning)	tokens	1B			
	LR scheduler	constant			
	learning rate	$1e{-5}$			
	sequence length	16K			
	batch size	128			
	warmup steps	50			
	training steps	500			

Algorithm 1 FlashMorph Transformer-to-Hybrid Conversion

Require: Full-attention model $\mathcal{M}_{\text{full}}$ with L layers; training dataset $\mathcal{D}$; synthetic retrieval dataset $\mathcal{D}_{\text{syn}}$; layer selection optimization steps S; full-attention budget K; linearization regularization weight λ.

Distill an all-linear model $\mathcal{M}_{\text{all-linear}}$ from $\mathcal{M}_{\text{full}}$ on $\mathcal{D}$ by hidden-state alignment.
Construct a morphable model by equipping each full-attention branch $A^{(l)}_{\text{full}}$ with its trained linear-attention replacement $A^{(l)}_{\text{lin}}$.
Freeze both the full-attention model backbone and the linear-attention branches.
Initialize layerwise gates $\alpha^{(l)} \leftarrow 1$ for all $l = 1, \ldots, L$.
for $s = 1, \ldots, S$ **do**
    Sample synthetic retrieval examples $x \sim \mathcal{D}_{\text{syn}}$.
    Compute mixed hidden states for each layer:

$$\mathbf{H}^{(l)}_{\text{mix}} = \alpha^{(l)} \mathbf{H}^{(l)}_{\text{full}} + \left(1 - \alpha^{(l)}\right) \mathbf{H}^{(l)}_{\text{lin}}.$$

    Compute the answer-token alignment loss $\mathcal{L}_{\text{align}}$.
    Compute the linearization regularizer $\mathcal{L}_{\text{reg}} = \sum_{l=1}^{L} \alpha^{(l)}$.
    Update only the gates $\boldsymbol{\alpha}$ by minimizing $\mathcal{L}_{\text{total}} = \mathcal{L}_{\text{align}} + \lambda \mathcal{L}_{\text{reg}}$.
end for
Select full-attention layers:

$$\mathcal{I}^{\text{Hybrid}}_{\text{full}} = \text{TopK}\left(\{\alpha^{(l)}\}_{l=1}^{L}, K\right).$$

Set linear-attention layers:

$$\mathcal{I}^{\text{Hybrid}}_{\text{lin}} = [L] \setminus \mathcal{I}^{\text{Hybrid}}_{\text{full}}.$$

Instantiate $\mathcal{M}_{\text{Hybrid}}$ with full attention on $\mathcal{I}^{\text{Hybrid}}_{\text{full}}$ and linear attention on $\mathcal{I}^{\text{Hybrid}}_{\text{lin}}$; discard the gates $\boldsymbol{\alpha}$.
Apply logits distillation and long-context finetuning on $\mathcal{D}$.
return $\mathcal{M}_{\text{Hybrid}}$.

C More Experiment Results

As presented in Table 6 and Table 7, we further evaluate FlashMorph on Qwen3-8B and Qwen3-30B-A3B under the HypeNet setting (Chen et al., 2026), comparing it against uniform interleaving and HALO. We exclude PostNAS and KL-LS at these scales because reproducing their layer-selection procedures would incur prohibitive computational costs, particularly for larger models.

Table 6: **NIAH Performance on 8B Dense and 30B-A3B MoE Backbones across 32K-256K Context Lengths.** The best and second-best results are in **bold** and underlined, respectively.

Model	LS. Tokens	NIAH-Single-1				NIAH-Single-2				NIAH-Single-3			
		32K	64K	128K	256K	32K	64K	128K	256K	32K	64K	128K	256K
8B backbone													
Qwen3	-	100	100	52.8	0	100	100	0	0	100	99.8	0	0
Qwen3+YaRN	-	100	100	100	100	99.4	99.6	98.6	74.8	99.6	98.0	98.8	92.8
Uniform	N/A	91.8	92.4	93.2	92.6	91.4	60.6	31.6	17.8	58.2	54.2	40.6	26.2
HALO	234M	99.2	99.4	99.2	98.4	96.8	95.6	**89.2**	68.4	89.8	85.2	75.0	50.8
FlashMorph	**20M**	**99.8**	**99.6**	**99.6**	**99.2**	**98.0**	**99.4**	85.6	**82.6**	**99.4**	**98.2**	**92.4**	**94.0**
30B-A3B backbone													
Qwen3	-	100	100	1.0	0	100	99.8	0	0	100	100	0	0
Qwen3+YaRN	-	98.2	4.4	2.0	20.0	84.6	61.0	79.6	73.0	37.4	17.2	26.6	9.0
Uniform	N/A	98.4	**99.4**	99.0	**99.6**	73.6	53.4	35.4	19.0	**55.2**	36.0	18.4	9.6
HALO	234M	97.6	98.6	**99.4**	99.0	**94.8**	**79.4**	**61.2**	**27.6**	47.0	17.8	12.0	0.8
FlashMorph	**20M**	**98.6**	96.2	94.0	91.4	70.2	53.6	21.4	9.6	32.6	**38.2**	**24.6**	**18.2**

Table 7: **Zero-shot Performance on Commonsense Reasoning and Long-context Recall-intensive Tasks under the HypeNet Setting.** The best and second-best results are in **bold** and underlined, respectively.

Method	LS. Tokens	ARC-e acc	ARC-c acc_n	PIQA acc	Hella. acc_n	Wino. acc	Avg.	SQuAD acc	FDA acc	SWDE acc	Avg.
0.6B backbone											
Qwen3	-	60.6	34.1	67.6	47.3	55.7	53.1	44.1	82.1	80.5	68.9
Uniform	N/A	61.2	**33.1**	66.8	44.7	**56.6**	52.1	20.7	52.6	70.6	48.0
KL-LS	20B	61.9	**33.1**	**67.8**	46.4	53.9	52.6	25.7	58.4	70.8	51.6
HALO	234M	**62.9**	32.5	67.1	**46.5**	55.7	**52.9**	22.8	**62.8**	73.5	53.0
FlashMorph	**20M**	**62.9**	30.6	67.0	46.0	55.1	52.3	**35.3**	58.2	**76.8**	**56.7**
1.7B backbone											
Qwen3	-	72.4	43.5	72.5	60.4	61.0	62.0	39.8	79.0	85.1	67.9
Uniform	N/A	73.2	42.6	**73.0**	60.1	62.8	**62.4**	40.2	64.5	80.4	61.7
PostNAS*	50B	**73.7**	42.3	72.9	**60.3**	61.6	62.1	**43.5**	63.5	**81.5**	62.8
KL-LS	20B	72.7	42.8	72.4	59.6	58.8	61.3	26.7	48.8	73.9	49.8
HALO	234M	72.5	41.9	72.6	60.0	**63.7**	62.1	38.5	67.8	80.7	62.3
FlashMorph	**20M**	73.1	**43.3**	**73.0**	60.1	61.8	62.3	39.8	**70.2**	81.4	**63.8**
8B backbone											
Qwen3	-	83.6	56.6	76.8	75.0	67.7	71.9	72.3	78.2	90.8	80.4
Uniform	N/A	81.4	56.5	**77.4**	**73.7**	70.1	71.8	49.7	63.3	84.1	65.7
HALO	234M	**82.5**	57.4	77.2	72.7	69.9	71.9	41.9	59.7	80.9	60.8
FlashMorph	**20M**	81.9	**57.8**	77.2	73.1	**71.1**	**72.2**	**52.7**	**73.5**	**87.3**	**71.2**
30B-A3B backbone											
Qwen3	-	79.5	56.1	79.5	77.7	70.9	72.7	58.7	81.0	90.8	76.8
Uniform	N/A	79.3	51.9	77.9	73.1	66.5	69.8	**23.0**	35.9	76.8	45.2
HALO	234M	75.6	47.9	76.4	68.8	61.6	66.0	20.8	41.0	77.0	46.3
FlashMorph	**20M**	**80.9**	**56.1**	**79.9**	**75.0**	**72.2**	**72.8**	21.4	**53.4**	**81.5**	**52.1**

D Complete Layer Importance Ranking

We provide the complete layer importance rankings in Table 8. For each backbone and linear-attention variant, layers are sorted from the most to the least important, and the top-ranked layers are retained as full-attention layers under the fixed hybrid budget.

E Related Work

E.1 Linear RNN and Hybrid Attention

To address the quadratic computational cost and massive memory usage of full attention, a variety of linear RNN architectures have been proposed to enable more efficient decoding and scalable long-context processing (Katharopoulos et al., 2020; Peng et al., 2023; Gu & Dao, 2023; Dao & Gu, 2024; Qin et al., 2023; 2024a; Chou et al., 2024; Sun et al., 2025b; Zhang et al., 2024c; Yang et al., 2024b;a; Du et al., 2025b; Lahoti et al., 2026; Hu et al., 2025). However, entirely replacing full attention with linear RNN-style sequence mixers can introduce a fixed-state memory bottleneck, limiting the model's capacity for recall-intensive operations such as associative retrieval and long-range information access (Arora et al., 2024; Wen et al., 2025).

Rather than replacing full attention entirely, hybrid attention models combine full attention with linear RNN mixers (Lieber et al., 2024; Glorioso et al., 2024; De et al., 2024; Ren et al., 2025; Chen et al., 2025; Zuo et al., 2025; Blakeman et al., 2025; Wang et al., 2025; Du et al., 2025a; Team et al., 2025; Qwen Team, 2026; Merrill et al., 2026). By retaining a small subset of full-attention layers, these models preserve global information access, while using linear RNN layers to improve inference efficiency. However, most existing hybrid architectures are designed and pretrained from scratch, typically relying on fixed allocation patterns such as uniform interleaving of full-attention and linear-attention layers. Although effective in the pretraining setting, such handcrafted patterns do not directly address the layer selection problem that arises when converting an existing pretrained Transformer into a hybrid attention model: determining which layers should retain full attention and which can be safely linearized.

E.2 TRANSFORMER-TO-HYBRID CONVERSION

Converting pretrained Transformers into efficient hybrid attention models has emerged as an appealing alternative to training from scratch. Existing conversion pipelines typically replace full-attention layers with linear-attention through weight transfer, followed by distillation and continued finetuning to recover model quality (Kasai et al., 2021; Mercat et al., 2024; Chen et al., 2024; Zhang et al., 2024b; Wang et al., 2024; Bick et al., 2024; Lan et al., 2025; Bick et al., 2025; Paliotta et al., 2025; Goldstein et al., 2025; Li et al., 2025; Chen et al., 2026). In the Transformer-to-hybrid conversion setting, only a limited number of layers can retain full attention, while the remaining layers are linearized. This makes layer selection a central challenge: under a fixed full-attention budget, the conversion process must determine which layers should preserve full attention and which layers can be safely replaced by linear attention.

Existing hybrid layer selection methods typically rely on uniform interleaving or layerwise importance estimation. Uniform interleaving is simple and architecture-agnostic, but it ignores the non-uniform functional roles of pretrained Transformer layers. Search-based methods such as PostNAS (Gu et al., 2025) move beyond purely layerwise scoring, but require training a supernet, which introduces substantial search overhead. Layerwise methods estimate marginal utility of each layer by perturbing, replacing, restoring, and scoring one layer at a time, as in SMART (Yang et al., 2026), KL-LS (Li et al., 2025), and HALO (Chen et al., 2026). Although practical, these layerwise procedures are cumbersome and treat layer importance in isolation, thereby overlooking interdependent layer effect. In contrast, FlashMorph formulates hybrid layer selection as a joint optimization problem. By learning layerwise gates within a morphable model via linearization regularization, FlashMorph accounts for inter-layer dependencies, redundancy, and complementarity under a global hybrid configuration, while simultaneously avoiding the repeated layer-by-layer replacement and evaluation required by prior methods, which makes the layer selection stage substantially more efficient.

Table 8: **Complete Layer Selection Results for FlashMorph and Baseline Methods.** Layer indices are sorted from most important to least important, and the boxed prefix denotes the top-25% layers retained as full-attention layers under the fixed hybrid ratio budget. Red indices in baseline rows indicate selected layers that are not shared with FlashMorph. * denotes the selected layer results taken from (Chen et al., 2026).

Attn.	Method	Layer indices (most important → least important)
Qwen3-0.6B backbone		
Lightning	FlashMorph (Ours)	[1, 16, 21, 11, 19, 24, 0,] 25, 18, 2, 6, 8, 20, 3, 26, 13, 9, 23, 22, 10, 14, 17, 4, 15, 12, 27, 5, 7
	HALO	[10, 21, 9, 5, 11, 1, 13,] 16, 25, 12, 19, 24, 6, 18, 15, 8, 2, 26, 14, 23, 0, 27, 20, 7, 17, 22, 3, 4
	KL-LS	[21, 16, 19, 20, 18, 22, 25,] 24, 26, 17, 23, 14, 8, 12, 6, 13, 11, 15, 3, 2, 9, 4, 1, 5, 0, 10, 7, 27
GLA	FlashMorph (Ours)	[21, 16, 11, 1, 2, 6, 25,] 19, 24, 18, 20, 8, 0, 26, 13, 22, 3, 23, 9, 10, 14, 17, 12, 15, 4, 7, 27, 5
	HALO	[8, 21, 13, 1, 24, 6, 25,] 19, 18, 11, 10, 16, 9, 12, 5, 15, 2, 26, 27, 23, 20, 4, 0, 14, 3, 22, 7, 17
	KL-LS	[21, 16, 20, 19, 18, 24, 8,] 22, 17, 6, 23, 25, 26, 14, 12, 11, 13, 2, 3, 15, 9, 4, 1, 0, 5, 10, 7, 27
GDN	FlashMorph (Ours)	[1, 11, 21, 16, 19, 18, 24,] 6, 25, 8, 2, 20, 0, 13, 26, 3, 9, 23, 14, 22, 10, 12, 17, 15, 4, 7, 5, 27
	HALO	[10, 5, 21, 6, 19, 24, 18,] 11, 9, 13, 12, 25, 8, 2, 16, 26, 1, 4, 15, 0, 17, 7, 23, 14, 27, 3, 22, 20
	KL-LS	[21, 16, 19, 20, 18, 6, 22,] 24, 25, 11, 8, 14, 26, 12, 23, 17, 13, 3, 2, 9, 1, 4, 15, 5, 0, 10, 27, 7
Qwen3-1.7B backbone		
Lightning	FlashMorph (Ours)	[1, 16, 13, 11, 21, 3, 8,] 20, 6, 19, 18, 9, 14, 24, 2, 17, 0, 10, 15, 25, 23, 26, 22, 12, 4, 5, 7, 27
	HALO	[3, 14, 9, 2, 6, 16, 21,] 25, 24, 8, 23, 12, 11, 26, 27, 19, 18, 17, 7, 15, 4, 13, 1, 20, 10, 22, 0, 5
	KL-LS	[21, 16, 20, 19, 18, 22, 24,] 26, 25, 23, 17, 6, 14, 8, 12, 13, 11, 3, 15, 2, 9, 1, 4, 0, 5, 10, 7, 27
	PostNAS*	[0, 21, 25, 19, 6, 11, 9,] 24, 12, 2, 26, 16, 17, 23, 18, 4, 7, 3, 14, 20, 1, 27, 10, 13, 8, 22, 15, 5
GLA	FlashMorph (Ours)	[1, 11, 21, 3, 13, 14, 16,] 19, 9, 20, 6, 18, 2, 0, 24, 8, 10, 17, 25, 15, 23, 26, 12, 22, 4, 7, 27, 5
	HALO	[3, 14, 6, 8, 4, 25, 11,] 21, 16, 2, 24, 18, 26, 17, 19, 23, 1, 27, 12, 0, 15, 13, 7, 20, 9, 22, 10, 5
	KL-LS	[21, 16, 20, 19, 18, 24, 22,] 6, 8, 26, 11, 25, 23, 14, 12, 17, 3, 13, 2, 1, 9, 4, 15, 0, 5, 10, 27, 7
	PostNAS*	[0, 21, 25, 19, 6, 11, 9,] 24, 12, 2, 26, 16, 17, 23, 18, 4, 7, 3, 14, 20, 1, 27, 10, 13, 8, 22, 15, 5
GDN	FlashMorph (Ours)	[1, 11, 13, 21, 16, 14, 6,] 10, 20, 19, 2, 18, 8, 24, 0, 3, 9, 25, 15, 22, 17, 26, 23, 12, 4, 27, 7, 5
	HALO	[3, 2, 25, 21, 11, 14, 6,] 4, 12, 8, 16, 18, 24, 17, 19, 23, 7, 26, 27, 9, 1, 20, 22, 13, 0, 15, 5, 10
	KL-LS	[21, 16, 20, 19, 6, 11, 18,] 24, 22, 8, 25, 26, 12, 14, 23, 13, 2, 3, 17, 1, 9, 4, 15, 5, 10, 0, 27, 7
	PostNAS*	[0, 21, 25, 19, 6, 11, 9,] 24, 12, 2, 26, 16, 17, 23, 18, 4, 7, 3, 14, 20, 1, 27, 10, 13, 8, 22, 15, 5
Qwen3-8B backbone		
Lightning	FlashMorph (Ours)	[1, 0, 7, 22, 29, 13, 3, 15, 24,] 20, 9, 19, 33, 34, 12, 2, 17, 8, 32, 16, 14, 23, 21, 35, 18, 5, 31, 11, 10, 4, 28, 26, 30, 6, 25, 27
	HALO	[7, 6, 0, 24, 8, 33, 1, 12, 34,] 22, 2, 15, 26, 9, 20, 16, 29, 31, 21, 4, 17, 30, 5, 35, 11, 32, 3, 25, 27, 14, 18, 19, 28, 13, 23, 10
Qwen3-30B-A3B backbone		
Lightning	FlashMorph (Ours)	[37, 2, 4, 3, 1, 38, 5, 0, 6, 42, 8, 26,] 13, 22, 18, 21, 11, 9, 14, 20, 15, 25, 36, 24, 10, 41, 12, 34, 47, 17, 23, 30, 7, 45, 39, 16, 19, 29, 35, 43, 33, 46, 27, 44, 40, 32, 28, 31
	HALO	[5, 45, 42, 8, 12, 43, 36, 19, 34, 41, 18, 22,] 46, 27, 37, 4, 31, 39, 13, 9, 35, 10, 24, 7, 33, 25, 47, 23, 28, 17, 21, 11, 14, 0, 30, 15, 20, 26, 40, 3, 1, 6, 32, 38, 2, 29, 16, 44