MANTA: LINEARIZING LARGE LANGUAGE MODELS WITH MULTI-CENTER TAYLOR ATTENTION

Anonymous authors
Paper under double-blind review

ABSTRACT

Softmax attention gives large language models flexible access to the preceding context, but its quadratic computation and growing key–value cache limit efficient long-context deployment. Converting a pretrained softmax Transformer into a fixed-state model offers a practical alternative to pretraining a new architecture from scratch. However, Taylor-based linearization typically expands all keys around one global center. This approximation is inaccurate when pretrained keys form broad, anisotropic distributions. We introduce MANTA (**M**ulti-Center Linear **A**tte**N**tion via **T**aylor **A**pproximation), which uses query-aware routing to make the softmax approximation local. MANTA maintains an additive recurrent state for each routed group, while the total state size remains independent of context length. It projects the memory-dominant matrix in each state onto a learned low-rank basis to reduce memory. Finally, joint normalization combines the compressed history with sliding-window attention. Experiments show that MANTA recovers the language-modeling quality of the original softmax model, with an average score of 73.2 across six benchmarks. Compared with the softmax baseline, MANTA improves decoding efficiency and achieves up to 27.3$\times$ the throughput. By making the Taylor approximation local and memory-efficient, MANTA provides a practical path from pretrained softmax Transformers to accurate fixed-state models with linear total computation and memory independent of context length.

1 INTRODUCTION

Softmax attention is central to modern large language models because it allows each query to retrieve and reweight information from the entire preceding context (Vaswani et al., 2017). This flexibility becomes increasingly expensive at long sequence lengths. Evaluating all query–key pairs requires $O(N^2)$ computation (Dao et al., 2022), while autoregressive decoding retains a key–value (KV) cache that grows linearly with the context (Kwon et al., 2023). Linear attention and related recurrent models instead summarize the history in a fixed-size state, enabling $O(N)$ total computation and memory that is constant with respect to context length (Katharopoulos et al., 2020; Choromanski et al., 2020; Sun et al., 2023; Yang et al., 2024; 2025; Team et al., 2025). However, pretraining a new architecture from scratch remains costly. Converting an existing softmax Transformer into a recurrent model offers a more practical route to efficient deployment (Mercat et al., 2024; Wang et al., 2024; Zhang et al., 2025; Lan et al., 2025; McDermott et al., 2025; Van Nguyen et al., 2026; Meng et al., 2026; Kuzina et al., 2026). The central challenge is to preserve the behavior encoded by pretrained softmax attention under the constraints of a fixed recurrent state.

Kernelized linear attention approximates the exponential softmax kernel with feature maps (Katharopoulos et al., 2020; Choromanski et al., 2020; Zhang et al., 2024a). A complementary approach is to derive a recurrent computation directly from softmax. In particular, the mean-centered first-order Taylor expansion studied by Kuzina et al. (2026) yields an additive fixed-state readout. The construction nevertheless expands every visible key around a single global mean. A first-order Taylor approximation is accurate only locally, and its leading omitted term grows quadratically with the query-aligned score displacement from the expansion point. This locality matters for pretrained models. Our analysis of Llama-3.1-8B reveals broad, anisotropic key distributions with separated or elongated high-density regions rather than a compact cloud around the mean. A global center can therefore be far from many of the keys that it is intended to represent. This suggests that the

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.

main limitation is not the fixed-state formulation itself, but the use of a single expansion center for a geometrically heterogeneous key distribution.

We introduce MANTA (**M**ulti-Center Linear **A**tte**N**tion via **T**aylor **A**pproximation), an adaptive multi-center Taylor attention mechanism. It preserves the fixed-state efficiency of linear attention while making the softmax approximation local. MANTA softly routes normalized pre-RoPE keys among multiple centers. Its routing metric is derived from the query second moment, so it emphasizes key directions that most strongly affect attention scores. Each center maintains its own additive statistics. For each group, the routing-weighted mean of the post-RoPE keys serves as the Taylor expansion point. Multiple centers increase the recurrent-state size. We reduce this overhead by projecting the state matrix that dominates recurrent-state memory into a learned low-rank basis shared by all centers within each KV head. Additionally, following prior work (Beltagy et al., 2020; Xiao et al., 2024; Arora et al., 2024; Lan et al., 2025; Kuzina et al., 2026), we retain exact softmax attention for sink and recent tokens. Finally, MANTA jointly normalizes the exact and compressed branches. This design recovers full softmax whenever the history summaries are exact. The resulting layer retains $O(N)$ total computation and recurrent memory independent of context length.

Our contributions can be summarized as follows:

- We identify the single expansion center as a concrete bottleneck in Taylor-based softmax linearization. We relate the leading approximation error to query-aligned key displacement and show that pretrained keys occupy broad, anisotropic regions that are poorly represented by one global center.
- We propose query-aware multi-center Taylor attention. An entropy-regularized routing objective uses the query second moment to emphasize directions that affect attention scores. Clusterwise additive statistics then provide local Taylor readouts while preserving the recurrent structure of linear attention.
- We make the multi-center construction practical with learned low-rank state compression and exact attention over sink and recent tokens. A partition-preserving joint normalization combines their unnormalized outputs and masses, recovering full softmax when the history summaries are exact.
- Experiments show that MANTA recovers the language-modeling quality of the original softmax model and remains effective on long-context tasks. It reduces recurrent-state memory and improves decoding throughput. Ablations confirm the contributions of query-aware routing, low-rank compression, and joint normalization.

2 RELATED WORK

2.1 LINEAR ATTENTION AND RECURRENT SEQUENCE MODELS

Linear attention avoids materializing the pairwise softmax attention matrix by using either a decomposable kernel or a recurrent state. Kernel-based methods replace the exponential kernel with feature maps, as in Linear Transformers and Performers (Katharopoulos et al., 2020; Choromanski et al., 2020; Qin et al., 2023). Other methods develop more expressive recurrent updates (Gu & Dao, 2023; Dao & Gu, 2024; Katsch, 2023; Peng et al., 2024; Qin et al., 2024; Zhang et al., 2024b). RetNet introduces retention with data-dependent decay (Sun et al., 2023), Gated Linear Attention uses input-dependent forgetting (Yang et al., 2024), and Gated Delta Networks add a key-dependent delta rule (Yang et al., 2025). Kimi Linear further combines gated and delta-style updates in a scalable architecture (Team et al., 2025). These recurrent architectures were introduced for pretraining from scratch rather than for converting pretrained softmax Transformers. Consequently, they do not provide a direct way to reuse the parameters and capabilities of the many softmax models already available. In contrast, MANTA focuses on converting an existing softmax model into a fixed-state linear model.

2.2 CONVERTING PRETRAINED TRANSFORMERS TO LINEAR MODELS

Converting an existing Transformer avoids the cost of pretraining a new linear architecture (Kasai et al., 2021). Prior work learns feature maps or attention-transfer objectives to mimic the original attention (Zhang et al., 2024a; Mercat et al., 2024; Chen et al., 2024a; Zhang et al., 2025), distills

pretrained Transformers into recurrent models (Wang et al., 2024; Bick et al., 2024), or reuses pretrained projections and adapts a small number of parameters (Lan et al., 2025). Recent methods also retain a limited exact-attention cache to compensate for information that is difficult to store in a fixed recurrent state. LoLA combines low-rank linear attention with sparse caching (McDermott et al., 2025), while Lizard and STILL select tokens for hybrid attention (Van Nguyen et al., 2026; Meng et al., 2026). Most conversion methods begin with a chosen linear mechanism and then use training or exact token retention to reduce its gap from softmax. MANTA instead derives its recurrent construction from a local approximation of softmax.

3 PRELIMINARIES

3.1 SOFTMAX ATTENTION

Consider a causal self-attention head applied to an input sequence $\mathbf{X} \in \mathbb{R}^{N\times D}$, where N is the sequence length and D is the input dimension. The query, key, and value projections are $\mathbf{Q} = \mathbf{X}\mathbf{W}_Q$, $\mathbf{K} = \mathbf{X}\mathbf{W}_K$, and $\mathbf{V} = \mathbf{X}\mathbf{W}_V$, where the learnable projection weights $\mathbf{W}_Q, \mathbf{W}_K \in \mathbb{R}^{D\times d}$ and $\mathbf{W}_V \in \mathbb{R}^{D\times d_v}$ map each input to its per-head query, key, and value dimensions. In autoregressive modeling, a causal mask $\mathbf{M}$ prevents position t from attending to future positions $i > t$. The parallel attention computation used during training is

$$\mathbf{O} = \mathrm{softmax}_{\mathrm{row}}\Big(\mathbf{Q}\mathbf{K}^\top/\sqrt{d} + \mathbf{M}\Big)\,\mathbf{V}. \tag{1}$$

The t-th rows of $\mathbf{Q}$, $\mathbf{K}$, and $\mathbf{V}$ are respectively $\mathbf{q}_t^\top$, $\mathbf{k}_t^\top$, and $\mathbf{v}_t^\top$. During autoregressive inference, Equation 1 is evaluated one row at a time. For the visible positions $\mathcal{H}_t$, the row-t output can be written as

$$\mathbf{o}_t^{\mathrm{softmax}} = \sum_{i\in\mathcal{H}_t} \exp(\mathbf{q}_t^\top \mathbf{k}_i/\sqrt{d})\mathbf{v}_i \Big/ \sum_{i\in\mathcal{H}_t} \exp(\mathbf{q}_t^\top \mathbf{k}_i/\sqrt{d}). \tag{2}$$

Computing $\mathbf{o}_t^{\mathrm{softmax}}$ requires all preceding keys and values, so the decoder must retain them in a key–value (KV) cache. For a sequence of length N, this cache uses $O(N(d + d_v))$ memory. Evaluating the readout for all positions has $O(N^2)$ computation because each position compares with all visible keys.

3.2 LINEAR ATTENTION

Linear attention approximates the exponential operation in softmax attention with a dot-product kernel (Katharopoulos et al., 2020). Specifically, let $\phi : \mathbb{R}^d \to \mathbb{R}^r$ satisfy $\exp(\mathbf{q}^\top\mathbf{k}/\sqrt{d}) \approx \phi(\mathbf{q})^\top\phi(\mathbf{k})$. The causal readout then becomes

$$\mathbf{o}_t^{\mathrm{lin}} = \sum_{i\in\mathcal{H}_t} \phi(\mathbf{q}_t)^\top\phi(\mathbf{k}_i)\mathbf{v}_i \Big/ \sum_{i\in\mathcal{H}_t} \phi(\mathbf{q}_t)^\top\phi(\mathbf{k}_i) = \phi(\mathbf{q}_t)^\top\mathbf{S}_t / \phi(\mathbf{q}_t)^\top\mathbf{z}_t, \tag{3}$$

where $\mathbf{S}_t = \sum_{i\in\mathcal{H}_t} \phi(\mathbf{k}_i)\mathbf{v}_i^\top$ and $\mathbf{z}_t = \sum_{i\in\mathcal{H}_t} \phi(\mathbf{k}_i)$. These statistics obey the recurrent updates $\mathbf{S}_t = \mathbf{S}_{t-1} + \phi(\mathbf{k}_t)\mathbf{v}_t^\top$ and $\mathbf{z}_t = \mathbf{z}_{t-1} + \phi(\mathbf{k}_t)$. For fixed feature and value dimensions, each update and readout has constant cost with respect to N, so all N positions require $O(N)$ computation. The state $(\mathbf{S}_t, \mathbf{z}_t)$ has $O(rd_v + r)$ entries independent of N, giving constant recurrent memory.

3.3 FIRST-ORDER TAYLOR APPROXIMATION AS LINEAR ATTENTION

The same fixed-state form can be obtained directly by a first-order expansion of the softmax weights. Kuzina et al. (2026) use the following mean-centered form.

Lemma 1 (Mean-centered first-order softmax expansion)**.** *Let* $\mathbf{p}(\mathbf{z}) = \mathrm{softmax}(\mathbf{z})$ *for* $\mathbf{z} \in \mathbb{R}^m$*, and let* $\bar{z} = m^{-1}\sum_{j=1}^m z_j$*. For every* i*,*

$$p_i(\mathbf{z}) = \frac{1}{m} + \frac{z_i - \bar{z}}{m} + r_i(\mathbf{z}), \qquad \|\mathbf{r}(\mathbf{z})\|_\infty = O\big(\|\mathbf{z} - \bar{z}\mathbf{1}\|_2^2\big) \quad \textit{as } \|\mathbf{z} - \bar{z}\mathbf{1}\|_2 \to 0. \tag{4}$$

Applying Lemma 1 to the scores $z_i = \mathbf{q}_t^\top \mathbf{k}_i/\sqrt{d}$ in Eq. 2 and dropping $r_i(\mathbf{z})$ gives

$$\widehat{\mathbf{o}}_t = \frac{1}{n_t} \sum_{i \in \mathcal{H}_t} \left(1 + \frac{\mathbf{q}_t^\top (\mathbf{k}_i - \boldsymbol{\mu}_t)}{\sqrt{d}}\right) \mathbf{v}_i = \frac{\mathbf{A}_t^\top \mathbf{q}_t/\sqrt{d} + (1 - a_t)\mathbf{V}_t}{n_t}. \tag{5}$$

Here $n_t = |\mathcal{H}_t|$ is the number of visible tokens, $\mathbf{K}_t = \sum_{i \in \mathcal{H}_t} \mathbf{k}_i$ the key sum, $\boldsymbol{\mu}_t = \mathbf{K}_t/n_t$ the mean key, $\mathbf{V}_t = \sum_{i \in \mathcal{H}_t} \mathbf{v}_i$ the value sum, $\mathbf{A}_t = \sum_{i \in \mathcal{H}_t} \mathbf{k}_i \mathbf{v}_i^\top$ the key-value cross-moment, and $a_t = \mathbf{q}_t^\top \boldsymbol{\mu}_t/\sqrt{d}$ the mean-key score. The statistics $(n_t, \mathbf{K}_t, \mathbf{V}_t, \mathbf{A}_t)$ admit additive online updates, so Equation 5 has the same additive fixed-state structure as the linear-attention construction in Section 3.2.

4 METHOD

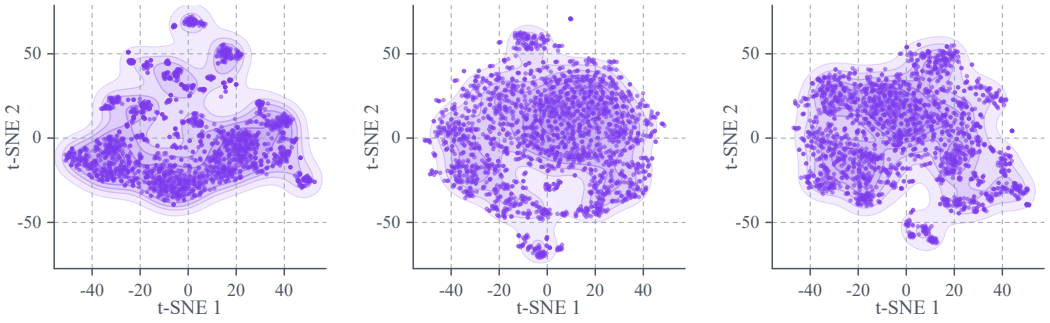
4.1 MOTIVATION: LIMITS OF A SINGLE TAYLOR CENTER

In Section 3.3, we described how Kuzina et al. (2026) expand all visible keys around their single mean key with a first-order Taylor approximation. Let $\mathcal{H}_t = \{i_1, i_2, \ldots, i_{n_t}\}$ list the visible positions in sequence order. For the current query, define the centered score-deviation vector as

$$\boldsymbol{\delta}_t = \frac{1}{\sqrt{d}} \left[\mathbf{q}_t^\top (\mathbf{k}_{i_1} - \boldsymbol{\mu}_t),\ \mathbf{q}_t^\top (\mathbf{k}_{i_2} - \boldsymbol{\mu}_t),\ \ldots,\ \mathbf{q}_t^\top (\mathbf{k}_{i_{n_t}} - \boldsymbol{\mu}_t)\right]^\top \in \mathbb{R}^{n_t}. \tag{6}$$

By Lemma 1, the leading omitted term is $O(\|\boldsymbol{\delta}_t\|_2^2)$, so large query-aligned displacements from $\boldsymbol{\mu}_t$ make the first-order approximation less faithful to softmax. A single global mean is therefore a poor expansion point when the key distribution contains separated or anisotropic regions.

To make this limitation concrete, Figure 1 visualizes real pre-RoPE keys from Llama-3.1-8B for one KV head at an early, middle, and late layer. All three panels occupy a broad region and contain separated or elongated high-density neighborhoods rather than one compact cloud around a global mean, showing that a single global Taylor center does not capture the geometry of real keys.



(a) Early layer ($\ell = 4$) (b) Middle layer ($\ell = 16$) (c) Late layer ($\ell = 28$)

Figure 1: t-SNE visualizations of Llama-3.1-8B pre-RoPE keys from KV head 0 at early, middle, and late layers.

4.2 QUERY-AWARE MULTI-CENTER TAYLOR ATTENTION

To address this limitation, we replace the single global Taylor center with multiple local centers. Each normalized pre-RoPE key is softly routed to these centers using a metric estimated from the query distribution, and each routed group maintains its own Taylor state. This keeps each expansion local to a portion of the key distribution.

Query-aware routing. Let $\mathbf{x}_i$ be a normalized pre-RoPE key, let $\mathbf{c}_j$ be a candidate routing center, and let $g_{i,j}$ be the assignment weight of $\mathbf{x}_i$ to center j. For a normalized pre-RoPE query $\mathbf{q}$, the query-aligned score displacement from $\mathbf{x}_i$ to $\mathbf{c}_j$ is $\mathbf{q}^\top(\mathbf{x}_i - \mathbf{c}_j)/\sqrt{d}$. Lemma 1 shows that the first

omitted Taylor term is quadratic in this displacement. An ordinary Euclidean distance would treat every key direction equally, even though only directions aligned with the queries affect this error. We therefore measure the displacement with the query second moment $\mathbf{M} = \mathbb{E}[\mathbf{q}\mathbf{q}^\top] \succ 0$ and define the query-weighted squared Taylor radius by

$$D^2_{\mathbf{M}}(\mathbf{x}_i, \mathbf{c}_j) = \frac{(\mathbf{x}_i - \mathbf{c}_j)^\top \mathbf{M}(\mathbf{x}_i - \mathbf{c}_j)}{d} = \mathbb{E}_{\mathbf{q}}\left[\left(\frac{\mathbf{q}^\top(\mathbf{x}_i - \mathbf{c}_j)}{\sqrt{d}}\right)^2\right]. \tag{7}$$

Thus $D^2_{\mathbf{M}}$ is the expected quadratic factor of the first neglected Taylor term. We choose the centers and assignment weights by minimizing the following entropy-regularized surrogate for this Taylor error over assignments $\mathbf{g}_i \in \Delta^{R-1}$ and centers $\mathbf{c}_j$:

$$\mathcal{J}(\mathbf{g}, \mathbf{c}) = \sum_{i,j} g_{i,j} D^2_{\mathbf{M}}(\mathbf{x}_i, \mathbf{c}_j) + \tau \sum_{i,j} g_{i,j} \log g_{i,j}. \tag{8}$$

The first sum penalizes a large expected squared score displacement. By Eq. 7, this is the quadratic factor of the leading neglected Taylor term, so minimizing it favors centers with smaller local approximation error. The second sum favors soft assignments. For a fixed token, $\sum_j g_{i,j} \log g_{i,j}$ is more negative when its mass is spread across centers and equals zero for a one-hot assignment. The temperature τ controls how strongly this preference competes with the radius term.

Proposition 1 (Minimizers of the routing surrogate)**.** *For fixed centers and $\tau > 0$, the minimizer of Eq. 8 over $\mathbf{g}_i \in \Delta^{R-1}$ is $g^\star_{i,j} = \exp(-D^2_{\mathbf{M}}(\mathbf{x}_i, \mathbf{c}_j)/\tau) / \sum_{a=1}^{R} \exp(-D^2_{\mathbf{M}}(\mathbf{x}_i, \mathbf{c}_a)/\tau)$. These weights satisfy $g^\star_{i,j} > 0$ and $\sum_j g^\star_{i,j} = 1$. For fixed assignments, the entropy term is independent of the centers. Each nonempty center therefore has the minimizer $\mathbf{c}^\star_j = \sum_i g_{i,j}\mathbf{x}_i / \sum_i g_{i,j}$.*

The population moment $\mathbf{M}$ in $D^2_{\mathbf{M}}$ is not available directly, and a finite-sample estimate can be ill-conditioned. We therefore replace $D^2_{\mathbf{M}}$ in the router with a regularized empirical distance. To obtain it, collect the normalized pre-RoPE queries in a set $\mathcal{Q}$ over all query heads sharing a KV head and compute $\mathbf{M}_0 = |\mathcal{Q}|^{-1} \sum_{\mathbf{q}\in\mathcal{Q}} \mathbf{q}\mathbf{q}^\top$. Shrink this empirical moment toward an isotropic matrix:

$$\widetilde{\mathbf{M}}_q = (1 - \lambda)\mathbf{M}_0 + \lambda \frac{\mathrm{tr}(\mathbf{M}_0)}{d}\mathbf{I}. \tag{9}$$

We set $\lambda = 0.05$ and use $\widetilde{\mathbf{M}}_q$ as the regularized estimate of $\mathbf{M}$ in the router. To compute the resulting distance with a fixed linear map, factor the metric as

$$D^2(\mathbf{x}_i, \mathbf{c}_j) = \|\mathbf{L}(\mathbf{x}_i - \mathbf{c}_j)\|_2^2, \qquad \mathbf{L}^\top\mathbf{L} = \widetilde{\mathbf{M}}_q/d. \tag{10}$$

Routing is performed before RoPE so that the assignments reflect token content rather than an absolute-position rotation. During initialization, we collect pre-RoPE key vectors from a small calibration set of representative input sequences. We transform these calibration keys by $\mathbf{L}$, initialize the prototypes with K-means++ and refine them with Lloyd updates (Arthur & Vassilvitskii, 2007). We set τ to the median squared distance to the nearest prototype. Subsequent language-model training adapts the prototypes and temperatures to the downstream objective.

Clusterwise expansion. For the history $\mathcal{H}_t$, we maintain, for every cluster j,

$$n_{t,j} = \sum_{i\in\mathcal{H}_t} g_{i,j}, \quad \mathbf{K}_{t,j} = \sum_{i\in\mathcal{H}_t} g_{i,j}\mathbf{k}_i, \quad \mathbf{V}_{t,j} = \sum_{i\in\mathcal{H}_t} g_{i,j}\mathbf{v}_i, \quad \mathbf{A}_{t,j} = \sum_{i\in\mathcal{H}_t} g_{i,j}\mathbf{k}_i\mathbf{v}_i^\top. \tag{11}$$

Routing and Taylor expansion use different representations for different purposes. Routing uses the pre-RoPE key $\mathbf{x}_i$ and fixed prototypes $\mathbf{c}_j$. Keeping the prototypes fixed makes each assignment position-independent, so the assignments can be computed once and assembled with a parallel prefix scan. The Taylor scores instead approximate the actual post-RoPE score $\mathbf{q}_t^\top\mathbf{k}_i/\sqrt{d}$, so the expansion center must reflect the post-RoPE keys visible at position t. We therefore accumulate the routed keys and use $\boldsymbol{\mu}_{t,j} = \mathbf{K}_{t,j}/n_{t,j}$ as the center for cluster j. Repeating the derivation of Eq. 5 within each cluster gives

$$a_{t,j} = \frac{\mathbf{q}_t^\top\boldsymbol{\mu}_{t,j}}{\sqrt{d}}, \qquad \ell_{t,j} = a_{t,j} + \log n_{t,j}, \qquad \mathbf{u}_{t,j} = \frac{\mathbf{A}_{t,j}^\top\mathbf{q}_t/\sqrt{d} + (1 - a_{t,j})\mathbf{V}_{t,j}}{n_{t,j}}. \tag{12}$$

Here $e^{\ell_{t,j}}$ is the approximate unnormalized mass and $\mathbf{u}_{t,j}$ is the conditional value for cluster j. We aggregate the clusters as $\widehat{\mathbf{o}}_t^{\text{hist}} = \sum_{j=1}^{R} e^{\ell_{t,j}} \mathbf{u}_{t,j} / \sum_{j=1}^{R} e^{\ell_{t,j}}$. Algorithm 1 in Appendix B summarizes the one-step decoding procedure.

4.3 Low-rank recurrent states and two-stage training

Multiple centers improve the approximation but replicate the recurrent state across clusters. Within each state, the matrix $\mathbf{A}_{t,j} \in \mathbb{R}^{d \times d_v}$ is the dominant storage term. The resulting uncompressed multi-center state contains $R(1 + d + d_v + dd_v)$ scalars per KV head. With Llama-3.1-8B dimensions and $R = 4$, this is 2.03 MiB per layer and live request, compared with 0.50 MiB for the reference Liger recurrent state (Lan et al., 2025). Because this cache must be maintained separately for every active sequence and read at every decoding step, it creates both capacity and memory-bandwidth pressure.

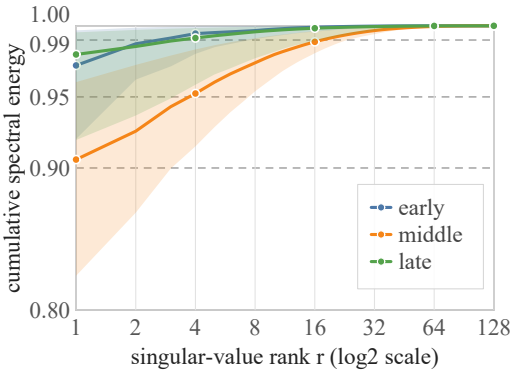


Figure 2: Cumulative singular-value energy of $\mathbf{A}_{t,j}$ in early, middle, and late layers, with 10th–90th percentile bands across heads and clusters.

The matrices are nevertheless strongly compressible, as illustrated in Figure 2. In an 8K Llama-3.1-8B diagnostic, the median ranks required for 90%, 95%, and 99% spectral energy are only 1, 2, and 9, respectively, despite an algebraic dimension of 128. We therefore reduce the recurrent-state memory by projecting each $\mathbf{A}_{t,j}$ onto a rank-r subspace. Specifically, we learn a fixed basis $\mathbf{P} \in \mathbb{R}^{d \times r}$ with $\mathbf{P}^\top \mathbf{P} = \mathbf{I}_r$. The basis is shared by all clusters of a KV head, and $\mathbf{A}_{t,j}$ is replaced by

$$\mathbf{z}_i = \mathbf{P}^\top \mathbf{k}_i, \qquad \mathbf{H}_{t,j} = \sum_{i \in \mathcal{H}_t} g_{i,j} \mathbf{z}_i \mathbf{v}_i^\top \in \mathbb{R}^{r \times d_v}, \qquad \mathbf{A}_{t,j}^\top \mathbf{q}_t \approx \mathbf{H}_{t,j}^\top (\mathbf{P}^\top \mathbf{q}_t). \tag{13}$$

Sharing $\mathbf{P}$ lets us compute $\mathbf{P}^\top \mathbf{q}_t$ and $\mathbf{P}^\top \mathbf{k}_t$ once per head. The projected state has $R(1+d+d_v+rd_v)$ scalars. At $R = 4$ and $r = 16$, it occupies 0.28 MiB per layer and request, a 7.22× reduction from dense MANTA and below the reference Liger recurrent state.

Proposition 2 (Optimal rank-r projection). *Let $\mathcal{A}_h$ be a collection of dense matrices $\mathbf{A} \in \mathbb{R}^{d \times d_v}$ for KV head h, and define $\mathbf{\Sigma}_h = \sum_{\mathbf{A} \in \mathcal{A}_h} \mathbf{A}\mathbf{A}^\top$. Under the constraint $\mathbf{P}^\top \mathbf{P} = \mathbf{I}_r$, the reconstruction objective $\sum_{\mathbf{A} \in \mathcal{A}_h} \|\mathbf{A} - \mathbf{P}\mathbf{P}^\top \mathbf{A}\|_F^2$ is minimized by any $\mathbf{P}$ whose columns form an orthonormal basis of the top-r eigenspace of $\mathbf{\Sigma}_h$.*

Proposition 2 gives a data-dependent initialization: form $\mathbf{\Sigma}_h$ from the collected dense matrices and initialize $\mathbf{P}$ with its leading r eigenvectors. Training then proceeds in two stages. In the first stage, we train the dense multi-center model and router and collect $\mathcal{A}_h$ from the resulting trajectories. We then initialize $\mathbf{P}$ using the procedure above. In the second stage, we replace $\mathbf{A}_{t,j}$ with $\mathbf{H}_{t,j}$, freeze the router, and train $\mathbf{P}$ together with the Q/K/V/O LoRA parameters. A reduced QR step after each update maintains $\mathbf{P}^\top \mathbf{P} = \mathbf{I}_r$.

4.4 A complete layer: compressed history plus exact SWA

Sliding-window attention (SWA) is widely used in methods that replace softmax attention with linear attention (Arora et al., 2024; Lan et al., 2025; Kuzina et al., 2026). For each query, SWA evaluates exact causal softmax only over a prescribed subset of visible tokens. The size of this subset is fixed, so its per-token computation and KV storage do not grow with the context length. At position t, our exact-attention set $\mathcal{E}_t$ contains the first S sink tokens and the most recent W tokens (Kuzina et al., 2026). The remaining visible positions form $\mathcal{F}_t = \mathcal{H}_t \setminus \mathcal{E}_t$, which we call the *compressed-history set*. These positions are summarized by the MANTA recurrent states. We retain the recent window because short-range attention can be highly position-sensitive and sharply concentrated. We retain the prefix because these positions often act as attention sinks and receive substantial mass from later queries. Figure 3 summarizes the complete layer and visualizes this token partition.

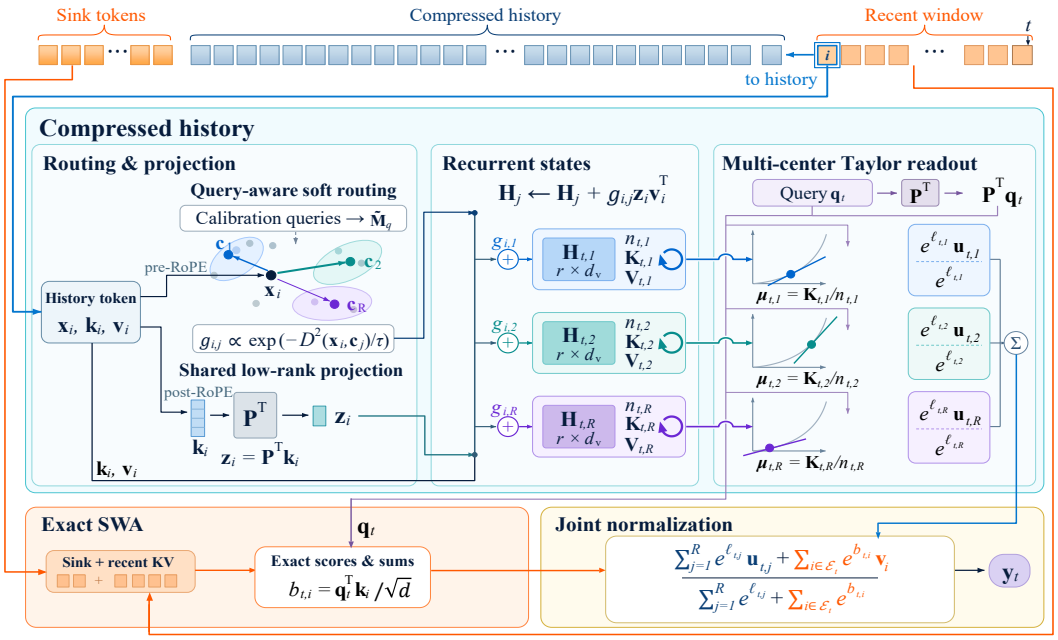


Figure 3: Overview of a MANTA layer, including token partitioning, the exact SWA and compressed-history branches, and joint normalization.

Existing intra-layer hybrids often normalize the SWA and linear-attention branches independently, then blend their outputs using fixed mixing weights (Lan et al., 2025). We instead add the two branches' unnormalized weighted-value sums and partition masses before a single division. With $b_{t,i} = \mathbf{q}_t^\top \mathbf{k}_i/\sqrt{d}$, this gives $\mathbf{y}_t = \frac{\sum_{j=1}^{R} e^{\ell_{t,j}} \mathbf{u}_{t,j} + \sum_{i\in\mathcal{E}_t} e^{b_{t,i}} \mathbf{v}_i}{\sum_{j=1}^{R} e^{\ell_{t,j}} + \sum_{i\in\mathcal{E}_t} e^{b_{t,i}}}$. This joint normalization recovers full softmax whenever the $\mathcal{F}_t$ summaries are exact. In contrast, independently normalizing the two branches discards their partition masses. Even when the $\mathcal{F}_t$ summaries are exact, such a blend recovers full softmax only when its mixing weight equals the query-dependent ratio of exact-attention mass to total mass.

5 EXPERIMENTS

5.1 MAIN RESULTS

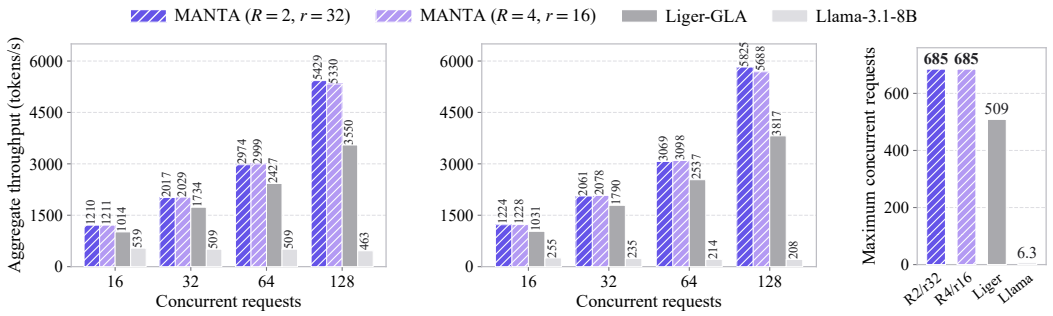
Performance. We evaluate whether linearization preserves the downstream accuracy of Llama-3.1-8B on PIQA, ARC-Easy, ARC-Challenge, HellaSwag, WinoGrande, and MMLU. The final column of Table 1 is the unweighted average over these six benchmarks. We compare MANTA with Llamba (Wang et al., 2024), LoLCaT (Zhang et al., 2025), Liger-GLA (Lan et al., 2025), GDN (Kuzina et al., 2026), LoLA (McDermott et al., 2025), and STILL (Meng et al., 2026) under their reported cache budgets.

With a 64-token SWA cache (8 sink and 56 recent tokens), MANTA obtains an average accuracy of 72.4. This is within 0.1 points of the original softmax model and 1.1 points above the best prior linearized method at the same cache size. Increasing the SWA cache to 128 tokens (8 sink and 120 recent tokens) raises the average to 73.2, which exceeds both the softmax model and the strongest prior linearized result by 0.7 points. At this setting, MANTA matches or improves upon softmax on five of the six benchmarks.

Inference efficiency. We measure end-to-end autoregressive decoding on one RTX 5090 with BF16 and vLLM 0.11.1. Each request contains a 512-token prompt and generates either 8K or 16K tokens. We vary concurrency from 16 to 128 and compare two MANTA configurations with Liger-GLA and the original Llama-3.1-8B.

Table 1: Linearization results on Llama-3.1-8B compared with prior work. MANTA results are reported as mean and standard deviation over two seeds.

Model	Cache size	PIQA	ARC-e	ARC-c	Hella.	Wino.	MMLU	Avg.
Llama-3.1-8B	∞	79.1	82.2	54.8	79.3	74.6	65.3	72.5
Linearized								
Llamba	0	80.9	82.5	54.6	77.6	73.3	60.0	71.5
LoLCaT	64	81.0	82.4	54.4	79.1	69.7	54.9	70.2
Liger-GLA	64	80.5	81.8	55.6	75.4	69.5	46.9	68.3
GDN	64	79.1	82.0	54.9	78.8	74.1	59.2	71.3
MANTA	64	80.3±0.19	82.9±0.28	58.0±0.77	80.6±0.01	74.4±0.00	58.3±0.18	72.4±0.03
Extended cache budget								
LoLa	128	81.6	82.5	55.4	79.8	73.6	57.6	71.8
STILL	NA	81.3	83.0	56.7	79.0	73.4	61.3	72.5
GDN	128	79.2	82.2	54.9	79.3	74.1	63.2	72.1
MANTA	128	80.2±0.25	82.6±0.51	57.8±0.18	81.3±0.09	75.1±0.16	62.4±0.21	73.2±0.10



(a) 8K output. (b) 16K output. (c) Request capacity.

Figure 4: Aggregate decoding throughput across concurrency levels at 8K and 16K output lengths, and maximum request capacity at 16.9K tokens, measured on one RTX 5090.

Figure 4 shows that the advantage of fixed-state decoding grows with both concurrency and generation length. At concurrency 128, the $R = 4, r = 16$ configuration reaches 5.33K and 5.69K output tokens/s for 8K and 16K generations, respectively. These throughputs are $11.5\times$ and $27.3\times$ those of Llama-3.1-8B, and approximately $1.5\times$ those of Liger-GLA. Both MANTA configurations support 685 concurrent 16.9K-token requests, compared with 509 for Liger-GLA and 6.3 for Llama-3.1-8B.

5.2 ABLATION STUDIES

Component ablations. Table 2 starts from a four-center hard-routing baseline and adds each component cumulatively. A2 replaces hard assignments with soft routing, A3 introduces the query-aware metric, A4 adds jointly normalized SWA, and A5 applies the learned low-rank projection.

Table 2: Cumulative component ablation of MANTA on Llama-3.1-8B.

Variant	Components				Accuracy (%)						
	Soft routing	Query-aware	Joint SWA	Low-rank	PIQA	ARC-e	ARC-c	Hella.	Wino.	MMLU	Avg.
A1	–	–	–	–	49.5	25.1	22.7	25.0	49.6	23.0	32.5
A2	✓	–	–	–	72.6	66.1	38.7	62.6	54.7	23.9	53.1
A3	✓	✓	–	–	76.5	73.0	45.9	68.7	63.1	24.9	58.7
A4	✓	✓	✓	–	80.3	82.3	58.8	81.1	73.8	58.8	72.5
A5	✓	✓	✓	✓	80.3	82.9	58.0	80.6	74.4	58.3	72.4

Soft routing provides the largest initial improvement, raising the average accuracy from 32.5 to 53.1. Query-aware routing contributes another 5.6 points. Jointly normalized SWA then raises the average from 58.7 to 72.5, showing the importance of retaining exact attention for the selected tokens and combining the two branches with a shared normalization. Finally, the low-rank model retains an average of 72.4, within 0.1 points of its uncompressed counterpart. This result shows that the memory reduction in Section 4.3 incurs negligible loss in downstream accuracy.

Number of Taylor centers. To isolate the effect of multiple expansion points, we vary R in dense MANTA without SWA or low-rank projection. Figure 5 shows that the five-task average increases monotonically from 49.1 at $R = 1$ to 58.7, 64.6, and 66.0 at $R = 2$, 4, and 8, respectively. Every benchmark improves between one and four centers. The smaller gain from four to eight centers indicates diminishing returns, supporting $R = 4$ as the default balance between accuracy and recurrent-state size.

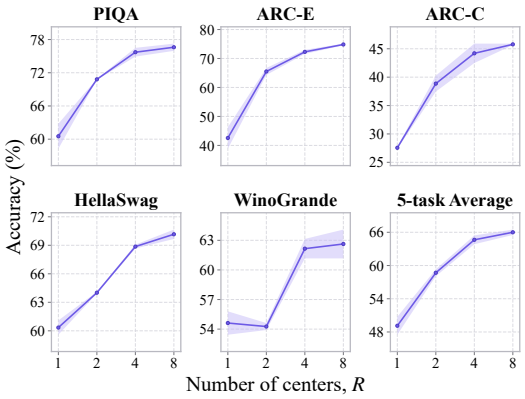


Figure 5: Accuracy as the number of Taylor centers R varies. Shading shows the standard deviation over two training seeds.

5.3 LONG-CONTEXT MODELING EVALUATION

To assess long-context modeling ability, we evaluate MANTA and Liger both before and after SFT on long-context data across seven LongBench tasks: Qasper (QQA), MultiFieldQA-English (MFQ), HotpotQA (HQA), 2WikiMQA (2WM), MuSiQue (Mus), TREC (TRC), and TriviaQA (TQA). For long-context SFT, both methods use the same 512-example schedule from LongAlpaca-12k (80% long and 20% short examples) for 256 optimization steps, with a maximum training length of 16,384 tokens. MANTA uses $R = 4$, $r = 16$, and a 64-token SWA cache. Figure 6 shows that SFT on long-context data raises MANTA's average from 23.51 to 27.86 and Liger's from 9.27 to 20.47. MANTA outperforms Liger on every task both before and after adaptation, retaining a 7.40-point average lead after SFT. These results demonstrate stronger long-context modeling ability than Liger on the evaluated tasks.

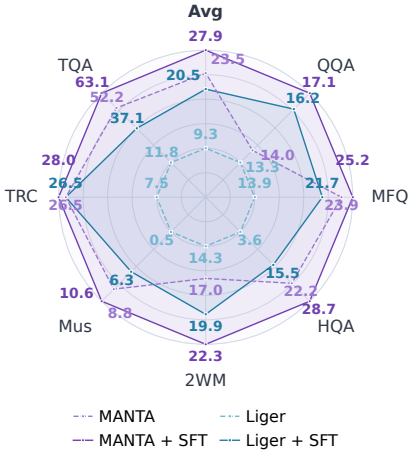


Figure 6: LongBench scores with independently scaled axes; Avg denotes the seven-task mean.

6 CONCLUSION

We presented MANTA, a method for converting pretrained softmax Transformers into fixed-state recurrent models through multi-center Taylor attention. Our analysis identifies the mismatch between a single expansion center and heterogeneous pretrained key distributions as a source of approximation error. Query-aware soft routing addresses this mismatch by forming local Taylor readouts, while learned low-rank state compression controls their memory cost. Joint normalization combines the compressed history with exact attention over sink and recent tokens. Together, these components preserve linear total computation and context-independent recurrent memory. On Llama-3.1-8B, MANTA achieves a six-benchmark average of 73.2 with a 128-token exact cache and up to 27.3× decoding throughput relative to softmax in our measured settings. It also outperforms Liger on all seven evaluated LongBench tasks before and after long-context SFT. These findings support local, query-aware Taylor approximation as a practical route to accurate and efficient linearization of pretrained softmax Transformers.

AI USE STATEMENT

We used generative AI tools to assist with research ideation and execution. This included refining methodological ideas and mathematical claims, planning experiments, implementing and reviewing research code, and interpreting results. Generative AI also assisted with formulating and checking proofs, and with drafting and polishing parts of the manuscript. We reviewed AI-assisted mathematical arguments against the proofs in Appendix A, checked numerical claims against recorded experimental results, and revised the final text. The authors take full responsibility for all claims, proofs, code, results, and text in this work.

REPRODUCIBILITY STATEMENT

The main text specifies the model and evaluation configurations. Complete proofs of the theoretical claims are provided in Appendix A, and the one-step decoding procedure is given in Appendix B. Appendix C documents the datasets and preprocessing, training settings, inference and scoring protocols, hardware, and additional results. The accompanying anonymized source-code supplement contains the core implementation and scripts for conversion, training, and short-context evaluation.

REFERENCES

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.

David Arthur and Sergei Vassilvitskii. k-means++: The advantages of careful seeding. In *Soda*, volume 7, pp. 1027–1035, 2007.

Yushi Bai, Xin Lv, Jiajie Zhang, Hongchang Lyu, Jiankai Tang, Zhidian Huang, Zhengxiao Du, Xiao Liu, Aohan Zeng, Lei Hou, et al. 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)*, pp. 3119–3137, 2024.

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

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.

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

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

Yukang Chen, Shengju Qian, Haotian Tang, Xin Lai, Zhijian Liu, Song Han, and Jiaya Jia. Longlora: Efficient fine-tuning of long-context large language models. In *International Conference on Learning Representations*, volume 2024, pp. 8220–8238, 2024b.

Krzysztof Choromanski, Valerii Likhosherstov, David Dohan, Xingyou Song, Andreea Gane, Tamas Sarlos, Peter Hawkins, Jared Davis, Afroz Mohiuddin, Lukasz Kaiser, et al. Rethinking attention with performers. *arXiv preprint arXiv:2009.14794*, 2020.

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.

Tri Dao, Dan Fu, Stefano Ermon, Atri Rudra, and Christopher Ré. Flashattention: Fast and memory-efficient exact attention with io-awareness. *Advances in neural information processing systems*, 35: 16344–16359, 2022.

Pradeep Dasigi, Kyle Lo, Iz Beltagy, Arman Cohan, Noah A Smith, and Matt Gardner. A dataset of information-seeking questions and answers anchored in research papers. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pp. 4599–4610, 2021.

Alexander Richard Fabbri, Irene Li, Tianwei She, Suyi Li, and Dragomir Radev. Multi-news: A large-scale multi-document summarization dataset and abstractive hierarchical model. In *Proceedings of the 57th annual meeting of the association for computational linguistics*, pp. 1074–1084, 2019.

Bogdan Gliwa, Iwona Mochol, Maciej Biesek, and Aleksander Wawer. Samsum corpus: A human-annotated dialogue dataset for abstractive summarization. In *Proceedings of the 2nd Workshop on New Frontiers in Summarization*, pp. 70–79, 2019.

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

Daya Guo, Canwen Xu, Nan Duan, Jian Yin, and Julian McAuley. Longcoder: A long-range pre-trained language model for code completion. In *International Conference on Machine Learning*, pp. 12098–12107. PMLR, 2023.

Wei He, Kai Liu, Jing Liu, Yajuan Lyu, Shiqi Zhao, Xinyan Xiao, Yuan Liu, Yizhong Wang, Hua Wu, Qiaoqiao She, et al. Dureader: a chinese machine reading comprehension dataset from real-world applications. In *Proceedings of the workshop on machine reading for question answering*, pp. 37–46, 2018.

Dan Hendrycks, Collin Burns, Steven Basart, Andy Zou, Mantas Mazeika, Dawn Song, and Jacob Steinhardt. Measuring massive multitask language understanding. *arXiv preprint arXiv:2009.03300*, 2020.

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

Luyang Huang, Shuyang Cao, Nikolaus Parulian, Heng Ji, and Lu Wang. Efficient attentions for long document summarization. In *Proceedings of the 2021 conference of the north American chapter of the association for computational linguistics: Human language technologies*, pp. 1419–1436, 2021.

Mandar Joshi, Eunsol Choi, Daniel S Weld, and Luke Zettlemoyer. Triviaqa: A large scale distantly supervised challenge dataset for reading comprehension. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 1601–1611, 2017.

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.

Tobias Katsch. Gateloop: Fully data-controlled linear recurrence for sequence modeling. *arXiv preprint arXiv:2311.01927*, 2023.

Tomáš Kočiskỳ, Jonathan Schwarz, Phil Blunsom, Chris Dyer, Karl Moritz Hermann, Gábor Melis, and Edward Grefenstette. The narrativeqa reading comprehension challenge. *Transactions of the Association for Computational Linguistics*, 6:317–328, 2018.

Anna Kuzina, Paul N Whatmough, and Babak Ehteshami Bejnordi. The key to going linear: Analysis-driven transformer linearization. *arXiv preprint arXiv:2607.07706*, 2026.

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.

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.

Xin Li and Dan Roth. Learning question classifiers. In *Coling 2002: The 19th international conference on computational linguistics*, 2002.

Tianyang Liu, Canwen Xu, and Julian McAuley. RepoBench: Benchmarking repository-level code auto-completion systems. *arXiv preprint arXiv:2306.03091*, 2023. URL `https://arxiv.org/abs/2306.03091`.

Luke McDermott, Robert W Heath Jr, and Rahul Parhi. Lola: Low-rank linear attention with sparse caching. *arXiv preprint arXiv:2505.23666*, 2025.

Weikang Meng, Liangyu Huo, Yadan Luo, Jiawen Guan, Jingyi Zhang, Yingjian Li, and Zheng Zhang. Still: Selecting tokens for intra-layer hybrid attention to linearize llms. *arXiv preprint arXiv:2602.02180*, 2026.

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.

Bo Peng, Daniel Goldstein, Quentin Anthony, Alon Albalak, Eric Alcaide, Stella Biderman, Eugene Cheah, Xingjian Du, Teddy Ferdinan, Haowen Hou, et al. Eagle and finch: Rwkv with matrix-valued states and dynamic recurrence. *arXiv preprint arXiv:2404.05892*, 2024.

Zhen Qin, Dong Li, Weigao Sun, Weixuan Sun, Xuyang Shen, Xiaodong Han, Yunshen Wei, Baohong Lv, Xiao Luo, Yu Qiao, et al. Transnormerllm: A faster and better large language model with improved transnormer. *arXiv preprint arXiv:2307.14995*, 2023.

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*, 2024.

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.

Yutao Sun, Li Dong, Shaohan Huang, Shuming Ma, Yuqing Xia, Jilong Xue, Jianyong Wang, and Furu Wei. Retentive network: A successor to transformer for large language models. *arXiv preprint arXiv:2307.08621*, 2023.

Rohan Taori, Ishaan Gulrajani, Tianyi Zhang, Yann Dubois, Xuechen Li, Carlos Guestrin, Percy Liang, and Tatsunori B Hashimoto. Alpaca: A strong, replicable instruction-following model. *Stanford Center for Research on Foundation Models. https://crfm. stanford. edu/2023/03/13/alpaca. html*, 3(6):7, 2023.

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, 2025. *URL https://arxiv. org/abs/2510.26692*, 2025.

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

Chien Van Nguyen, Huy Huu Nguyen, Ruiyi Zhang, Hanieh Deilamsalehy, Puneet Mathur, Viet Dac Lai, Haoliang Wang, Jayakumar Subramanian, Ryan A Rossi, Trung Bui, et al. Lizard: An efficient linearization framework for large language models. In *Proceedings of the 64th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 34935–34948, 2026.

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.

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.

Han Wu, Mingjie Zhan, Haochen Tan, Zhaohui Hou, Ding Liang, and Linqi Song. Vcsum: A versatile chinese meeting summarization dataset. In *Findings of the Association for Computational Linguistics: ACL 2023*, pp. 6065–6079, 2023.

Guangxuan Xiao, Yuandong Tian, Beidi Chen, Song Han, and Mike Lewis. Efficient streaming language models with attention sinks. In *International Conference on Learning Representations*, volume 2024, pp. 21875–21895, 2024.

Songlin Yang, Bailin Wang, Yikang Shen, Rameswar Panda, and Yoon Kim. Gated linear attention transformers with hardware-efficient training. In *Proceedings of the 41st International Conference on Machine Learning*, 2024.

Songlin Yang, Jan Kautz, and Ali Hatamizadeh. Gated delta networks: Improving mamba2 with delta rule. In *International Conference on Learning Representations*, volume 2025, pp. 29687–29707, 2025.

Zhilin Yang, Peng Qi, Saizheng Zhang, Yoshua Bengio, William Cohen, Ruslan Salakhutdinov, and Christopher D Manning. Hotpotqa: A dataset for diverse, explainable multi-hop question answering. In *Proceedings of the 2018 conference on empirical methods in natural language processing*, pp. 2369–2380, 2018.

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, Kush Bhatia, Hermann Kumbong, and Christopher Ré. The hedgehog & the porcupine: Expressive linear attentions with softmax mimicry. In *International Conference on Learning Representations*, volume 2024, pp. 53551–53580, 2024a.

Michael Zhang, Simran Arora, Rahul Chalamala, Benjamin Spector, Alan Wu, Krithik Ramesh, Aaryan Singhal, and Christopher Ré. Lolcats: On low-rank linearizing of large language models. In *International Conference on Learning Representations*, volume 2025, pp. 46200–46253, 2025.

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, 2024b.

Ming Zhong, Da Yin, Tao Yu, Ahmad Zaidi, Mutethia Mutuma, Rahul Jha, Ahmed Hassan, Asli Celikyilmaz, Yang Liu, Xipeng Qiu, et al. Qmsum: A new benchmark for query-based multi-domain meeting summarization. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pp. 5905–5921, 2021.

A PROOFS

A.1 PROOF OF LEMMA 1

Proof. Set $\boldsymbol{\delta} = \mathbf{z} - \bar{z}\mathbf{1}$. Then $\mathbf{1}^\top \boldsymbol{\delta} = 0$, and shift invariance of softmax gives $\mathbf{p}(\mathbf{z}) = \mathbf{p}(\boldsymbol{\delta})$. At the origin, $p_i(\mathbf{0}) = 1/m$ and

$$\frac{\partial p_i}{\partial z_j}(\mathbf{0}) = \frac{1}{m}\left(\mathbb{1}\{i = j\} - \frac{1}{m}\right). \tag{14}$$

Consequently, the linear term along $\boldsymbol{\delta}$ is

$$\nabla p_i(\mathbf{0})^\top \boldsymbol{\delta} = \frac{\delta_i}{m} - \frac{1}{m^2}\sum_{j=1}^{m} \delta_j = \frac{z_i - \bar{z}}{m}. \tag{15}$$

Taylor's theorem with integral remainder yields

$$p_i(\boldsymbol{\delta}) = \frac{1}{m} + \frac{\delta_i}{m} + \underbrace{\int_0^1 (1-s)\, \boldsymbol{\delta}^\top \nabla^2 p_i(s\boldsymbol{\delta})\boldsymbol{\delta}\, \mathrm{d}s}_{r_i(\mathbf{z})}. \tag{16}$$

For fixed m and any fixed radius $\rho > 0$, smoothness of softmax implies that its Hessians have a finite maximum operator norm $C_{m,\rho}$ on $\{\mathbf{w} : \|\mathbf{w}\|_2 \le \rho\}$, uniformly over i. Thus, whenever $\|\boldsymbol{\delta}\|_2 \le \rho$,

$$\|\mathbf{r}(\mathbf{z})\|_\infty \le \tfrac{1}{2} C_{m,\rho} \|\boldsymbol{\delta}\|_2^2, \tag{17}$$

which proves the stated local remainder bound. The bound depends on the centered scores, and is unaffected by the common shift $\bar{z}$. □

A.2 PROOF OF PROPOSITION 1

Proof. For fixed centers, the objective separates across tokens. For one token i, write $D_{i,j} = D_{\mathbf{M}}^2(\mathbf{x}_i, \mathbf{c}_j)$ and define

$$Z_i = \sum_{a=1}^{R} \exp(-D_{i,a}/\tau), \qquad \pi_{i,j} = \frac{\exp(-D_{i,j}/\tau)}{Z_i}. \tag{18}$$

Since $\tau > 0$ and the distances are finite, $\pi_{i,j} > 0$ and $\sum_j \pi_{i,j} = 1$. Using $0 \log 0 = 0$, for every $\mathbf{g}_i \in \Delta^{R-1}$ we have

$$\begin{aligned} \sum_j g_{i,j} D_{i,j} + \tau \sum_j g_{i,j} \log g_{i,j} &= \tau \sum_j g_{i,j} \log \frac{g_{i,j}}{\pi_{i,j}} - \tau \log Z_i \\ &= \tau\, \mathrm{KL}(\mathbf{g}_i \| \boldsymbol{\pi}_i) - \tau \log Z_i. \end{aligned} \tag{19}$$

The KL divergence is nonnegative and vanishes exactly when $\mathbf{g}_i = \boldsymbol{\pi}_i$. This proves the unique assignment minimizer, including positivity and normalization.

For fixed assignments, consider a center with mass $m_j = \sum_i g_{i,j} > 0$. Its center-dependent objective is

$$f_j(\mathbf{c}) = \frac{1}{d}\sum_i g_{i,j} (\mathbf{x}_i - \mathbf{c})^\top \mathbf{M} (\mathbf{x}_i - \mathbf{c}). \tag{20}$$

Its gradient and Hessian are

$$\nabla f_j(\mathbf{c}) = \frac{2}{d}\mathbf{M}\left(m_j \mathbf{c} - \sum_i g_{i,j}\mathbf{x}_i\right), \qquad \nabla^2 f_j(\mathbf{c}) = \frac{2m_j}{d}\mathbf{M} \succ 0. \tag{21}$$

Hence the unique minimizer is $\mathbf{c}_j^\star = m_j^{-1} \sum_i g_{i,j}\mathbf{x}_i$. If $m_j = 0$, every assignment to that center is zero and the objective is independent of its location, explaining the nonempty-center qualification. These are minimizers of the two conditional subproblems; the result does not assert that alternating their updates finds a global joint minimum. □

A.3 Proof of Proposition 2

Proof. Let $\mathbf{Q} = \mathbf{P}\mathbf{P}^\top$. The constraint $\mathbf{P}^\top\mathbf{P} = \mathbf{I}_r$ implies that $\mathbf{Q}$ is an orthogonal projector, so $\mathbf{Q}^\top = \mathbf{Q}$ and $\mathbf{Q}^2 = \mathbf{Q}$. Therefore,

$$\sum_{\mathbf{A}\in\mathcal{A}_h} \|\mathbf{A} - \mathbf{Q}\mathbf{A}\|_F^2 = \sum_{\mathbf{A}\in\mathcal{A}_h} \operatorname{tr}\big(\mathbf{A}^\top(\mathbf{I}-\mathbf{Q})\mathbf{A}\big) = \operatorname{tr}(\mathbf{\Sigma}_h) - \operatorname{tr}(\mathbf{P}^\top\mathbf{\Sigma}_h\mathbf{P}). \tag{22}$$

To maximize the second trace, write

$$\mathbf{\Sigma}_h = \mathbf{U}\operatorname{diag}(\lambda_1,\dots,\lambda_d)\mathbf{U}^\top, \qquad \lambda_1 \geq \cdots \geq \lambda_d \geq 0, \tag{23}$$

and set $\mathbf{B} = \mathbf{U}^\top\mathbf{P}$. Its columns are orthonormal. The squared row norms $\alpha_k = \sum_{a=1}^r B_{k,a}^2$ consequently satisfy $0 \leq \alpha_k \leq 1$ and $\sum_k \alpha_k = r$. Thus

$$\operatorname{tr}(\mathbf{P}^\top\mathbf{\Sigma}_h\mathbf{P}) = \sum_{k=1}^d \lambda_k\alpha_k \leq \sum_{k=1}^r \lambda_k. \tag{24}$$

For $r < d$, the inequality follows explicitly from

$$\sum_{k=1}^r \lambda_k - \sum_{k=1}^d \lambda_k\alpha_k = \sum_{k\leq r}(\lambda_k - \lambda_r)(1-\alpha_k) + \sum_{k>r}(\lambda_r - \lambda_k)\alpha_k \geq 0. \tag{25}$$

For $r = d$, equality is immediate. Taking the columns of $\mathbf{P}$ to span r leading eigenvectors attains equality. The minimum reconstruction error is therefore $\sum_{k=r+1}^d \lambda_k$. Any orthonormal basis of an optimal subspace gives the same projector; a tie at the r-th eigenvalue can additionally make the optimal subspace nonunique. □

B One-step decoding

Algorithm 1 describes one decoding step of MANTA. Given the current query, key, and value, it updates the cached states and computes the attention output, returning the updated states for the next step.

C Datasets, experimental settings, and additional results

C.1 Datasets

Instruction-tuning data. The two-stage conversion uses Alpaca-cleaned, a curated version of the instruction–response data released with Stanford Alpaca (Taori et al., 2023). We use the `yahma/alpaca-cleaned` release,[1] reserve the first 200 examples for validation, and pack the remaining formatted examples into 1,024-token blocks. Prompt tokens are masked from the loss, which supervises response tokens. Router calibration uses formatted training examples. For long-context adaptation, we use LongAlpaca-12k (Chen et al., 2024b), which contains long-context instruction examples based on books and research papers as well as short instruction examples. We retain complete examples with at most 512 response tokens, including EOS, and at most 15,872 prompt tokens. Long prompts contain at least 8,192 tokens; short prompts contain at most 2,048 tokens. Intermediate-length examples are excluded. Document-level grouping and exact-overlap screening against the LongBench evaluation bundle produce 5,243 training examples and 570 development examples. This screening checks exact text overlap, not semantic equivalence.

Short-context benchmarks. PIQA tests physical commonsense by choosing between two solutions to an everyday goal (Bisk et al., 2020). ARC-Easy and ARC-Challenge contain grade-school science questions, with the Challenge split selected to defeat simple retrieval and word co-occurrence baselines (Clark et al., 2018). HellaSwag evaluates plausible event continuations (Zellers et al., 2019); WinoGrande evaluates commonsense reference resolution (Sakaguchi et al., 2021); and MMLU covers 57 subjects spanning factual knowledge and reasoning (Hendrycks et al., 2020).

[1] Dataset: `https://huggingface.co/datasets/yahma/alpaca-cleaned`; cleaning project: `https://github.com/gururise/AlpacaDataCleaned`.

Algorithm 1 One-step decoding with MANTA

Require: current $\mathbf{q}_t$, normalized pre-RoPE key $\mathbf{x}_t$, post-RoPE key $\mathbf{k}_t$, value $\mathbf{v}_t$
Require: fixed $\{\mathbf{c}_j\}_{j=1}^{R}$, $\mathbf{L}$, $\tau > 0$, $\mathbf{P} \in \mathbb{R}^{d\times r}$ with $\mathbf{P}^\top\mathbf{P} = \mathbf{I}_r$; sink count $S \geq 0$, window size $W \geq 1$
Require: previous states $\{(n_{t-1,j}, \mathbf{K}_{t-1,j}, \mathbf{V}_{t-1,j}, \mathbf{H}_{t-1,j})\}_{j=1}^{R}$; cache $\mathcal{C}_{t-1} = \{(i, \mathbf{k}_i, \mathbf{v}_i, \mathbf{g}_i) : i \in \mathcal{E}_{t-1}\}$
Initialize once: all states to zero, $\mathcal{E}_0 = \mathcal{C}_0 = \varnothing$; $\mathbf{H}_{0,j} \in \mathbb{R}^{r\times d_v}$
$g_{t,j} \leftarrow \mathrm{softmax}_j\big(-\|\mathbf{L}(\mathbf{x}_t - \mathbf{c}_j)\|_2^2/\tau\big)$, $j = 1, \ldots, R$
$\mathcal{E}_t \leftarrow \{1, \ldots, \min(S, t)\} \cup \{\max(1, t - W + 1), \ldots, t\}$
$\mathcal{C} \leftarrow \mathcal{C}_{t-1} \cup \{(t, \mathbf{k}_t, \mathbf{v}_t, \mathbf{g}_t)\}$; $\mathcal{D}_t \leftarrow (\mathcal{E}_{t-1} \cup \{t\}) \setminus \mathcal{E}_t$
$(n, \mathbf{K}, \mathbf{V}, \mathbf{H})_{t,j} \leftarrow (n, \mathbf{K}, \mathbf{V}, \mathbf{H})_{t-1,j}$ for all j
for each evicted index $i \in \mathcal{D}_t$ **do**
    Read $(\mathbf{k}_i, \mathbf{v}_i, \mathbf{g}_i)$ from $\mathcal{C}$; $\mathbf{z}_i \leftarrow \mathbf{P}^\top\mathbf{k}_i$
    for $j = 1, \ldots, R$ **do**
        $(n, \mathbf{K}, \mathbf{V}, \mathbf{H})_{t,j} \leftarrow (n, \mathbf{K}, \mathbf{V}, \mathbf{H})_{t,j} + (g_{i,j}, g_{i,j}\mathbf{k}_i, g_{i,j}\mathbf{v}_i, g_{i,j}\mathbf{z}_i\mathbf{v}_i^\top)$
    end for
end for
$\mathcal{C}_t \leftarrow \{(i, \mathbf{k}_i, \mathbf{v}_i, \mathbf{g}_i) \in \mathcal{C} : i \in \mathcal{E}_t\}$
$\widetilde{\mathbf{q}}_t \leftarrow \mathbf{P}^\top\mathbf{q}_t$; $\mathcal{J}_t \leftarrow \{j : n_{t,j} > 0\}$
for $j \in \mathcal{J}_t$ **do**
    $\boldsymbol{\mu}_{t,j} \leftarrow \mathbf{K}_{t,j}/n_{t,j}$; $a_{t,j} \leftarrow \mathbf{q}_t^\top\boldsymbol{\mu}_{t,j}/\sqrt{d}$; $\ell_{t,j} \leftarrow a_{t,j} + \log n_{t,j}$
    $\mathbf{u}_{t,j} \leftarrow \big(\mathbf{H}_{t,j}^\top\widetilde{\mathbf{q}}_t/\sqrt{d} + (1 - a_{t,j})\mathbf{V}_{t,j}\big)/n_{t,j}$
end for
$b_{t,i} \leftarrow \mathbf{q}_t^\top\mathbf{k}_i/\sqrt{d}$ for each $i \in \mathcal{E}_t$
$m_t \leftarrow \max\big(\{\ell_{t,j} : j \in \mathcal{J}_t\} \cup \{b_{t,i} : i \in \mathcal{E}_t\}\big)$
$\mathbf{y}_t \leftarrow \dfrac{\sum_{j\in\mathcal{J}_t} e^{\ell_{t,j}-m_t}\mathbf{u}_{t,j} + \sum_{i\in\mathcal{E}_t} e^{b_{t,i}-m_t}\mathbf{v}_i}{\sum_{j\in\mathcal{J}_t} e^{\ell_{t,j}-m_t} + \sum_{i\in\mathcal{E}_t} e^{b_{t,i}-m_t}}$
return $\mathbf{y}_t$, $\mathcal{C}_t$, and $\{(n_{t,j}, \mathbf{K}_{t,j}, \mathbf{V}_{t,j}, \mathbf{H}_{t,j})\}_{j=1}^{R}$

Long-context benchmarks. We use the original LongBench v1 test sets (Bai et al., 2024): 21 datasets and 4,750 examples across six task categories. Single-document QA covers stories, scientific papers, and documents from multiple domains. Multi-document QA requires combining evidence across documents. Summarization covers government reports, meetings, and news. Few-shot tasks place demonstrations in the context for classification, question answering, or dialogue summarization. Synthetic tasks test passage counting and retrieval, while code tasks test completion with long in-file or cross-file context. Table 3 lists the exact evaluation sizes and generation limits. These are the LongBench subsets, not the sizes of the original source datasets. MultiFieldQA and the synthetic tasks are cited to LongBench itself; LSHT is attributed through LongBench's released task specification. This evaluation uses neither LongBench-E nor LongBench v2.

C.2 EXPERIMENTAL SETTINGS

Conversion and low-rank training. We convert Llama-3.1-8B using four Taylor centers per KV head and a rank-16 projected state. The 64-token exact cache contains eight sink tokens and 56 recent tokens; the 128-token variant retains eight sinks and 120 recent tokens. Each of the two training stages runs for 2,560 optimizer updates on one H100, with a microbatch of two 1,024-token sequences and four gradient-accumulation steps (effective batch size eight). This amounts to 20,971,520 input-token positions per stage. Both stages use BF16 model computation, gradient checkpointing, AdamW, a peak learning rate of 2×10^{-4}, and 3% warmup followed by linear decay. LoRA has rank 16 and scaling parameter 32 and is applied to the Q/K/V/O projections. Router parameters use one tenth of the main learning rate in stage one and are frozen in stage two. The projected basis remains trainable in stage two and is reorthonormalized by reduced QR after each update.

The router is initialized from 128 calibration sequences of length 1,024, with up to 32,768 sampled keys per KV head and 15 Lloyd iterations. The query-moment shrinkage coefficient is 0.05. The projection basis is initialized from dense recurrent-state trajectories collected over 8,192 tokens after stage one. Dense component ablations retain the same two-stage update budget but omit the projection where specified. The center-count ablation disables both SWA and projection.

Table 3: LongBench v1 evaluation inventory. EN/ZH denote English/Chinese; n is the number of evaluated examples and G is the maximum number of generated tokens. Metrics follow the official LongBench scorer, including its Chinese normalization and code edit-similarity functions.

Dataset and source	Metric	n	G
Single-document QA			
NarrativeQA (Kočiskỳ et al., 2018)	F1	200	128
Qasper (Dasigi et al., 2021)	F1	200	128
MultiFieldQA EN (Bai et al., 2024)	F1	150	64
MultiFieldQA ZH (Bai et al., 2024)	F1 (ZH)	200	64
Multi-document QA			
HotpotQA (Yang et al., 2018)	F1	200	32
2WikiMQA (Ho et al., 2020)	F1	200	32
MuSiQue (Trivedi et al., 2022)	F1	200	32
DuReader (He et al., 2018)	ROUGE-L (ZH)	200	128
Summarization			
GovReport (Huang et al., 2021)	ROUGE-L	200	512
QMSum (Zhong et al., 2021)	ROUGE-L	200	512
MultiNews (Fabbri et al., 2019)	ROUGE-L	200	512
VCSum (Wu et al., 2023)	ROUGE-L (ZH)	200	512
Few-shot learning			
TREC (Li & Roth, 2002)	Classif.	200	64
TriviaQA (Joshi et al., 2017)	F1	200	32
SAMSum (Gliwa et al., 2019)	ROUGE-L	200	128
LSHT (Bai et al., 2024)	Classif.	200	64
Synthetic tasks			
Passage Count (Bai et al., 2024)	Count	200	32
Passage Retrieval EN (Bai et al., 2024)	Retrieval	200	32
Passage Retrieval ZH (Bai et al., 2024)	Retrieval	200	32
Code completion			
LCC (Guo et al., 2023)	Code sim.	500	64
RepoBench-P (Liu et al., 2023)	Code sim.	500	64

Short-context evaluation. We score candidate answers with the language-model evaluation harness. PIQA, ARC-Easy, ARC-Challenge, HellaSwag, and WinoGrande use zero-shot evaluation; MMLU uses five in-context demonstrations. We report accuracy for PIQA, ARC-Easy, WinoGrande, and MMLU, and length-normalized accuracy for ARC-Challenge and HellaSwag, following the stored evaluation configuration. The six-benchmark average gives each benchmark equal weight. Prior-method entries in Table 1 are reported results from the cited work; the LongBench Liger results below instead use our local reproduction.

Long-context SFT. MANTA and Liger use the same fixed 512-example schedule with seed 42: 410 long and 102 short examples, corresponding to the requested 80/20 mix after integer rounding. We train for 256 updates with microbatch size one and gradient accumulation two, using a 16,384-token maximum sequence length. The objective is response-only cross entropy. AdamW uses learning rate 2×10^{-5}, weight decay 0.01, 16 warmup steps, linear decay, and gradient clipping at norm 1.0. The schedule contains 5,115,962 input tokens and 127,719 supervised response tokens. Evaluation on 32 held-out examples and checkpoint saving occur every 64 steps; the reported LongBench results use the final step-256 checkpoints. MANTA adapts its existing LoRA parameters, router, and state basis, with router learning rate 2×10^{-6}; Liger adapts its existing Q/K/V LoRA parameters (rank 8, scaling parameter 8).

LongBench inference and scoring. All reported LongBench runs use the same frozen test bundle, official base-model prompts without a chat wrapper, greedy decoding, and a maximum input length of 16,384 tokens. Overlong prompts use the official head/tail decode–re-encode truncation, followed

Table 4: Complete LongBench v1 results for all four evaluations (0–100; higher is better). "+SFT" denotes the matched 16K adaptation.

	Liger		MANTA	
Task	Before SFT	+SFT	Before SFT	+SFT
Single-document QA				
NarrativeQA	0.55	9.38	11.78	13.69
Qasper	13.30	16.24	13.97	17.11
MultiFieldQA EN	13.88	21.70	23.92	25.22
MultiFieldQA ZH	7.36	11.48	20.98	20.55
Multi-document QA				
HotpotQA	3.59	15.52	22.19	28.70
2WikiMQA	14.34	19.89	16.96	22.33
MuSiQue	0.50	6.30	8.83	10.59
DuReader	1.04	7.40	18.55	19.75
Summarization				
GovReport	5.32	11.74	11.65	16.94
QMSum	3.51	14.80	18.28	19.92
MultiNews	13.36	14.37	13.01	17.11
VCSum	0.86	1.71	1.78	3.83
Few-shot learning				
TREC	7.50	26.50	26.50	28.00
TriviaQA	11.78	37.12	52.18	63.08
SAMSum	6.09	14.15	24.59	28.63
LSHT	0.00	0.50	2.50	1.00
Synthetic tasks				
Passage Count	0.36	0.23	2.00	1.00
Passage Retrieval EN	1.14	4.25	5.50	5.50
Passage Retrieval ZH	0.68	1.76	4.00	3.50
Code completion				
LCC	41.28	46.44	49.69	53.79
RepoBench-P	27.20	43.08	46.34	50.14
21-task mean	8.27	15.45	18.82	21.45

by a strict final token-cap check. We use batch size one, 512-token prefill chunks, and a continuous recurrent cache across chunks. Model computation is BF16; Liger's validated recurrent implementation retains FP32 recurrent arithmetic. Evaluation runs use one RTX 5090 per worker. For Liger+SFT, 19 tasks use PyTorch 2.8.0a0 (NGC 25.06), while GovReport and DuReader use PyTorch 2.7.1+cu128 after migration to a separate RTX 5090 worker. Both environments pass the same numerical and generation checks; the checkpoint, inference code, and evaluation rules are identical.

Generation stops at EOS or the task-specific cap in Table 3. SAMSum also uses the official newline stop token and permits at least one generated token. We use the unmodified official task-to-metric mapping, take the maximum score over the supplied reference answers, and apply the first-line scoring rule to TREC, TriviaQA, SAMSum, and LSHT. All scores are on a 0–100 scale. Empty predictions remain in the evaluation.

Efficiency measurements. The throughput experiments use vLLM 0.11.1 on one RTX 5090 with BF16 model weights, a 512-token prompt, output lengths of 8,192 or 16,384 tokens, and concurrency levels 16, 32, 64, and 128. Throughput denotes aggregate output tokens per second across concurrent requests. The two MANTA deployments use $(R, r) = (2, 32)$ and $(4, 16)$ with a 64-token exact cache.

C.3 COMPLETE LONGBENCH RESULTS

Table 4 reports all 21 task scores for the models in Figure 6. All four evaluations cover the same 4,750 examples. We report the full-suite mean, computed from unrounded task scores and then rounded to two decimal places.

Long-context SFT increases MANTA's full-suite mean from 18.82 to 21.45, an improvement of 2.63 points. Sixteen tasks improve, four decline, and one is unchanged. The largest gain is on TriviaQA (+10.90 points), followed by HotpotQA (+6.51) and 2WikiMQA (+5.37). Across all 21 tasks, Liger improves from 8.27 to 15.45 after SFT, while MANTA+SFT reaches 21.45, a 5.99-point advantage computed from unrounded means. MANTA+SFT scores higher on all 21 tasks in these single-run evaluations.

Table 5: LongBench category means for all four complete evaluations. Each row averages its constituent task scores without weighting by sample count.

Category	Liger	Liger+SFT	MANTA	MANTA+SFT
Single-document QA	8.77	14.70	17.66	19.15
Multi-document QA	4.86	12.28	16.64	20.34
Summarization	5.76	10.65	11.18	14.45
Few-shot learning	6.34	19.57	26.44	30.18
Synthetic tasks	0.73	2.08	3.83	3.33
Code completion	34.24	44.76	48.02	51.96

Table 5 shows that MANTA's improvements extend to QA, summarization, few-shot learning, and code completion, while the synthetic-task mean declines from 3.83 to 3.33. In particular, English passage retrieval remains at 5.50, and passage counting and Chinese retrieval decline.

C.4 NUMERICAL RESULTS FOR THE NUMBER OF TAYLOR CENTERS

Table 6 provides the values underlying Figure 5. We report the mean and population standard deviation across two independent training seeds. The average is first computed over the five benchmarks within each seed, then summarized across seeds; its standard deviation is not the average of the taskwise deviations.

Table 6: Dense center-count ablation without SWA or low-rank projection: mean $\pm$ population standard deviation across training seeds 42 and 43, in percentage points.

R	PIQA	ARC-e	ARC-c	HellaSwag	WinoGrande	Avg.
1	60.53 ± 2.26	42.57 ± 3.72	27.56 ± 0.17	60.34 ± 0.75	54.62 ± 1.18	49.12 ± 1.62
2	70.81 ± 0.08	65.51 ± 1.20	38.87 ± 1.41	64.01 ± 0.09	54.26 ± 0.36	58.69 ± 0.63
4	75.71 ± 0.79	72.29 ± 0.69	44.20 ± 1.71	68.87 ± 0.20	62.15 ± 0.99	64.64 ± 0.79
8	76.61 ± 0.65	74.83 ± 0.29	45.78 ± 0.13	70.16 ± 0.49	62.63 ± 1.46	66.00 ± 0.49

Increasing R from one to four improves every task mean and raises the five-task average by 15.52 points. Moving from four to eight centers adds 1.36 points while doubling the number of dense cluster states. This supports the choice of four centers within the evaluated configuration.