Selective State-Space Adaptation and Retrieval for Language Model Reasoning

Anonymous ACL submission

Abstract

Low-rank adaptation introduces a static learned update applied identically to every input. The update provides task-level adaptation but does not explicitly represent token-level or instance-level state variation. A family of adapters is proposed that introduces selective state-space recurrence at two complementary granularities. At the token level, **MaLoRA** (Mamba-modulated low-rank adaptation) makes the adapter's scaling factor a dynamic input-dependent function with recurrent state across tokens, in contrast to the stateless modulators of prior work. At the context level, **MaRA** (Mamba Retrieval Adapter) tracks cross-segment state and selects the segments most relevant to the query, before the modulated language model generates its answer. Across three frozen backbones (Qwen-2.5-7B, Llama-3.1-8B, Gemma-2-9B) and two reasoning benchmarks (MuSiQue, 2WikiMultihopQA), the family improves reasoning accuracy on every cell of the 3×2 grid, by $+6.8$ F1 ($+10.5\%$ relative) on average and up to $+9.3$ F1 ($+18.2\%$ relative) on the hardest cell over the LoRA baseline, and the token-level gains carry to RULER QA-2 under length stress.

1 Introduction

Low-rank adaptation (Hu et al., 2022) has become the standard approach for adjusting a frozen language model to a downstream task. The adapter is trained once and applied uniformly to every input. For tasks with multiple latent states, such as multi-step reasoning, this static update is shared across all of those states, encouraging reliance on training-set patterns rather than per-input adjustment. The goal of this study is to make adaptation more input-adaptive and better aligned with the state structure of the query, while leaving the backbone frozen.

Adaptation can operate at three levels: at the *task* level, where LoRA already operates; at the *token* level, where the strength of adaptation can become a function of the current token; and at the *context* level, where the model can attend to the input segments most relevant to the query.

The first contribution of this work is **MaLoRA** (Mamba-modulated Low-Rank Adaptation), a recurrent token-level modulator for Low-Rank Adaptation. Unlike prior token modulators that use a stateless per-token scaling factor, MaLoRA uses a Mamba-based selective state-space module (Gu and Dao, 2024) to propagate state across tokens. The second contribution is **MaRA** (Mamba Retrieval Adapter), a *context-level* adapter that scores each input segment via a state-space module operating over segment embeddings, and selects the most relevant segments for the adapted model.

Selective state-space recurrence contributes at both levels of adaptation. At the token level, MaLoRA's recurrent state yields consistent gains over stateless modulation across backbones (Table 1). At the context level, the state-space module performs evidence selection, which the token modulator cannot do because it is relevance-blind at the paragraph level (Section 3.4). The same Mamba block therefore plays two different jobs in the same system.

The family is evaluated on MuSiQue (Trivedi et al., 2022) and 2WikiMultihopQA (Ho et al., 2020) across three frozen backbones (Qwen-2.5-7B, Llama-3.1-8B, Gemma-2-9B). RULER (Hsieh et al., 2024) measures retention of base-model capability under length stress. The token-level modulator and MaRA compose, and the combined system improves over LoRA on every cell of the 3×2 grid, with the two mechanisms contributing through complementary channels: modulation reshapes the low-rank update by token role, MaRA selects query-relevant segments before generation, and the gains accumulate rather than overlap.

2 Related Work

Parameter-efficient adaptation. Low-rank adaptation (Hu et al., 2022) adapts a frozen base model via a learned static update. Extensions add weight decomposition (Liu et al., 2024), adaptive rank allocation (Zhang et al., 2023), recurrent adapters (Nguyen et al., 2024), an asymmetric shared-A multi-head architecture (Tian et al., 2024), and mixture-of-LoRA routing (Dou et al., 2024; Gao et al., 2025). All share the assumption that the update is a static reparameterization of the base weights, which the input-dependent variants below relax.

Token-aware and gated adaptation. Making the low-rank update input-dependent at the token level has been explored recently. TopLoRA (Li et al., 2025) learns a token-dependent *diagonal* matrix between the LoRA factors and serves as the stateless prior-art baseline in this work. On top of this baseline, a *recurrent* token modulator (MaLoRA) propagates state across tokens, and state-space recurrence is also placed at the *context-segment* level, where it performs model-attached evidence selection over the frozen generator's own hidden states, a setting none of these token-level methods address. The token-level analysis (Section 3.4) shows that modulation there is *role-sensitive but relevance-blind*, which motivates moving evidence-relevance decisions to the segment level.

State-space models in language modeling. State-space models provide sub-quadratic sequence modeling (Gu et al., 2021; Smith et al., 2022; Gu and Dao, 2024; Dao and Gu, 2024). Two prior adapters apply such dynamics, and both place the recurrence *across layers* of the adapter stack: Nguyen et al. (2024) (READ) propagates state through the adapter stack across layers, and Yu et al. (2025) (SSMLoRA) uses an SSM to interconnect LoRA matrices inserted into the same module type at different layers. MaLoRA differs in *axis*: a selective state-space module is the *per-token gate* over the LoRA update, with recurrence *across tokens of the same input*, so that each token's adapter strength carries memory of the trajectory leading up to it.

Reasoning and evidence selection. Multi-step QA benchmarks (Ho et al., 2020; Trivedi et al., 2022) expose a gap between the language model's possession of the answer and its use of the supporting evidence; long-context evaluation surfaces related shifts (Hsieh et al., 2024; Bai et al., 2024; Laban et al., 2025).

Retrieval baselines. Most retrieval baselines used in open-domain and multi-hop QA are external modules. BM25 (Robertson and Zaragoza, 2009) is a sparse retriever, and dense or late-interaction systems such as DPR (Karpukhin et al., 2020), Contriever (Izacard et al., 2021), E5 (Wang et al., 2022), ColBERT (Khattab and Zaharia, 2020), and recent embedding encoders (Zhang et al., 2025) all train a separate retrieval model rather than reusing the downstream language model's hidden states. MaRA, the retrieval adapter introduced here, is instead *model-attached*: it scores segments from the intermediate hidden states of the same frozen backbone used for generation, then applies a small segment-level state-space module on top. It is therefore closer to a context-level adapter than to a conventional outsourced retriever.

3 MaLoRA: Stateful Token Modulation

3.1 Preliminaries

Low-rank adaptation. Low-rank adaptation (Hu et al., 2022) reparameterises the update to a frozen weight $W_0 \in \mathbb{R}^{d_{\text{out}} \times d_{\text{in}}}$ as a low-rank product. For an input $x \in \mathbb{R}^{d_{\text{in}}}$, the output becomes

$$h \;=\; \big(W_0 + BA\big)\, x, \quad A \in \mathbb{R}^{r \times d_{\text{in}}},\; B \in \mathbb{R}^{d_{\text{out}} \times r}, \tag{1}$$

where $r \ll \min(d_{\text{in}}, d_{\text{out}})$ is the rank. The base weight W_0 is held frozen; only A and B are trained. The same update is applied to every token of every input, so the magnitude of the adaptation is independent of the model's representation of the current token.

TopLoRA: stateless per-token modulation. TopLoRA (Li et al., 2025) relaxes this static assumption by inserting a learned per-token diagonal modulator $\Lambda(x_t) \in \mathbb{R}^{r \times r}_{>0}$ between the LoRA factors:

$$h_t \;=\; \big(W_0 \;+\; B\,\Lambda(x_t)\,A\big)\, x_t, \tag{2}$$

where $\Lambda(x_t) \;=\; \operatorname{diag}\big(\boldsymbol{\lambda}(x_t)\big)$ with $\boldsymbol{\lambda}(x_t) \;=\; \exp\!\big(\operatorname{RMSNorm}(Px_t)\big) \;\in\; \mathbb{R}^{r}_{>0}$ and $P \in \mathbb{R}^{r \times d_{\text{in}}}$ a learned low-rank projection (Fig. 1b). The modulator is *stateless*: $\Lambda(x_t)$ depends only on the current token, with no propagation across positions.

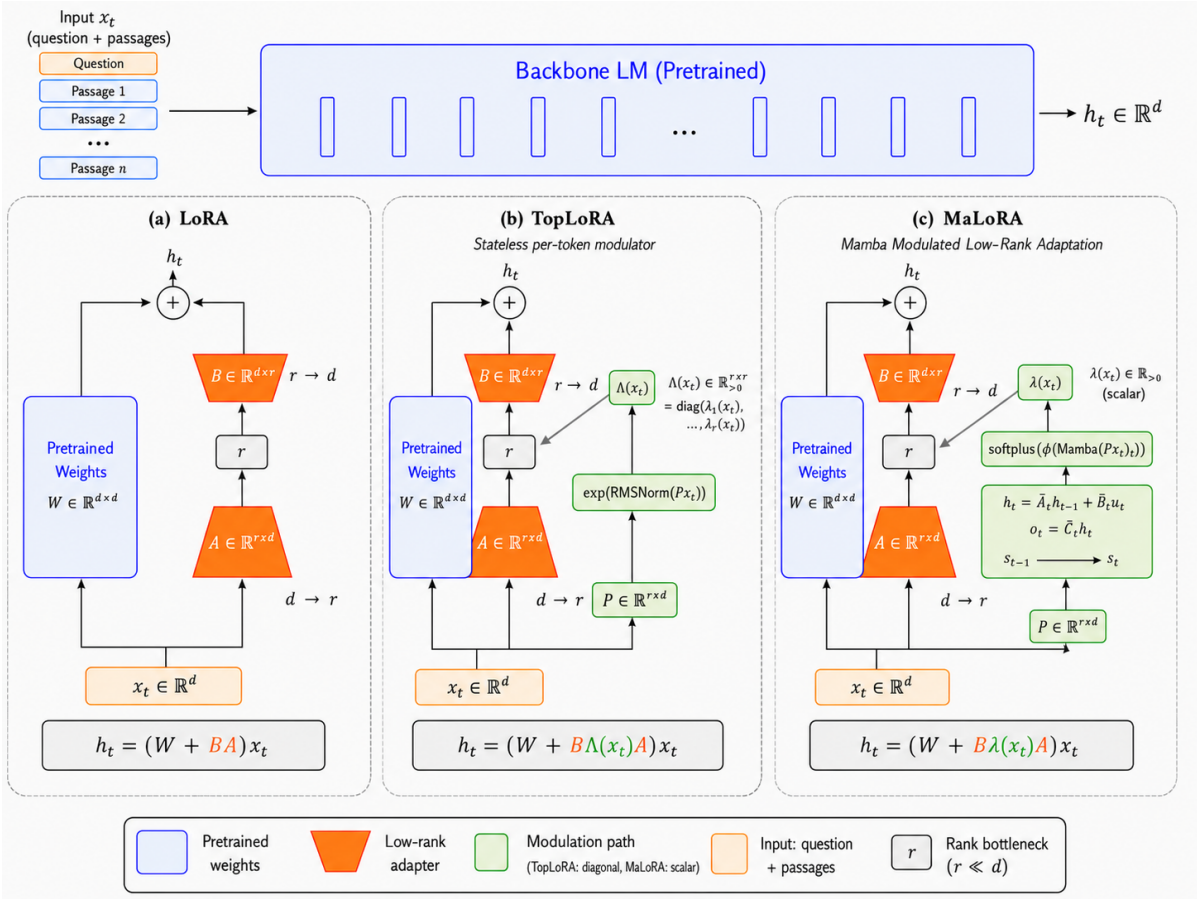

Figure 1: **From LoRA to MaLoRA. (a) LoRA**: a static low-rank update BAx_t adds to the frozen output Wx_t. **(b) TopLoRA (Li et al., 2025)**: a per-token diagonal modulator $\Lambda(x_t) = \mathrm{diag}\big(\exp(\mathrm{RMSNorm}(Px_t))\big)$ is inserted between the LoRA factors, yielding $h_t = (W + B\,\Lambda(x_t)\,A)\,x_t$. The modulator is stateless: $\Lambda(x_t)$ depends only on the current token. **(c) MaLoRA**: the per-token head is replaced by a selective state-space module (Mamba) over the projected sequence $\{Px_t\}$, and the output is a scalar $\lambda(x_t) = \mathrm{softplus}(\phi(\mathrm{Mamba}(Px_t)_t))$ that scales the r-dimensional intermediate.

3.2 MaLoRA

MaLoRA replaces TopLoRA's stateless per-token diagonal with a stateful scalar gate. The static low-rank update in Eq. 1 is multiplied by a per-token scalar $\lambda(x_t) \in \mathbb{R}_{>0}$ produced by a selective state-space module (Gu and Dao, 2024) over the projected sequence:

$$\begin{aligned} h_t &= \big(W_0 + B\,\lambda(x_t)\,A\big)\,x_t, \\ \lambda(x_t) &= \mathrm{softplus}\big(\phi(o_t)\big), \end{aligned} \tag{3}$$

where $P \in \mathbb{R}^{r\times d_{\text{in}}}$ is a learned low-rank projection sharing dimensions with A, ϕ is a small learned head, and o_t is the Mamba output at position t. The Mamba module maintains a hidden state $s_t \in \mathbb{R}^{d_s}$ with input-dependent dynamics:

$$s_t = \bar{A}_t\, s_{t-1} + \bar{B}_t\, Px_t, \quad o_t = \bar{C}_t\, s_t, \tag{4}$$

where $\bar{A}_t, \bar{B}_t, \bar{C}_t$ are computed from Px_t in the standard Mamba parameterisation (Fig. 1c). Eq. 3 reduces to Eq. 1 when $\lambda(x_t) = 1$.

The Mamba block is the central contribution: it brings selective state-space recurrence inside the parameter-efficient adapter itself, giving the per-token gate memory of the trajectory $x_1, \ldots, x_t$ rather than reacting only to x_t. The remaining design choices are as follows. **(i)** The output is a single scalar $\lambda(x_t)$ rather than TopLoRA's diagonal $\Lambda(x_t)$, which exposes one interpretable value per token for the behavioural analyses of Section 3.4. **(ii)** The activation is softplus so the modulator can both suppress ($\lambda < 1$) and amplify ($\lambda > 1$) the LoRA update; activation alternatives are ablated in Appendix D.

MaLoRA is trained jointly with the LoRA matrices, attached to $\{q_\mathrm{proj}, k_\mathrm{proj}, v_\mathrm{proj}, \mathrm{up_proj}, \mathrm{down_proj}\}$ with projection- and layer-specific parameters. The Mamba module adds approximately 50% to the

Table 1: **Token modulators across three backbones.** F1 on MuSiQue and 2WikiMultihopQA, and RULER QA-2 accuracy (%) at 4k context using the MuSiQue-trained checkpoints. Means over three seeds; per-seed std in Appendix D (Table 7).

BB	Method	MuSiQue F1	2Wiki F1	QA-2 Acc@4k
Qwen-7B	Base	14.4	26.0	*54.1*
	LoRA (Hu et al., 2022)	51.1	77.3	56.8
	TopLoRA (Li et al., 2025)	55.6	77.9	59.8
	MaLoRA	**56.4**	**79.3**	**60.7**
Llama-8B	Base	17.4	31.9	*10.8*
	LoRA (Hu et al., 2022)	54.4	78.5	60.9
	TopLoRA (Li et al., 2025)	61.4	80.4	63.0
	MaLoRA	**62.3**	**83.3**	**64.7**
Gemma-9B	Base	25.3	34.2	*38.2*
	LoRA (Hu et al., 2022)	59.8	77.2	64.4
	TopLoRA (Li et al., 2025)	59.5	79.3	65.4
	MaLoRA	**63.7**	**82.2**	**65.8**

LoRA parameter count; the TopLoRA baseline adds approximately 54% (its diagonal head is wider than MaLoRA's scalar output). Exact counts are in Appendix C.

3.3 Modulator results

Two multi-hop QA datasets are used: MuSiQue (Trivedi et al., 2022) (2–4 hops over 20 candidates) and 2WikiMultihopQA (Ho et al., 2020) (up to 4 hops over 10 candidates). All three frozen backbones (Qwen-2.5-7B, Llama-3.1-8B, Gemma-2-9B) are evaluated at rank r=16 with EM/F1, beam 4, over three seeds; full hyperparameters in Appendix B. MaLoRA improves over both LoRA and the TopLoRA baseline on every cell of Table 1, with the largest gains over LoRA on MuSiQue (up to +7.9 F1 on Llama). The recurrent-state contribution is dissected in Section 3.5, and the same pattern extends to commonsense reasoning (Appendix H).

3.4 Modulation behavior

The trained modulator is suppression-dominant: over 90% of $\lambda(x_t)$ fall below one, with structural separators and query tokens suppressed most heavily and content tokens retaining higher values (Fig. 2). A variance decomposition shows roughly 80% of $\lambda(x_t)$ variance is per-token rather than per-projection or per-layer, so the per-token component cannot be reproduced by any static rescaling; suppression concentrates in v_proj, q_proj, and the final transformer layer (Appendix E).

Table 2: **Mean $\lambda(x_t)$ by token region** (full MuSiQue validation, MaLoRA across three backbones). Mark. / Ques. / Supp. / Dist. = marker / question / supporting / distractor tokens. Discussion in Section 3.4.

BB	Mark.	Ques.	Supp.	Dist.
Qwen	0.792	0.772	0.904	0.904
Llama	0.725	0.603	0.708	0.709
Gemma	1.212	1.730	0.929	0.930

Mamba state usage. A direct check compares the Mamba block's stateful output $z_t = \mathrm{Mamba}(Px_{1:t})_t$ against a stateless counterfactual $z_t^{\text{stateless}} = \mathrm{Mamba}(Px_t)$ that processes each position independently. The relative discrepancy $\|z_t - z_t^{\text{stateless}}\|/\|z_t\|$, averaged across the six MaLoRA cells (Qwen/Llama/Gemma × MuSiQue/2WikiMultihopQA), rises from 0.13 at position 1 to 0.61–0.63 across mid-prompt positions and remains in that range out to the longest positions tested. The stateless variant therefore diverges from the stateful one by a non-trivial fraction at every position past the first token: the SSM state is genuinely accumulating information rather than acting as a decorative recurrence. Per-cell curves are in Appendix G.

Aggregating $\lambda(x_t)$ over prompt regions on the full validation set across the three MaLoRA-backbone cells (Table 2; extended discussion in Appendix F) confirms a clear role-based pattern: **the modulator differentiates tokens by their structural role**. Marker, query, and paragraph-content regions receive distinct λ values in every cell, and on Qwen and Llama, structural (marker) and query tokens are suppressed more than paragraph content (Gemma is the exception, amplifying marker and query tokens instead). Within paragraph content, however, the modulator is *relevance*-blind: supporting and distractor paragraphs receive almost identical λ, with differences below 0.002 in every cell. Token-level modulation thus reshapes how strongly the adapter applies to different prompt roles, which is a useful axis of adaptation, but evidence selection requires a separate mechanism at the segment level (Section 4).

3.5 From input-dependence to stateful input-dependence

The gain over LoRA can be decomposed along two axes. *Input-dependence* (TopLoRA, prior art):

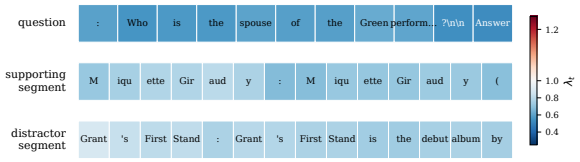

Figure 2: **Per-token modulation $\lambda(x_t)$ for MaLoRA on a MuSiQue example** (Llama-3.1-8B, seed-43). Cell colour averages $\lambda(x_t)$ over 160 modulated projections; darker = stronger suppression. Question tokens are suppressed most; supporting and distractor content retain higher λ. Per-region breakdown in Table 2.

replacing LoRA's static scaling with a learned per-token modulator already lifts F1 across most cells of Table 1 (up to +7.0 F1 on MuSiQue Llama), with no propagation of state across positions. *Stateful input-dependence* (MaLoRA, this work): a selective state-space module makes $\lambda(x_t)$ depend on the trajectory leading up to position t, and adds a further consistent lift over TopLoRA on every cell.

The same Mamba primitive is used at both granularities of adaptation. At the token level (MaLoRA, this section), it provides a consistent lift over stateless per-token modulation. At the segment level (next), it performs evidence selection, which the token-level modulator cannot do because it is relevance-blind at the paragraph level (Section 3.4).

4 MaRA: Context-Level Adaptation

MaRA operates one level above the token modulator: the input is split into context segments, each segment is scored by relevance to the query, and the top-k are exposed to the modulated language model for generation. The adapter is small ($\approx$ 3M trainable parameters) and introduces no separate retrieval encoder: it reads the frozen backbone's own intermediate hidden states and adds only a segment-level mixer, with the recurrence running over segment embeddings rather than tokens. The evidence information already resides in the backbone's hidden states; the segment-level state-space module tracks the reasoning chain across segments and surfaces the supporting ones, in contrast to the token-level modulator's paragraph-level relevance-blindness (Section 3.4). Full per-stage exposition is in Appendix I.

MaLoRA modifies the frozen model's weights with a per-token low-rank update. MaRA instead modifies the input's reasoning state: it adds no weight update and reads the backbone's hidden states to select which segments the model integrates. The two are adapters at different targets: one reshapes the weight update, the other reshapes the context and "hops" the model reasons over.

Method. Given a query Q and N segments $\mathcal{C} = (C_1, \ldots, C_N)$, the retrieval adapter proceeds in four stages.

(i) Segment encoder. Each C_i is concatenated with Q and passed through the first K=16 layers of the frozen backbone, producing hidden states $H_i \in \mathbb{R}^{T_i \times d}$. No new encoder parameters are introduced.

(ii) Attention pool. A learned query vector q_a attends over each segment's hidden states with an additive score, and the segment embedding is the attention-weighted sum of those hidden states. The hidden states are already query-conditioned, since each segment is encoded together with Q in stage (i). With $W_k \in \mathbb{R}^{d_p \times d}$,

$$p_i \;=\; \sum_{j=1}^{T_i} \mathrm{softmax}_j\big(q_a^\top \tanh(W_k H_{i,j})\big)\, H_{i,j}. \tag{5}$$

(iii) Mamba over segments. The segment embeddings are passed through a two-layer selective state-space module (Gu and Dao, 2024), with the recurrence running over *segments* (not tokens):

$$\begin{aligned} z_i \;&=\; W_{\text{in}}\, p_i, \\ (s_1, \ldots, s_N) \;&=\; \mathrm{Mamba}\big(z_1, \ldots, z_N\big), \end{aligned} \tag{6}$$

where W_{in} projects each pooled segment embedding to the recurrence width. The recurrence is causal: s_i depends on segments 1 through i. The unit of state evolution is one context segment, matching the granularity at which contextual transitions occur in multi-step reasoning (Section 3.5).

(iv) Scoring heads. Two linear heads read the recurrent states. A local head scores each segment from its own state s_i, and a global head reads the final state $g = s_N$ (which has integrated all N segments through the causal recurrence) and predicts all N labels jointly:

$$g \;=\; s_N, \qquad r_i \;=\; \mathbf{w}_\ell^\top s_i, \qquad \tilde{\mathbf{r}} \;=\; W_g\, g \;\in\; \mathbb{R}^N. \tag{7}$$

The per-segment score r_i is used to rank and select the top-k segments, which are forwarded to the modulator-equipped language model. The global head $\tilde{\mathbf{r}}$ supplies an auxiliary supervision signal that conditions the shared recurrence (Appendix I); it is not used at selection time.

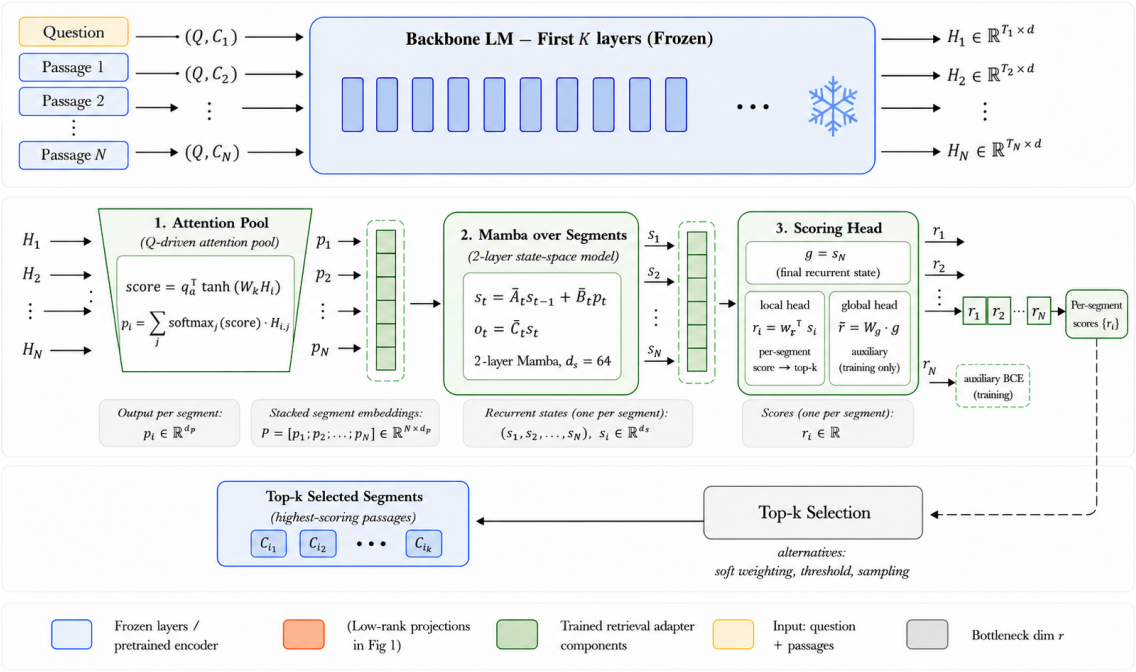

Figure 3: **MaRA architecture and data flow.** Each (query, segment) pair (Q, C_i) is encoded by the first K=16 layers of the frozen backbone, yielding hidden states H_i. A learned attention pool reduces each H_i to a single embedding p_i. A two-layer Mamba runs over the segment-level sequence $(p_1, \ldots, p_N)$, producing recurrent states $(s_1, \ldots, s_N)$. A local scoring head maps each recurrent state s_i to a relevance score r_i; a separate global head reads the final state $g = s_N$ and provides auxiliary supervision during training.

Training objective. Binary supporting-segment labels $y_i \in \{0, 1\}$ supervise the retrieval adapter. The loss combines weighted BCE on the local scores (positive class up-weighted to offset class imbalance), an intra-example pairwise margin that pushes supporting scores above non-supporting scores, and an auxiliary BCE on the global head:

$$\begin{aligned}\mathcal{L} \;=\; & \sum_i \mathrm{BCE}(r_i, y_i) \\ & + \beta \sum_{i:\, y_i=1} \sum_{j:\, y_j=0} \max(0,\, m - (r_i - r_j)) \\ & + \gamma \sum_i \mathrm{BCE}(\tilde{r}_i, y_i),\end{aligned} \tag{8}$$

where $\tilde{r}_i$ is the i-th logit of the global head (Eq. 7) and γ=0.5. Selection uses only the local scores r_i; the global term is an auxiliary regularizer on the shared recurrence. The retrieval adapter is trained with the adapter frozen; no end-to-end fine-tuning of the two stages is performed. Hyperparameters and the canonical training pool are detailed in Appendix B.

Iterative selection. At inference the retrieval adapter is applied over N passes. Each pass re-scores the candidate segments conditioned on the segments already selected, then appends the highest-scoring unselected segment to an ordered anchor set. The final top-k places these anchors first and fills the remainder by the last pass's scores. The number of passes is matched to the reasoning depth of each dataset: N=2 for 2WikiMultihopQA and N=4 for MuSiQue. The procedure is given in Appendix J.

4.1 MaRA architecture comparison

MaRA is compared to six alternatives at matched supervision and trainable budget, on MuSiQue and 2WikiMultihopQA with the base language model (no adapter). The alternatives ablate the per-segment pool (**MeanPool**, **LastTok**), the segment-level mixer (**Transformer**, **MLP**, **PoolOnly**), and the recurrence granularity (**TokenMamba**, recurrence over concatenated tokens instead of pooled segments). See Appendix I for definitions.

The Mamba over segments wins every cell. Two gaps are diagnostic. The gap to **MeanPool** (segment-level Mamba retained, learned attention pool replaced by a plain mean) is 7 pp on MuSiQue, isolating the contribution of the learned within-segment attention pool. The gap to **TokenMamba** (same Mamba module applied to concatenated to-

Table 3: **MaRA architecture comparison.** Recall@4 on MuSiQue and 2WikiMultihopQA validation splits, no adapter attached. Best per column in bold.

	MuSiQue		2WikiMultihopQA	
Architecture	Qwen	Llama	Qwen	Llama
MLP	20.6	19.6	93.8	95.3
LastTok	30.1	32.3	84.8	83.7
TokenMamba	61.4	54.8	82.5	76.1
PoolOnly	57.3	58.2	90.0	94.7
Transformer	63.8	60.5	95.5	95.2
MeanPool	73.8	76.8	98.6	98.1
Mamba (ours)	**80.6**	**84.1**	**99.2**	**99.5**

kens) is 19 to 29 pp on MuSiQue, isolating the contribution of placing the recurrence at the segment level rather than the token level. On 2WikiMultihopQA, every architecture beyond MLP and LastTok exceeds 90% R@4 and differences compress, but the Mamba still leads. Per-architecture analyses are in Appendix I.

Comparison to classical retrieval baselines. BM25 (Robertson and Zaragoza, 2009) reaches R@4=47.0 on MuSiQue and 66.4 on 2WikiMultihopQA. **Qwen3-Embedding-8B** (Zhang et al., 2025), an 8B-parameter dense retriever (larger than the backbone slice used here), reaches 75.5 on MuSiQue via zero-shot cosine similarity. MaRA reaches 80.6 and 99.2 respectively, above both and above every retrieval baseline in Appendix I (Table 10). BM25 is an external sparse scoring system and Qwen3-Embedding-8B is a separate dense encoder; in contrast, MaRA reuses the frozen generator's own hidden states and adds only a small segment-level mixer (Section 4). The advantage is not encoder size: it comes from training a small ($\approx$ 3M) segment-level mixer on those hidden states with a supervised objective tied to the downstream task.

5 Main Results

The full system combines the token-level modulator of Section 3 (MaLoRA) with MaRA of Section 4, both attached to a frozen low-rank adapter. MaRA restricts the language model's context to the top-k retrieved segments before generation. Training and evaluation follow the setup of Sections 3 and 4 on MuSiQue and 2WikiMultihopQA. All three backbones from Section 3 (Qwen-2.5-7B, Llama-3.1-8B, Gemma-2-9B) remain frozen. All adapters share the canonical training pool described in Appendix B. Evaluation uses full-context decoding for baselines and MaRA top-k for our system. The number of retrieved segments k is fixed once per dataset by a recall criterion on the validation supporting-paragraph labels, not by end-task F1, and is then held fixed across all backbones and methods. For 2WikiMultihopQA (10 candidates) the default k=4 already attains near-complete supporting-paragraph recall ($\approx$ 0.99 across backbones). MuSiQue has 20 candidates and harder multi-hop retrieval; MaRA's recall@k on the Qwen and Llama validation splits is 0.81 / 0.80 at k=4 and 0.92 / 0.93 at k=8 (both below 0.95), and we therefore set k=12, the smallest value exceeding 0.95 recall on both (0.97). The same k is held fixed for Gemma, whose router (safer training schedule, Appendix B) yields lower recall than Qwen/Llama; the per-backbone breakdown is in Appendix K. End-task scores are exact match (EM) and F1.

Table 4 reports the full system against the LoRA baseline across the grid. The MaLoRA+MaRA system improves over LoRA on every cell, by +4.0 to +9.3 F1: the largest margin is on MuSiQue Qwen (51.1 $\rightarrow$ 60.4, +9.3) and the smallest on MuSiQue Gemma (59.8 $\rightarrow$ 63.8, +4.0); on 2WikiMultihopQA the gains are +6.2/+6.4/+7.1 F1 across Qwen/Llama/Gemma. The mechanism analyses for the two components are in Section 3 (modulator alone, Table 1) and Section 4.1 (MaRA architecture, Table 3).

The two mechanisms are complementary across task difficulty. The component-decomposition rows (LoRA+MaRA for retrieval alone; the modulator-only columns of Table 1) show how the two mechanisms combine. On MuSiQue, the harder dataset (20 candidate paragraphs, multi-hop chains up to length 4), both mechanisms contribute substantially: modulation alone lifts LoRA by +5.3 F1 (Qwen, 51.1 $\rightarrow$ 56.4) and retrieval alone by +6.0 ($\rightarrow$ 57.1); the full system (60.4, +9.3 over LoRA) captures most of the sum but with approximately 18% overlap, indicating the two paths share some signal. The smallest full-system gain is on MuSiQue Gemma (+4.0 F1); Gemma's hidden states yield a less discriminating router than Qwen/Llama on this dataset (recall 0.93 vs. 0.97 at k=12, Appendix K), so retrieval contributes less of the lift. On 2WikiMultihopQA, the easier dataset (only 10 candidates, recall $\approx$ 0.99 at k=4), retrieval leads: retrieval alone lifts LoRA by +4.4 F1 (Qwen, 77.3 $\rightarrow$ 81.7), exceeding modulation

Table 4: **Main results and component decomposition.** EM / F1 across two multi-hop reasoning datasets and three frozen backbones. Baselines use the full candidate set; all retrieval-adapter rows use top-k retrieved paragraphs (k=12 on MuSiQue, k=4 on 2WikiMultihopQA, fixed by validation recall criterion). Best per column in bold. The LoRA+BM25, LoRA+BGE (fine-tuned), LoRA+MaRA, and TopLoRA+MaRA rows isolate the evidence-selection contribution under different adapter choices. The MaLoRA+oracle row uses gold supporting paragraphs as an upper bound.

	MuSiQue (EM / F1)			2WikiMultihopQA (EM / F1)		
Method	Qwen	Llama	Gemma	Qwen	Llama	Gemma
No adapter						
Base, full context	5.1/14.4	6.0/17.4	13.8/25.3	15.3/26.0	20.3/31.9	17.9/34.2
Static adapter baselines						
LoRA (Hu et al., 2022)	38.4/51.1	41.8/54.4	45.4/59.8	71.7/77.3	72.9/78.5	71.5/77.2
DoRA (Liu et al., 2024)	37.4/51.7	41.7/56.0	43.2/58.2	70.7/77.3	74.6/80.4	70.7/76.9
AdaLoRA (Zhang et al., 2023)	39.3/53.9	44.4/59.7	43.5/60.7	68.0/73.7	73.5/78.9	75.3/80.4
Token-level modulated adapters						
TopLoRA (Li et al., 2025)	41.5/55.6	47.3/61.4	43.8/59.5	73.0/77.9	75.1/80.4	72.9/79.3
MaLoRA	**42.6/56.4**	**48.1/62.3**	**48.7/63.7**	**73.7/79.3**	**78.1/83.3**	**77.1/82.2**
Modulator + Retrieval Adapter						
LoRA + BM25	28.2/40.5	29.6/41.1	34.1/45.8	51.3/56.3	51.5/56.6	50.0/55.4
LoRA + BGE-large (FT) (Xiao et al., 2024)	38.1/50.8	40.6/54.0	46.9/59.0	75.2/80.8	74.6/80.4	66.2/73.2
LoRA + MaRA	43.3/57.1	44.2/57.9	46.2/61.6	76.1/81.7	75.5/80.9	74.0/80.0
TopLoRA (Li et al., 2025) + MaRA	45.3/59.9	46.6/61.5	44.8/62.0	77.5/83.0	77.5/82.9	76.7/82.2
MaLoRA + MaRA	**46.0/60.4**	**47.2/62.4**	**47.6/63.8**	**78.0/83.5**	**79.8/84.9**	**78.4/84.3**
Modulator + Oracle						
MaLoRA + *oracle gold evidence*	52.6/67.5	57.0/71.2	57.2/72.8	78.4/83.4	80.9/86.0	79.2/84.3

alone (+2.0 F1, → 79.3). MaRA's effectiveness is bounded by how well the frozen backbone already discriminates evidence, which is easiest precisely when reasoning is shallow; the modulator operates at the weight level and is unaffected by retrieval difficulty. The full system is robust because the two mechanisms contribute through different channels and their gains largely accumulate, with some overlap (approximately 18% on the hardest cell). Although TopLoRA uses a more expressive per-rank diagonal gate and MaLoRA uses a scalar gate, MaLoRA still improves consistently in both standalone (Table 1) and combined (Table 4) settings, suggesting that recurrent state can compensate for lower per-token gate dimensionality.

Component decomposition (MuSiQue Qwen). On the cell with the most headroom, MaRA beats BM25 by +16.6 F1 (57.1 vs 40.5), confirming a large gain over classical sparse retrieval. The gap from MaLoRA+MaRA (60.4 F1) to MaLoRA+oracle (67.5 F1, +7.1 F1) shows that retrieval remains the main bottleneck on MuSiQue, even after model-attached retrieval improves over BM25.

Efficiency. Parameter overhead, training time, peak memory, and inference latency are detailed in Appendix C (Table 5).

6 Conclusion

A family of selective state-space adapters was introduced for frozen language models, operating at the token level (MaLoRA) and at the context-segment level (MaRA). Across three backbones and two multi-hop reasoning datasets, the combined system improved over LoRA on every cell of the 3×2 grid, with the two components contributing through complementary channels: at the segment level when the backbone can already discriminate evidence, and at the token level when reasoning depth gives the modulator more to act on.

Three directions extend this work. **(i)** The hierarchical recipe is expected to transfer to reasoning tasks beyond multi-hop QA (math, code) with a task-appropriate notion of segment. **(ii)** Multi-query settings (conversational agents, tool-use loops) have an explicit per-turn state and are a natural fit for MaRA, with the turn as the unit of state evolution. **(iii)** MaRA was used here only for segment selection; a natural extension is to inject its per-segment state directly into the generator through steering vectors or activation patching at chosen layers.

Limitations

Evaluation in this work is restricted to multi-hop reasoning question answering. Mathematical reasoning was attempted with a solution-step variant of the retrieval adapter and did not yield a clean lift; this is reported here as future work rather than a generalisation claim. Other reasoning settings, including code generation and conversational agents with evolving state, are similarly out of scope of the present empirical evaluation.

The backbones used here span the 7–9B parameter range. The granularity hypothesis is not tested on substantially smaller backbones (below 3B) or substantially larger backbones (above 30B). It is possible that the relative contributions of the modulator and the retrieval adapter change with backbone capacity; this remains to be measured.

The retrieval adapter is trained with supervised binary labels over the supporting paragraphs for each example. Settings without such labels, or with noisy labels, are out of scope. Distantly supervised or self-supervised variants of the retrieval objective would be needed to extend the method to such settings.

The training schedule of both the adapter and the retrieval adapter is sensitive to the choice of backbone. Gemma-2-9B required a smaller learning rate and a tighter gradient-norm clip than Qwen-2.5-7B and Llama-3.1-8B to train stably. Practitioners applying this method to a new backbone should expect to retune these two values. Each main-result cell consumed approximately 10 to 20 GPU-hours of training on a single A100-class device (Table 5). With five adapter methods across the 3×2 grid trained at three seeds, the full reported set required on the order of 1,000 GPU-hours, a non-trivial barrier to seed-replication at larger scales.

Training is staged rather than end-to-end. The modulated adapter (LoRA + MaLoRA) is trained first on the target task, and MaRA is then trained with the adapter and encoder both frozen. Joint end-to-end optimisation of the two stages is left as future work; we expect modest additional gains from co-adapting MaRA's segment selection to MaLoRA's modulated hidden states, but staged training is a clean fit for the model-attached design (MaRA reads frozen hidden states, so its training cost stays low) and we did not chase the joint-training variant here.

Ethics Statement

All datasets used in this work are publicly released benchmarks for multi-hop question answering. No human-subject data is collected or annotated. The backbones used (Qwen-2.5-7B, Llama-3.1-8B, Gemma-2-9B) are released under their respective licenses and used in accordance with them. No known dual-use concerns arise from the contributions of this paper: the modulator and retrieval adapter are general-purpose adaptation components and do not enable new harmful capabilities beyond those already present in the underlying language models. The compute footprint is documented in the Limitations section above.

Use of AI assistants. AI-based writing assistants (large language model chat assistants) were used for prose polishing, surfacing inconsistencies between sections, drafting LaTeX scaffolding for tables, and code-assisted error checking during the experimental pipeline (e.g., auditing eval configurations for mismatched checkpoints). All research design, methodological decisions, experiments, analyses, and final wording remain the responsibility of the human authors.

References

Yushi Bai, Xin Lv, Jiajie Zhang, Hongchang Lyu, Jiankai Tang, Zhidian Huang, Zhengxiao Du, Xiao Liu, Aohan Zeng, Lei Hou, Yuxiao Dong, Jie Tang, and Juanzi Li. 2024. LongBench: A bilingual, multitask benchmark for long context understanding. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 3119–3137, Bangkok, Thailand. Association for Computational Linguistics.

Tri Dao and Albert Gu. 2024. Transformers are SSMs: Generalized models and efficient algorithms through structured state space duality. In *Proceedings of the 41st International Conference on Machine Learning*, volume 235 of *Proceedings of Machine Learning Research*, pages 10041–10071. PMLR.

Shihan Dou, Enyu Zhou, Yan Liu, Songyang Gao, Wei Shen, Limao Xiong, Yuhao Zhou, Xiao Wang, Zhiheng Xi, Xiaoran Fan, Shiliang Pu, Jiang Zhu, Rui Zheng, Tao Gui, Qi Zhang, and Xuanjing Huang. 2024. LoRAMoE: Alleviating world knowledge forgetting in large language models via MoE-style plugin. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1932–1945, Bangkok, Thailand. Association for Computational Linguistics.

Chongyang Gao, Kezhen Chen, Jinmeng Rao, Ruibo Liu, Baochen Sun, Yawen Zhang, Daiyi Peng, Xi-

aoyuan Guo, and Vs Subrahmanian. 2025. MoLA: MoE LoRA with layer-wise expert allocation. In *Findings of the Association for Computational Linguistics: NAACL 2025*, pages 5112–5127, Albuquerque, New Mexico. Association for Computational Linguistics.

Albert Gu and Tri Dao. 2024. Mamba: Linear-time sequence modeling with selective state spaces. *Preprint*, arXiv:2312.00752.

Albert Gu, Karan Goel, and Christopher Ré. 2021. Efficiently modeling long sequences with structured state spaces. *CoRR*, abs/2111.00396.

Xanh Ho, Anh-Khoa Duong Nguyen, Saku Sugawara, and Akiko Aizawa. 2020. Constructing a multi-hop QA dataset for comprehensive evaluation of reasoning steps. In *Proceedings of the 28th International Conference on Computational Linguistics*, pages 6609–6625, Barcelona, Spain (Online). International Committee on Computational Linguistics.

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

Edward J Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, and Weizhu Chen. 2022. LoRA: Low-rank adaptation of large language models. *International Conference on Learning Representations (ICLR)*.

Zhiqiang Hu, Lei Wang, Yihuai Lan, Wanyu Xu, Ee-Peng Lim, Lidong Bing, Xing Xu, Soujanya Poria, and Roy Lee. 2023. LLM-adapters: An adapter family for parameter-efficient fine-tuning of large language models. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pages 5254–5276, Singapore. Association for Computational Linguistics.

Gautier Izacard, Mathilde Caron, Lucas Hosseini, Sebastian Riedel, Piotr Bojanowski, Armand Joulin, and Edouard Grave. 2021. Unsupervised dense information retrieval with contrastive learning. *arXiv preprint arXiv:2112.09118*.

Vladimir Karpukhin, Barlas Oguz, Sewon Min, Patrick Lewis, Ledell Wu, Sergey Edunov, Danqi Chen, and Wen-tau Yih. 2020. Dense passage retrieval for open-domain question answering. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 6769–6781, Online. Association for Computational Linguistics.

Omar Khattab and Matei Zaharia. 2020. Colbert: Efficient and effective passage search via contextualized late interaction over bert. In *Proceedings of the 43rd International ACM SIGIR conference on research and development in Information Retrieval*, pages 39–48.

Philippe Laban, Hiroaki Hayashi, Yingbo Zhou, and Jennifer Neville. 2025. Llms get lost in multi-turn conversation. *arXiv preprint arXiv:2505.06120*.

Shiwei Li, Xiandi Luo, Haozhao Wang, Xing Tang, Ziqiang Cui, Dugang Liu, Yuhua Li, Xiuqiang He, and Ruixuan Li. 2025. Beyond higher rank: Token-wise input-output projections for efficient low-rank adaptation. *Advances in Neural Information Processing Systems*, 38:136414–136435.

Shih-Yang Liu, Chien-Yi Wang, Hongxu Yin, Pavlo Molchanov, Yu-Chiang Frank Wang, Kwang-Ting Cheng, and Min-Hung Chen. 2024. Dora: Weight-decomposed low-rank adaptation. In *Forty-first International Conference on Machine Learning*.

John Nguyen, Sid Wang, Ke Li, and Carole-Jean Wu. 2024. Read: Recurrent adaptation of large transformers. *Preprint*, arXiv:2305.15348.

Rodrigo Nogueira and Kyunghyun Cho. 2020. Passage re-ranking with bert. *Preprint*, arXiv:1901.04085.

Stephen E. Robertson and Hugo Zaragoza. 2009. The probabilistic relevance framework: Bm25 and beyond. *Found. Trends Inf. Retr.*, 3:333–389.

Jimmy TH Smith, Andrew Warrington, and Scott W Linderman. 2022. Simplified state space layers for sequence modeling. *arXiv preprint arXiv:2208.04933*.

Chunlin Tian, Zhan Shi, Zhijiang Guo, Li Li, and Chengzhong Xu. 2024. Hydralora: An asymmetric lora architecture for efficient fine-tuning. *Advances in Neural Information Processing Systems*, 37:9565–9584.

Harsh Trivedi, Niranjan Balasubramanian, Tushar Khot, and Ashish Sabharwal. 2022. MuSiQue: Multi-hop questions via single-hop question composition. *Transactions of the Association for Computational Linguistics*, 10:539–554.

Liang Wang, Nan Yang, Xiaolong Huang, Binxing Jiao, Linjun Yang, Daxin Jiang, Rangan Majumder, and Furu Wei. 2022. Text embeddings by weakly-supervised contrastive pre-training. *arXiv preprint arXiv:2212.03533*.

Shitao Xiao, Zheng Liu, Peitian Zhang, Niklas Muennighoff, Defu Lian, and Jian-Yun Nie. 2024. C-pack: Packed resources for general chinese embeddings. In *Proceedings of the 47th international ACM SIGIR conference on research and development in information retrieval*, pages 641–649.

Jiayang Yu, Yihang Zhang, Bin Wang, Peiqin Lin, YongKang Liu, and Shi Feng. 2025. SSMLoRA: Enhancing low-rank adaptation with state space model. In *Proceedings of the 2025 Conference of the Nations of the Americas Chapter of the Association for Computational Linguistics: Human Language Technologies (Volume 1: Long Papers)*, pages 4493–4506, Albuquerque, New Mexico. Association for Computational Linguistics.

Qingru Zhang, Minshuo Chen, Alexander Bukharin, Nikos Karampatziakis, Pengcheng He, Yu Cheng, Weizhu Chen, and Tuo Zhao. 2023. Adalora: Adaptive budget allocation for parameter-efficient fine-tuning. *arXiv preprint arXiv:2303.10512*.

Yanzhao Zhang, Mingxin Li, Dingkun Long, Xin Zhang, Huan Lin, Baosong Yang, Pengjun Xie, An Yang, Dayiheng Liu, Junyang Lin, Fei Huang, and Jingren Zhou. 2025. Qwen3 embedding: Advancing text embedding and reranking through foundation models. *Preprint*, arXiv:2506.05176.

A System overview

Figure 4 places the two adapters of Sections 3 and 4 in a single view. The frozen backbone is shared. MaLoRA (left) operates at the token level: a Mamba block consumes the hidden state x_t and emits a scalar $\lambda(x_t)$ that rescales the LoRA update BAx_t inside every adapted projection, as in Eq. 3. MaRA (right) operates at the context level: a separate Mamba mixer runs over per-segment summaries obtained from the first K layers of the same backbone and scores each candidate paragraph, with the top-k retained for the answer pass. The two adapters act on different inputs (token states vs. segment summaries) and produce different outputs (a per-token modulation vs. a per-segment selection), so their gains compose rather than overlap (Section 5).

B Hyperparameter and training details

Datasets. MuSiQue (Trivedi et al., 2022) (denoted MQ) has 2-, 3-, and 4-hop compositional questions over 20 candidate paragraphs. 2WikiMultihopQA (Ho et al., 2020) (2Wiki) has compositional, comparison, inference, and bridge-comparison questions over 10 candidate paragraphs. The fraction of paragraphs that are supporting is small in both.

Canonical training pool. A single shared training pool of 10,000 examples per dataset is constructed for fair cross-backbone comparison. An example is admitted if the prompt length under both the Qwen and Llama tokenisers is at most the dataset's sequence-length budget (2048 for 2Wiki, 4096 for MQ), and the example must satisfy a chunk-detection check under both tokenisers. Remaining examples are sampled with stratification by hop count (MQ) or by question type (2Wiki) under a fixed random seed of 42. Every model is trained on the same pool, so cross-backbone differences are attributable to the model, not to the data subset.

Evaluation splits. End-task evaluation uses the released validation splits. 2WikiMultihopQA takes the first 3,000 examples (`dataset[:3000]`). MuSiQue uses the full answerable subset of its validation split (2,417 examples after filtering to `answerable=True`); see `eval_musique.py` for the procedure. All evaluations use beam search of width 4.

Adapter training (LoRA, modulator). LoRA rank r=16; the standard LoRA scaling is absorbed into the initialization of A and B (Eq. 1). Adapters are attached to $\{q_\text{proj}, k_\text{proj}, v_\text{proj}, \text{up_proj}, \text{down_proj}\}$. Training is 2 epochs with learning rate 2×10^{-4}, batch size 4, gradient-accumulation 2, weight decay 0, 100 warmup steps, bf16 mixed precision. The modulated variants (Section 3) use the same schedule, loss, and module set as LoRA. Modulator-only parameters add approximately 54% (TopLoRA) or approximately 50% (MaLoRA) over the LoRA parameter count; see Appendix C.

Adapter training on Gemma. Gemma-2-9B uses a safer schedule: learning rate 1×10^{-4}, $\max \|g\|_2 = 0.5$, 300 warmup steps. All other settings match the table above. This applies uniformly to LoRA, DoRA, AdaLoRA, and the modulated variants.

Retrieval adapter training. MaRA (Section 4) is trained for 2 epochs with batch size 1 and gradient-accumulation 4. Qwen and Llama backbones use learning rate 5×10^{-4} and $\max \|g\|_2 = 1.0$. The chunk encoder uses the first K=16 transformer layers of the backbone with frozen weights. The recurrence width is D=256, attention-pool dimension is 256, scoring-head MLP hidden dimension is 256, Mamba state dim d_s=64. Positive-class BCE weight is 8.0; pairwise margin loss weight 0.5 with margin 1.0. About 3.1M MaRA parameters per cell.

Retrieval adapter training on Gemma. MaRA on Gemma uses the safer schedule: learning rate 1×10^{-4}, $\max \|g\|_2 = 0.5$. Other settings unchanged.

Seeds and checkpoint selection. All cells (modulators and MaRA) are trained with three random

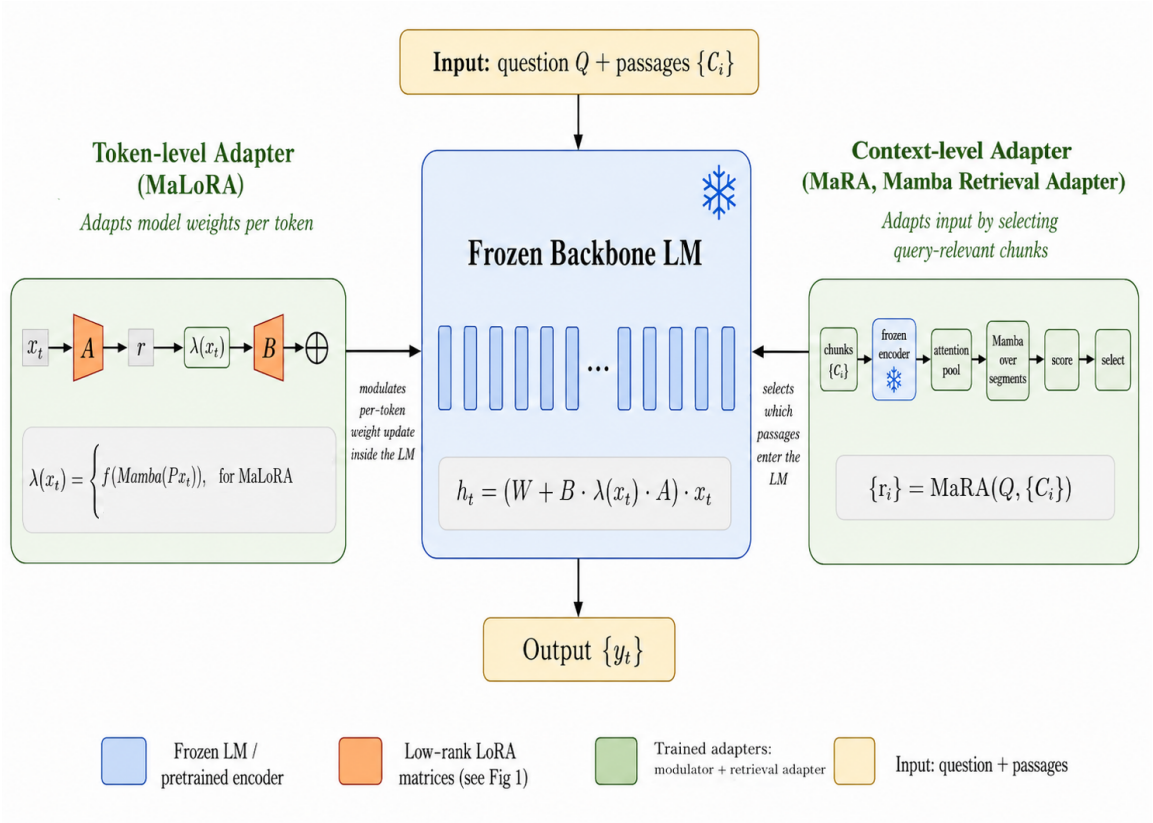

Figure 4: **Two-level selective state-space adaptation around a frozen LM.** Token-level adapter (MaLoRA, left) modulates the LoRA scaling factor per token; context-level adapter (MaRA, right) selects query-relevant input segments. Same frozen backbone, different granularities, different targets.

seeds (42, 43, 44). All headline numbers in Table 4 and Table 1 are means over the three seeds, with no best-seed selection. For analyses that operate on a single checkpoint (the state-usage probe of Appendix G, the per-projection / per-layer marginals of Appendix E, and the per-token region analysis of Appendix F), the seed with the highest validation F1 is used per cell; this is a methodological choice (these analyses require a specific checkpoint) and does not affect any reported headline metric.

C Efficiency breakdown: parameters, training time, peak memory, inference latency

This appendix combines the four cost dimensions of each adapter into a single view: **(i)** trainable parameter count, **(ii)** end-to-end training time, **(iii)** peak GPU memory during training, and **(iv)** inference latency. Table 5 reports each dimension averaged over the six cells of Table 4 (3 backbones $\times$ 2 datasets); per-cell numbers are in the supplementary CSV. Configs for every cell are committed under `configs/canonical_pool/`.

Adapter training (modulators against static baselines). DoRA is the most expensive among the static baselines: its weight-decomposed reparameterisation takes $1.88\times$ the LoRA training time on average (20.3 h vs. 10.8 h), and up to 30+ h per cell on MuSiQue Gemma. AdaLoRA is closer to LoRA at $1.03\times$. **Both token modulators are cheaper than DoRA at training:** TopLoRA trains at $0.95\times$ LoRA time, and MaLoRA at $1.07\times$ LoRA time despite adding a Mamba module. Peak GPU memory is within approximately 4% of LoRA for every method: peak training memory is dominated by the frozen backbone and activations, not by the adapter delta at rank $r{=}16$.

Retrieval adapter (MaRA) training. MaRA is trained as a separate stage with the language model frozen and adds only the segment-level mixer and scoring heads (approximately 3.1M parameters, $\approx$ 10% of the LoRA adapter). Average training time is $\approx$ 2.5 h per (backbone, dataset) cell, peak GPU memory $\approx$ 12 GB. Because MaRA reuses the first $K{=}16$ frozen layers of the backbone as encoder and trains only the small mixer on top, both its

training time and its memory footprint are well below those of the LoRA-side adapter.

Inference latency. At fixed beam width (4) and adapter rank (r=16), inference latency has two components.

(a) LM forward. The modulated language model's forward pass is dominated by the backbone and the dataset's input length, not by the adapter choice. Representative measurements on 2WikiMultihopQA (mean $\approx$ 1,000 input tokens, n=3000, beam 4): 0.70 s/sample on Qwen-2.5-7B, 0.62 s/sample on Llama-3.1-8B, 3.0 s/sample on Gemma-2-9B (Gemma is slower due to logit soft-capping). Across methods at fixed backbone the variance is below 5%: the modulator adds one scalar matmul per token plus, for MaLoRA, a d_s=16-state Mamba step, which is negligible relative to the backbone's matmuls.

(b) MaRA segment scoring (pre-pass). MaRA re-encodes the candidate segments through the first K=16 frozen layers of the backbone, once per iterative pass (Appendix J). The dataset's pass count and candidate count determine the cost: N=2 passes over M=10 candidates on 2WikiMultihopQA, N=4 passes over M=20 candidates on MuSiQue. Encoding $N \cdot M$ segments through K layers, with each segment roughly the size of a per-paragraph chunk of the prompt, is approximately $\frac{N \cdot M \cdot K}{L_{\text{full}}}$ of a single full LM forward, where L_{full} is the backbone's layer count. For Qwen-2.5-7B (L_{full}=28) the pre-pass adds roughly 15 to 25% to inference on 2WikiMultihopQA and 40 to 60% on MuSiQue; on Gemma the relative pre-pass cost is smaller because the LM forward is more expensive to begin with. This is the principal inference-time cost of the system, and the trade-off behind it (more passes $\rightarrow$ better retrieval recall on hard multi-hop, more inference compute) is reflected in the recall-vs-k curve in Appendix K.

The combined picture: among static baselines, MaLoRA matches the training cost of LoRA and AdaLoRA (1.07$\times$ vs. 1.03$\times$) while delivering a larger F1 gain (Table 4), and is 1.76$\times$ faster to train than DoRA. Adding MaRA on top costs a separate 2.5 h per cell with a small (12 GB) memory footprint, well below the main adapter's profile. At inference, adapter choice is essentially free at rank 16 and beam 4.

D Stateless modulator head: activation choice

This appendix isolates the per-token *stateless* modulator head — the scalar $\lambda(x_t) = f(\phi(Px_t))$ family without the Mamba recurrence — under three activation choices (softplus, sigmoid, tanh). Table 6 reports the matrix under the Table 1 protocol, full-context, on MuSiQue and 2WikiMultihopQA across all three backbones. This ablation does *not* include the Mamba recurrence; MaLoRA in the main paper is the softplus-scalar variant with a Mamba block added on top (Section 3.2). Its Table 1 numbers are higher than the corresponding stateless row on five of six cells; on Llama 2WikiMultihopQA the stateless variant is marginally higher (84.1 vs. 83.3), within the per-seed standard deviation ($\sigma \leq 0.9$ for Llama 2WikiMultihopQA in Table 7).

The three activations land within approximately 2 F1 of each other on every cell, with no consistent winner across backbones. Softplus is preferred over sigmoid because its unbounded positive range admits both suppression and amplification, the latter empirically observed for MaLoRA on Gemma-2-9B (Table 2, marker/question tokens with $\lambda > 1$); a sigmoid-bounded modulator ($\lambda \in (0, 1)$) cannot represent this amplification, and its near-equal end-task F1 reflects that on these benchmarks the modulator is suppression-dominated while the capacity to amplify is exercised mainly by the Gemma cell. A diagonal output form ($\boldsymbol{\lambda}(x_t) \in \mathbb{R}^r_{>0}$) was checked on MuSiQue Qwen (58.4 F1 sigmoid-diag, 54.3 tanh-diag vs. 54.2/55.2 scalar) and shows no clear win, motivating the simpler scalar form for the behavioural analyses of Section 3.4.

Per-seed variance of the three token modulators. All headline numbers in Table 4 are computed as means over three random seeds (42, 43, 44): the static baselines (LoRA, DoRA, AdaLoRA), the token modulators (TopLoRA, MaLoRA), and the modulator-plus-MaRA rows. Table 7 reports the corresponding per-seed standard deviations for the token modulators. Per-cell variance is small across all three token modulators ($\sigma \leq 2.6$ F1 in every cell).

E Per-projection and per-layer modulation marginals

Figure 5 reports the mean of $\lambda(x_t)$ broken down by projection type and by transformer layer for

Table 5: **Efficiency breakdown.** Trainable parameters, training time (mean over 3 backbones $\times$ 2 datasets), peak GPU memory during training, and average F1 gain over LoRA across the same six cells. "$\times$LoRA" is the ratio to the LoRA baseline's training time on the same six cells. DoRA's $1.88\times$ training time is the most expensive baseline; MaLoRA stays within $1.07\times$ despite adding a recurrent state-space module. MaRA is trained separately with the language model frozen and adds ≈ 3.1M parameters and ≈ 2.5 h training per cell.

Method	Params (M)	ΔLoRA	Train (h)	Train $\times$LoRA	Peak GPU (GB)
Static adapter baselines					
LoRA (Hu et al., 2022)	30.7	–	10.8	1.00$\times$	33.8
AdaLoRA (Zhang et al., 2023)	46.1	+50%	11.1	1.03$\times$	34.0
DoRA (Liu et al., 2024)	31.6	+3%	**20.3**	**1.88**$\times$	33.8
Token modulators					
+ TopLoRA (Li et al., 2025)	47.4	+54%	10.3	0.95$\times$	35.2
+ MaLoRA	46.2	+50%	11.6	1.07$\times$	34.3
Retrieval adapter (separate stage, language model frozen)					
+ MaRA (alone)	3.1	+10%	2.5	0.23$\times$	12.0

Table 6: **Stateless modulator-head activation ablation** (F1, full context, n=3000, beam 4; scalar output form). The per-token head is a linear ϕ over Px_t with no recurrence. †Tanh on MuSiQue Gemma-2-9B fails to learn (F1 26.3 vs. $\geq$ 63.2 for the other activations); the same tanh head reaches healthy F1 on 2WikiMultihopQA Gemma (83.1) and on Qwen/Llama.

	MuSiQue F1			2Wiki F1		
Activation	Qw	Ll	Ge	Qw	Ll	Ge
softplus	55.1	61.7	63.2	78.5	84.1	81.8
sigmoid	54.2	63.0	64.5	79.0	83.9	83.7
tanh	55.2	61.8	**26.3**†	78.4	83.0	83.1

Table 7: **Per-seed F1 standard deviation** (over 3 seeds: 42, 43, 44).

	MuSiQue			2Wiki		
Method	Qw	Ll	Ge	Qw	Ll	Ge
TopLoRA (Li et al., 2025)	2.5	2.4	2.6	0.8	0.9	2.2
MaLoRA	1.1	1.8	2.3	0.5	0.7	2.1

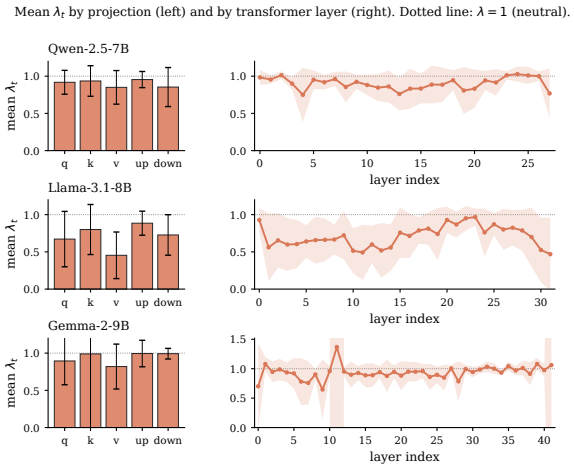

Figure 5: **Where the modulator suppresses.** Mean $\lambda(x_t)$ by projection type (left) and transformer layer (right), aggregated over the full MuSiQue validation set. Three cells: MaLoRA on Qwen-2.5-7B, Llama-3.1-8B, and Gemma-2-9B.

the three MaLoRA-backbone cells: Qwen-2.5-7B, Llama-3.1-8B, and Gemma-2-9B. The figure is referenced from Section 3.4. Suppression concentrates in v_proj, q_proj, and the final transformer layer across all three cells, consistent across backbones.

F Modulation by token role: extended discussion

The per-region means of $\lambda(x_t)$ reported in Table 2 (main body) are aggregated over the full MuSiQue validation set, across the three MaLoRA-backbone cells. Tokens are tagged as *marker* (format/separator tokens), *question* (the query), *para_support* / *para_distractor* (supporting vs. non-supporting paragraph content), and *context-other*.

Two findings are robust. First, λ **on supporting and distractor paragraphs is statistically identical in every cell** ($|\Delta| \leq 0.002$): the modulator does not separate relevant from irrelevant evidence. This is expected, since identifying supporting evidence requires the multi-hop reasoning that the modulator precedes; evidence selection is instead handled by MaRA (Section 4). Second, the modulator instead allocates modulation by **token role**: on Qwen and Llama, structural (marker) and query tokens are suppressed more than paragraph content, consistent with the per-token view in Figure 2. The exception is MaLoRA on Gemma-2-9B, which *amplifies* query and marker tokens ($\lambda \approx 1.2$–1.7)

rather than suppressing them, a backbone-specific behavior. The shared pattern (relevance-blind, role-sensitive) explains why the modulator and MaRA are non-redundant: the modulator reshapes the low-rank update by token role and cannot perform evidence selection, so MaRA contributes an orthogonal capability.

G State-usage probe: per-cell results

This appendix expands the state-usage check from Section 3.4. For each Mamba block in a trained MaLoRA cell, the block's full stateful output $z_t = \mathrm{Mamba}(Px_{1:t})_t$ is compared against a stateless variant $z_t^{\text{stateless}}$ in which each position is processed independently with the recurrent state zeroed. The relative discrepancy is

$$\rho_t = \frac{\left\| z_t - z_t^{\text{stateless}} \right\|_2}{\| z_t \|_2}.$$

A value of $\rho_t \approx 0$ at every position would indicate that the recurrent state contributes nothing and that the Mamba block could be replaced by a stateless head; $\rho_t > 0$ means the state genuinely accumulates information up to position t. The probe is run on 5 validation samples and 10 randomly sampled MambaBlocks per cell, on the six canonical MaLoRA cells, using the locked best-seed checkpoints from the manifest. Per-cell mean ρ_t at sampled positions $t \in \{1, 10, 10^2, 10^3\}$ is in Table 8. The aggregate across the six cells rises from 0.13 at position 1 to 0.61–0.63 across the rest of the prompt, with every cell above 0.45 from t=10 onward; the recurrence carries useful trajectory-level information rather than acting decoratively.

Table 8: **State-usage probe: mean ρ_t by token position.** Averaged over 10 sampled Mamba blocks and 5 validation prompts; larger ρ_t = more state contribution.

DS	BB	t=1	t=10	t=10^2	t=10^3
MQ	Qwen	0.18	0.55	0.56	0.62
MQ	Llama	0.10	0.60	0.61	0.65
MQ	Gemma	0.09	0.76	0.69	0.61
2Wiki	Qwen	0.23	0.59	0.64	0.56
2Wiki	Llama	0.05	0.47	0.53	0.52
2Wiki	Gemma	0.14	0.71	0.78	0.78
Mean (6 cells)		0.13	0.61	0.63	0.62

H Commonsense reasoning

Commonsense reasoning is a complementary single-context testbed: each example is a self-contained multiple-choice question, with no candidate paragraphs and no multi-hop chain to retrieve. MaRA does not apply and is omitted; only the token-level modulators are compared to LoRA. Adapters are trained on the standard 10,000-example Commonsense suite (Hu et al., 2023) and use the same rank r=16 and schedule as the main experiments. Table 9 reports the mean accuracy over eight multiple-choice tasks (BoolQ, PIQA, SIQA, HellaSwag, WinoGrande, ARC-easy, ARC-challenge, OpenBookQA).

Table 9: **Commonsense accuracy** (%, mean over eight multiple-choice tasks). Single-context; MaRA omitted. Best non-base per column in bold.

Method	Qwen	Llama	Gemma
Base	77.1	67.6	68.7
LoRA	88.3	80.6	**86.6**
TopLoRA	86.8	79.7	84.5
MaLoRA	**88.5**	**83.3**	86.2

On Llama-3.1-8B MaLoRA improves over LoRA (+2.7) while TopLoRA is slightly below (−0.9); on Qwen-2.5-7B MaLoRA is level with LoRA (+0.2) and TopLoRA is below (−1.5); on Gemma-2-9B all three fine-tuned variants are tightly clustered (LoRA 86.6, TopLoRA 84.5, MaLoRA 86.2; all +15.8 to +17.9 over the 68.7 base). Across backbones MaLoRA wins on average (86.0 vs. LoRA 85.2 vs. TopLoRA 83.7). Commonsense reasoning is largely saturated for these backbones once fine-tuned, so per-cell differences are small relative to the multi-hop gains in Table 4 and consistent with little headroom for token-level reshaping in the absence of multi-paragraph distractor structure.

I MaRA architecture details

This appendix expands the condensed Section 4 with per-stage discussion, the architecture-ablation deep dive, and the full classical-retrieval comparison.

I.1 Per-stage notes

Segment encoder. The frozen encoder is the same backbone used downstream, truncated to its first K=16 layers. Each segment C_i is concatenated with the query Q before encoding, so the resulting hidden states are already query-conditioned. No new encoder parameters are introduced. Encoding each (query, segment) pair independently is the standard cross-encoder pattern.

Q-driven attention pool. The attention pool of Eq. 5 reduces each segment's T_i-step hidden-state sequence to a single embedding p_i via additive attention with a learned query vector q_a. An ablation replacing the learned attention pool by a plain mean is reported in Table 3 as MeanPool.

Mamba over segments. The two-layer selective state-space module (Gu and Dao, 2024) of Eq. 6 follows the same SSM update as Eq. 4, but the recurrence runs over *segments* rather than tokens. This is the key architectural choice and is ablated against TokenMamba (token-level Mamba) and Transformer (segment-level transformer) in Table 3.

Scoring heads. Eq. 7 uses two linear heads: a local head $\mathbf{w}_\ell$ that scores each segment from its own recurrent state s_i (used for top-k selection), and a global head W_g that reads the final state $g = s_N$ and predicts all N supporting labels jointly as an auxiliary signal. There is no per-segment MLP combiner; global context reaches each score through the causal recurrence (each s_i depends on segments 1 through i) and through the shared parameters that the global-head objective conditions. The recurrence's contribution is isolated by the PoolOnly ablation (mixer removed) and by the MLP ablation (per-segment MLP without recurrence) in Table 3.

Auxiliary supervision. The global-head term in Eq. 8 (γ=0.5) is an auxiliary BCE over all N paragraphs, predicted jointly from the final state $g = s_N$. It shapes the shared recurrence but does not feed selection, and its contribution to final recall is small.

Composition with the modulator. MaRA operates before the modulated language model's forward pass; the modulator (Section 3) operates inside it. The two share only the language model's hidden states as a representation substrate. Training is staged: the modulated adapter is trained first on the target task; MaRA is then trained with the adapter and encoder both frozen.

I.2 Architecture ablation (full analysis)

The Mamba over segments wins on every cell of Table 3. The R@4 gap to the second-best architecture (MeanPool, which keeps segment-level Mamba but drops the attention pool) is 7 pp on both Qwen and Llama for MuSiQue. TokenMamba uses the same Mamba module as MaRA but applied to the concatenated token sequence; its R@4 is 19 to 29 pp below the segment-level Mamba on MuSiQue, isolating the contribution of placing the recurrence at the segment level. The encoder follows the standard cross-encoder pattern, encoding each (query, segment) pair independently, with a trained Mamba mixer on top of the per-segment embeddings carrying the cross-segment interaction (Karpukhin et al., 2020; Nogueira and Cho, 2020). On 2WikiMultihopQA, where the candidate set is smaller and the supporting evidence is less distractor-rich, every architecture beyond MLP and LastTok reaches $\geq 90\%$ R@4 and the differences between architectures are compressed, but the Mamba still leads on every cell.

I.3 Scoring-head ablation: linear vs. MLP

The local scoring head (Eq. 7) is a single linear map $\mathbf{w}_\ell$. Replacing it with a two-layer MLP (Linear $\rightarrow$ GELU $\rightarrow$ Linear, hidden 256), with the Mamba mixer and all other settings fixed, does not help: Recall@4 is 78.6 vs. 80.6 (MuSiQue Qwen), 83.2 vs. 84.1 (MuSiQue Llama), 99.4 vs. 99.2 (2WikiMultihopQA Qwen), and 99.4 vs. 99.5 (2WikiMultihopQA Llama). The MLP head is marginally worse on MuSiQue and indistinguishable on 2WikiMultihopQA. The recurrent state s_i already integrates cross-segment context, so a linear read-out suffices; the extra head capacity adds parameters without improving retrieval. The linear head is used throughout.

I.4 Retrieval baselines (full comparison)

MaRA is compared against a 2×2 grid of alternatives, crossing *outsourced* (a separate encoder over raw text) versus *model-attached* (reusing the frozen backbone's own hidden states), and *untrained* (off-the-shelf or cosine similarity) versus *trained* (fine-tuned on the supporting-paragraph supervision). Table 10 reports Recall@4 on the same n=500 validation split and gold labels for every method.

Three readings. First, MaRA wins every cell: on MuSiQue it reaches 80.6 (Qwen) and 84.1 (Llama), above the strongest outsourced baseline, the 8B Qwen3-Embedding encoder (75.5), despite using no separate encoder and only $\approx$ 3M new parameters. Second, training a *separate* dense bi-encoder on the supporting-paragraph labels is not enough: across three encoders (E5, Contriever, BGE-large-en), fine-tuning lifts 2WikiMultihopQA to 96.9–97.7 but plateaus on MuSiQue at 63.3–

Table 10: **Retrieval comparison** (Recall@4, n=500 validation, identical examples and gold labels). Outsourced methods run a separate encoder over raw text; model-attached methods score from the frozen backbone's own hidden states. Trained = fine-tuned on the supporting-paragraph labels. Best per column in bold. MaRA beats every outsourced baseline, including the 8B Qwen3-Embedding encoder, while adding only $\approx$ 3M parameters and no separate encoder.

Method	MuSiQue	2WikiMultihopQA
Outsourced, off-the-shelf (untrained)		
BM25 (sparse) (Robertson and Zaragoza, 2009)	47.0	66.4
E5-large-v2 (Wang et al., 2022)	69.1	83.7
Qwen3-Embedding-8B (Zhang et al., 2025)	75.5	83.5
Outsourced, trained (dense bi-encoder, best of LR sweep)		
E5-large-v2 (fine-tuned) (Wang et al., 2022)	63.3	97.1
Contriever (fine-tuned) (Izacard et al., 2021)	68.3	96.9
BGE-large-en (fine-tuned) (Xiao et al., 2024)	68.2	97.7
Model-attached, untrained (frozen-LM cosine)		
Frozen-LM cosine (Qwen-2.5-7B)	37.7	54.9
Frozen-LM cosine (Llama-3.1-8B)	48.4	48.2
Model-attached, trained (ours)		
MaRA (Qwen-2.5-7B)	80.6	99.2
MaRA (Llama-3.1-8B)	**84.1**	**99.5**

68.3, well below MaRA (80.6–84.1). For E5 the MuSiQue fine-tune even *degrades* relative to its own zero-shot (69.1 $\rightarrow$ 63.3), a catastrophic-forgetting effect when a general-purpose encoder is specialised on a small label set. The MuSiQue ceiling for text-only retrieval reflects that the bridge entity is absent from the surface query, which a separate encoder cannot recover but the backbone's reasoning hidden states can. Third, the untrained model-attached baseline (raw frozen-LM cosine) is weak (37.7–54.9), so the gain is not simply from reusing the backbone's hidden states; it comes from the trained segment-level mixer on top of them. BM25 R@8/R@12 reach 60.8/71.5 (MuSiQue) and 86.2/100 (2WikiMultihopQA); Qwen3-Embedding R@8/R@12 reach 88.9/93.9 and 96.4/100.

Algorithm 1 Iterative segment selection

Input: query Q, segments $\mathcal{C} = \{C_1, \ldots, C_M\}$, passes N, budget k, adapter R
Output: ordered segment indices, truncated to k
$A \leftarrow [\,]$ {ordered anchor list}
for $t = 1$ **to** N **do**
  $\mathbf{r} \leftarrow R(\mathcal{C}, Q \mid A)$ {score each segment, conditioned on selected anchors}
  $a_t \leftarrow \arg\max_{i \notin A} r_i$
  $A \leftarrow A \,\|\, [\,a_t\,]$
end for
rest $\leftarrow$ sort$(\{i : i \notin A\})$ by r_i descending {$\mathbf{r}$ from the final pass}
return $(A \,\|\, \text{rest})[:k]$

J Iterative segment selection

MaRA (Section 4) is applied iteratively at inference. Given query Q, candidate segments $\mathcal{C} = \{C_1, \ldots, C_M\}$, a pass budget N, and a selection budget k, an ordered anchor set is built one segment per pass; each pass conditions the segment scoring on the anchors already chosen (Algorithm 1). The two-pass form (N=2) is used for 2WikiMultihopQA and the four-pass form (N=4) for MuSiQue, matching the deeper multi-hop chains of the latter. During training, the intermediate passes 1:$N-1$ receive an auxiliary loss weight and the final pass weight 1.0.

K Top-k selection sweep

The headline k per dataset is fixed by a validation recall criterion (Section 5): the smallest k whose MaRA recall over the validation supporting-paragraph labels exceeds 0.95 on Qwen and Llama. The same k is then held fixed for every backbone and method. Per-backbone MaRA recall@k is shown in Table 11. On MuSiQue, Qwen and Llama cross the threshold at k=12 (recall 0.97); Gemma's router (safer training schedule, Appendix B) is lower across all k but the same k=12 is kept for cross-backbone consistency. On 2WikiMultihopQA every backbone reaches $\approx$ 0.99 recall at

k=4.

Table 11: **MaRA supporting-paragraph recall@k by backbone** (validation split). Threshold 0.95 is satisfied at k=12 on Qwen and Llama for MuSiQue, and at k=4 on every backbone for 2WikiMultihopQA. The same k is held fixed for Gemma.

	MuSiQue R@k			2Wiki R@k	
Backbone	k=4	k=8	**k=12**	**k=4**	k=8
Qwen	0.81	0.92	**0.97**	**0.99**	1.00
Llama	0.80	0.93	**0.97**	**0.99**	1.00
Gemma	0.68	0.84	**0.93**	**0.99**	1.00