MANTA: A MULTI-LEVEL AGGREGATION TRANSFORMER

Anonymous authors
Paper under double-blind review

ABSTRACT

Large language models are increasingly deployed on tasks that require access to large amounts of context, including document collections, code repositories, and long-running agent trajectories. Processing such extensive information with Transformers is computationally and memory intensive: during autoregressive generation, the key-value (KV) cache grows linearly with sequence length, increasing accelerator memory demands and limiting throughput. The challenge is therefore to reduce the memory required to represent the full sequence history without compromising the information available for future predictions. We introduce MANTA, a Multi-level Aggregation Transformer that progressively aggregates distant context while retaining recent tokens at full resolution. MANTA maintains a window size at each aggregation level, resulting in logarithmic KV-cache growth while retaining information spanning the full sequence history. We evaluate MANTA by adapting the Llama 3.2 1B base architecture and compare it with a dense-attention baseline. Despite its substantially smaller KV-cache, MANTA achieves competitive language-modeling performance and retains strong performance on long-context tasks. In a typical configuration, MANTA reduces the KV-cache size by 97.5% at a context length of 8K tokens. We further show that, with a natural modification, aggregation can replace tokenization, and that MANTA can be readily adapted to perform nearly lossless token compression. Overall, multi-level aggregation offers substantial architectural flexibility while directly addressing the memory capacity and bandwidth bottlenecks that increasingly constrain large-scale LLM inference.

1 INTRODUCTION

The growing ability of large language models (LLMs) to process extensive context has driven rapid increases in supported context lengths, from 2,048 tokens in GPT-3 (Brown et al., 2020) and Llama 2 (Touvron et al., 2023) to one million tokens in recent models such as Kimi K3 (Kimi Team, 2026) and DeepSeek-V4.1-Flash (DeepSeek-AI, 2026b). However, decoding over such long contexts remains computationally demanding, as each new token requires accessing both model parameters and representations of the preceding sequence.

For Transformer models, these representations are primarily stored in the key-value (KV) cache. Standard attention retains a key and value for every preceding token at each attention layer, causing the KV cache to grow linearly with sequence length. This cost is particularly important during serving: while model weights are shared across multiple requests, the KV cache is maintained for each sequence, so longer contexts increase per-request memory usage and reduce serving concurrency. During autoregressive decoding, attention layers repeatedly access the cached representations, making memory bandwidth a major bottleneck at long context lengths. Reducing KV-cache size is thus increasingly important for both memory capacity and bandwidth efficiency in LLM inference.

A growing body of work reduces KV-cache memory and bandwidth costs through techniques such as sharing keys and values across query heads (Shazeer, 2019; Ainslie et al., 2023), quantizing cached representations (Liu et al., 2024), and reducing their dimensionality (DeepSeek-AI et al., 2024). Recent architectures further combine these ideas with cross-layer KV sharing, window attention, or state-space models (Xu et al., 2026; Agarwal et al., 2025). We explore a complementary direction: rather than only reducing the cost of representing each past token, can a model learn to progressively aggregate the history itself?

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.

We introduce MANTA, a Multi-level Aggregation Transformer that represents sequence history at progressively coarser resolutions. MANTA retains recent tokens at full resolution, while groups of older representations are recursively aggregated into learned higher-level representations. During generation, the model attends to a bounded window at each level, retaining access to the full history without storing every past token independently. With aggregation factor c and per-level window size w, the resulting KV cache grows as $\mathcal{O}(w \log_c n)$ with sequence length n, compared with $\mathcal{O}(n)$ for standard attention. By retaining and accessing fewer historical representations, MANTA reduces both KV-cache memory and bandwidth requirements.

Our contributions are as follows:

- We introduce MANTA, a Multi-level Aggregation Transformer that recursively aggregates older representations while retaining recent context at full resolution, resulting in logarithmic rather than linear KV-cache growth with sequence length.
- We evaluate MANTA at the 1B-parameter scale and systematically study its aggregation hyperparameters. MANTA achieves competitive language-modeling performance while substantially reducing KV cache requirements, with a 97.5% reduction at 8K context in a typical configuration. Our implementation achieves up to a $10\times$ increase in generation speed at this context length. Furthermore, we show how MANTA can benefit from hybridization with dense attention.
- We show that MANTA retains strong long-context retrieval performance despite its small KV cache. On needle-in-a-haystack tasks at 8K context, the model successfully retrieves information propagated through multiple levels of aggregation, while its attention patterns allow us to trace retrieval through the representation hierarchy.
- We demonstrate that multi-level aggregation extends beyond KV-cache reduction. MANTA can be adapted into a language autoencoder for nearly lossless token compression and, with a natural architectural modification, can operate directly on bytes, replacing conventional tokenization while performing most computation on aggregated representations.

2 Related Work

Reducing the cost of attention over long sequences in Transformer-based LLMs has motivated a broad range of architectural approaches. Window attention restricts each token to a local neighborhood, while sparse-attention methods retain access to selected positions outside this window. Early approaches typically relied on fixed sparsity patterns (Child et al., 2019; Beltagy et al., 2020; Zaheer et al., 2020), whereas more recent methods dynamically select tokens or blocks based on the input (Li et al., 2024; Yuan et al., 2025; Lai et al., 2026). Linear-attention methods instead represent sequence history through a fixed-size recurrent state, avoiding explicit attention over all preceding tokens (Katharopoulos et al., 2020; Choromanski et al., 2021). These include delta-rule methods such as DeltaNet (Schlag et al., 2021) and Kimi Delta Attention (Kimi Team, 2025). Related approaches replace attention-based access to the full history more generally with recurrent states, including state-space models such as Mamba (Gu & Dao, 2024). In contrast, our approach retains standard softmax attention and changes how the history presented to it is represented: increasingly distant context is progressively aggregated into fewer representations.

Rather than modifying the attention operation or selecting a sparse subset of the context, another line of work reduces the amount of past information that must be represented explicitly through learned memory and compression. Transformer-XL (Dai et al., 2019) extends the context beyond the current segment by caching hidden states from preceding segments. Compressive Transformer (Rae et al., 2020) additionally introduces a compressed memory: as activations leave the high-resolution memory, they are compressed and retained at lower resolution rather than being discarded. Recurrent Memory Transformers (Bulatov et al., 2022) take a different approach, propagating a fixed set of learned memory tokens between consecutive segments and thereby compressing information from the preceding context into a bounded recurrent state. Related architectures such as MELODI (Chen et al., 2025) further combine local computation with compressed recurrent memory.

Learned context compression takes this idea further by training the model to represent spans of input using a small number of dedicated representations. Gist Tokens (Mu et al., 2023), AutoCompressors (Chevalier et al., 2023), and the In-context Autoencoder (Ge et al., 2024) learn special tokens that

summarize preceding context and can be used in place of the original input. Rather than passing only the resulting hidden representations, subsequent work shows that retaining the key-value states associated with these memory tokens provides a particularly effective interface for compressed context (Li et al., 2025b). Activation Beacon (Zhang et al., 2025) interleaves such memory tokens with the input and retains their KV states after the tokens they summarize are removed from the active context. UniGist (Deng et al., 2025) further develops this approach by removing the need for chunk-wise training. These methods demonstrate that learned representations can effectively substitute for portions of the original context; MANTA builds on this principle by recursively aggregating such representations into multiple levels of temporal resolution.

Learned compression can also be applied hierarchically. One class of architectures introduces a hierarchy across network depth: models such as the Hourglass Transformer (Nawrot et al., 2022) progressively reduce the sequence length in intermediate layers before expanding it again. H-Net (Hwang et al., 2026) extends this idea with learned, nested segmentation, allowing the model to dynamically construct increasingly coarse representations and operate directly on raw bytes. Such architectures reduce the number of representations processed as information passes through the network. Other methods organize the context itself hierarchically: BP-Transformer (Ye et al., 2019) uses binary partitioning to represent nearby context at fine resolution and distant context through coarser spans, while AdmTree (Li et al., 2025a) constructs hierarchical memory tokens through self-attention and averaging. MANTA likewise maintains a hierarchy over sequence history: representations become progressively coarser as they recede into the past, while multiple resolutions remain simultaneously available to attention. Its unified hierarchical causal mask enables attention within and across levels, with a sliding window at each resolution.

3 MANTA: THE MULTI-LEVEL AGGREGATION TRANSFORMER

MANTA retains the standard Transformer attention mechanism but augments the sequence with a hierarchy of memory tokens and modifies the causal attention mask to aggregate and retrieve information across multiple temporal resolutions.

Standard Transformer Attention. We first introduce notation used throughout this section. For a sequence of hidden representations $\boldsymbol{x} = (\boldsymbol{x}_0, \ldots, \boldsymbol{x}_{S-1})$, a Transformer attention layer computes queries, keys, and values as

$$\boldsymbol{q}_i = \boldsymbol{W}_Q \boldsymbol{x}_i, \qquad \boldsymbol{k}_i = \boldsymbol{W}_K \boldsymbol{x}_i, \qquad \boldsymbol{v}_i = \boldsymbol{W}_V \boldsymbol{x}_i \qquad \text{for } i = 0, \ldots S-1.$$

For attention all possible inner products $\boldsymbol{q}_i^\top \boldsymbol{k}_j$ are considered, leading to quadratic complexity. For masked attention with attention mask $\boldsymbol{M} \in \{0,1\}^{S \times S}$, query i may attend to key j if $M_{ij} = 1$. Standard causal attention for language modeling is given by $M_{ij} = 1$ if and only if $j \leq i$. During autoregressive generation, the key and value representations of previous tokens are retained in the KV cache, whose size therefore grows linearly with sequence length.

Prepending memory tokens. MANTA enhances the original sequence by a hierarchy of *memory tokens* at progressively coarser *levels*. All memory tokens are initialized from the same learned embedding and acquire their representations through the hierarchical attention mechanism described below.

We treat the original sequence $\boldsymbol{x} = (\boldsymbol{x}_0, \ldots, \boldsymbol{x}_{S-1})$ as level 0. Given an aggregation factor $c \geq 2$, level 1 contains $\lfloor S/c \rfloor$ memory tokens with hidden representations $\boldsymbol{x}' = (\boldsymbol{x}'_0, \ldots, \boldsymbol{x}'_{\lfloor S/c \rfloor - 1})$. Each $\boldsymbol{x}'_j$ represents a span of c original tokens. The construction is applied recursively: level 2 contains $\lfloor S/c^2 \rfloor$ memory tokens each corresponding to c level 1 memory tokens, and hence to a span of c^2 original tokens. More generally, level ℓ contains $\lfloor S/c^\ell \rfloor$ memory tokens, where each corresponds to c^ℓ original tokens. We write $\boldsymbol{x}_j^{(\ell)}$ for hidden representation j at level ℓ. Levels are added up to $\ell_{\max} = \lfloor \log_c S \rfloor$. The resulting sequence $\boldsymbol{x}^{(\ell_{\max})}, \ldots, \boldsymbol{x}'', \boldsymbol{x}', \boldsymbol{x}$ has length less than $\frac{c}{c-1} S$.

The hierarchical causal attention mask. Apart from the attention mask, the Transformer processes memory tokens identically to ordinary tokens. We design the attention mask to satisfy two requirements: it must preserve causality across hierarchy levels, and it must impose a structured sparsity that bounds the number of key-value pairs retained at each level.

To define causality across levels, we assign each token a timestamp corresponding to the last original token in its span. For level 0, $t(\boldsymbol{x}_j) = j$. For example, with $c = 2$, the first level-1 memory token x'_0 corresponds to original tokens x_0, x_1 and therefore has timestamp $t(\boldsymbol{x}'_0) = 1$, while $t(\boldsymbol{x}'_1) = 3$. In general,

$$t(\boldsymbol{x}_j^{(\ell)}) = (j+1)c^\ell - 1, \qquad \ell = 0, \ldots, \ell_{\max}. \tag{1}$$

The timestamps are illustrated in the right panel of Fig. 1. For a window size w, we define the MANTA attention mask as

$$\boldsymbol{M}(\boldsymbol{q}_j^{(i)}, \boldsymbol{k}_l^{(o)}) = \begin{cases} 1, & 0 \le t(\boldsymbol{q}_j^{(i)}) - t(\boldsymbol{k}_l^{(o)}) < wc^o, \\ 0, & \text{otherwise.} \end{cases} \tag{2}$$

Here, i and o denote the hierarchy levels of the query and key, respectively. The lower bound enforces causality: a query can only attend to keys whose spans end no later than its own. The upper bound implements a fixed window of w tokens on each key level, since consecutive tokens at level o have timestamps spaced by c^o. The attention mask can be viewed in the left panel of Fig. 1. There, attention on the same level is denoted by *self-attention* and indicated by a blue circle, while attention to lower levels is denoted as *aggregation* and indicated by a red circle and attention to higher levels is denoted as *retrieval* and shown as yellow circle.

MANTA can be viewed as multi-scale window attention. At level 0, a window covers w original-token positions; at levels $1, 2, \ldots$, the corresponding temporal ranges grow as $wc, wc^2, \ldots$. Consequently, increasingly distant context remains accessible through progressively coarser representations, while only the most recent w key-value pairs from each level need to be retained.

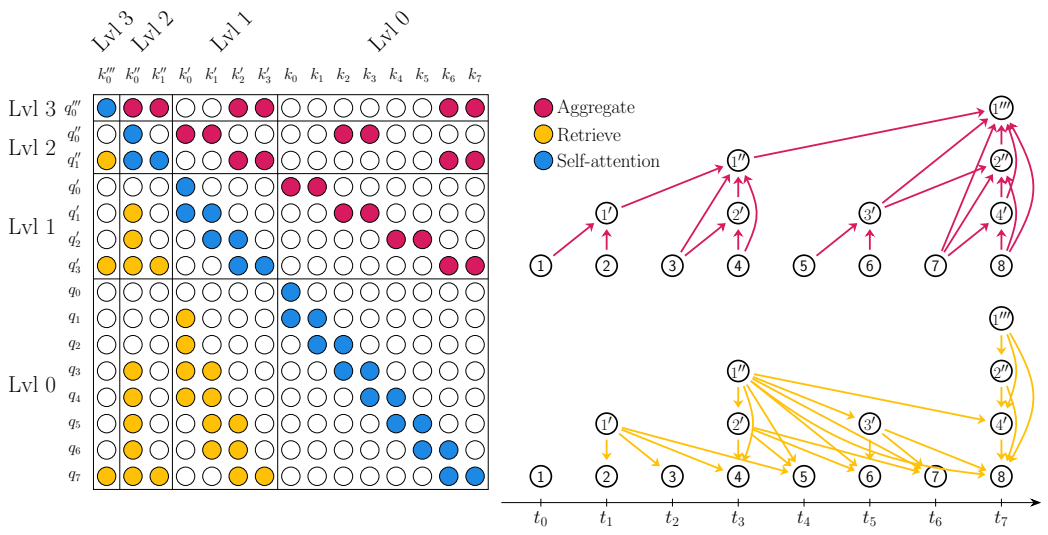


Figure 1: Multi-level attention with $S = 8$, $c = 2$, and $w = 2$. Left: hierarchical attention mask with local within-level and timestamp-based cross-level attention. Right: information flow. Memory tokens aggregate lower-level context and lower-level tokens retrieve from higher-level representations.

Rotary Position Embeddings. Following standard practice in modern Transformers, we use Rotary Position Embeddings (RoPE) (Su et al., 2024). We assign each memory token $\boldsymbol{x}_j^{(\ell)}$ the position of the first original token in its receptive field, $p(\boldsymbol{x}_j^{(\ell)}) = jc^\ell$. Further details are provided in Section A.1.

Performance considerations. MANTA trades additional computation for reduced KV-cache size and memory traffic. For $c = 2$, the number of representations processed by the Transformer nearly doubles; larger values of c reduce this overhead but result in coarser aggregation, yet a smaller KV-cache. This trade-off is particularly relevant during autoregressive decoding, which is often memory-bandwidth bound. Model-weight accesses can be amortized across sequences

through batching, but each sequence maintains an independent KV cache, making KV-cache traffic increasingly important at large batch sizes and context lengths. Reducing the KV cache lowers memory usage and can improve decoding throughput, as demonstrated by our measurements in Fig. 2. We analyze this trade-off with a simple roofline model in Section A.2. FLOPs during training for MANTA are discussed in Section A.3.

4 Language modeling

Training. We use Llama 3.2 1B as the backbone and augment it with the MANTA attention mechanism. We implement the MANTA masks from Section 3 using FlexAttention (Dong et al., 2025). As a baseline we use the original Llama architecture with a full causal mask and additionally a causal window mask.

We train from scratch on 4096–16384-token scientific PDFs from Dolma3 (Team Olmo et al., 2026), amounting to approximately 200B tokens. The documents are tokenized with the Llama 3.2 tokenizer and packed into 8192-token sequences, separated by `EOS`. We train with 524,388 tokens per optimizer step using AdamW ($\beta_1 = 0.9$, $\beta_2 = 0.95$, $\epsilon = 10^{-8}$, weight decay 0.1) and gradient norm clipping at 1.0. The learning rate follows a warmup-stable-decay schedule with 1% warmup, 10% decay, and a maximum learning rate of 10^{-4}. We evaluate different aggregation factors c and window sizes w.

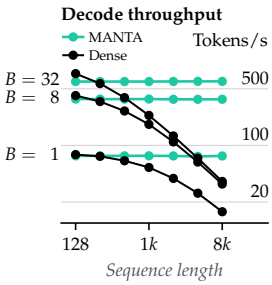


Figure 2: Measured throughput for dense and MANTA $(4/32)$ for different batch sizes.

Table 1: Language modeling benchmark results for two MANTA variants and the respective baselines of dense attention and window attention. NIAH is the mean over all needle-in-a-haystack tasks in RULER and `niah_single_1`.

Attention c/w	KV cache @8K (MiB)	KV-cached states/layer	ARC-Easy ± 0.010	Hella Swag ± 0.005	Wino Grande ± 0.014	Long Bench ± 0.003	RULER @8k
Dense	256.00	8192	0.532	0.449	0.552	0.149	0.585
Window $-/16$	0.50	16	0.533	0.412	0.530	0.067	0.033
MANTA $2/16$	5.47	175	0.533	0.428	0.550	0.114	0.180
MANTA $4/32$	5.31	170	0.547	0.434	0.532	0.119	0.176

Model evaluation. MANTA closely tracks dense attention during training and remains substantially better than window attention, while all models exhibit smooth and stable convergence as seen in Fig. 3. Following prior work by Lin et al. (2025), we evaluate the tokenwise perplexity on 1,000 sequences of length 8K from Dolma. MANTA's perplexity continues to decrease with token position, indicating that it benefits from additional context, whereas window attention levels off at a substantially higher value. The language-modeling benchmarks ARC-Easy (Clark et al., 2018), HellaSwag (Zellers et al., 2019) and WinoGrande (Sakaguchi et al., 2021) in Table 1 show broadly comparable performance across the considered dense and MANTA variants. Reported benchmark uncertainties are estimated standard errors; for the RULER-based benchmarks, where differences between models are much larger, we report the raw scores without attempting a more involved uncertainty analysis. More benchmark results can be found in Section A.5.

Long context evaluation. MANTA loses some performance relative to dense attention on long context benchmarks like LongBench (Bai et al., 2024) and especially RULER (Hsieh et al., 2024), whose NIAH tasks require a small piece of information to remain recoverable after being compressed together with surrounding context. However, Fig. 4 shows that NIAH performance improves steadily throughout training, suggesting that learning to preserve and retrieve such information through the hierarchy develops more slowly than language-modeling performance and has not yet saturated. To inspect this mechanism directly, Fig. 4 shows the mean layer-10 attention over all heads for 1K `niah_single_1` examples. Attention concentrates on memory tokens whose represented spans contain the needle, when possible also across several hierarchy levels, indicating that the model retrieves information through the hierarchy as intended. We also observe attention sinks (Xiao et al., 2024): because softmax attention must distribute its probability mass, high-level memory tokens that

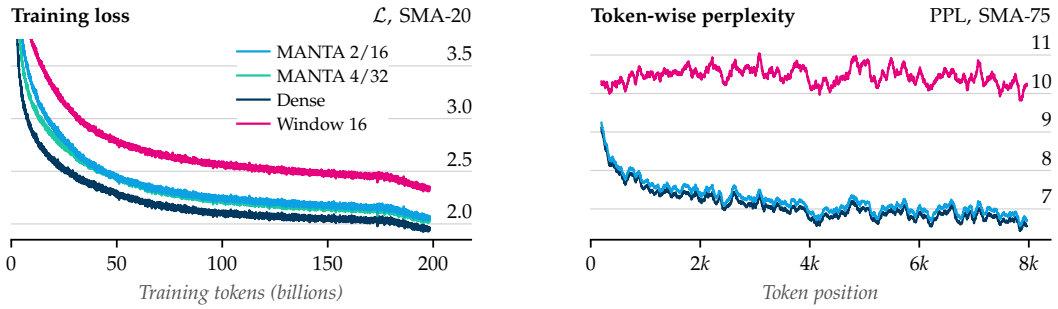


Figure 3: Left: Loss across model variants. Right: Token-wise perplexity evaluated on Dolma.

remain visible for many positions can attract substantial attention even when they carry little relevant information. Retaining the first four original tokens throughout the sequence as dedicated attention sinks largely removes this behavior, allowing attention on the memory hierarchy to become more selective. As this modification did not improve downstream performance, we do not use explicit attention sinks in our experiments.

5 MANTA'S VARIATIONS

MANTA with an unbounded top level. Standard MANTA progressively adds aggregation levels as the sequence grows, retaining a bounded window at each level. We also consider an *unbounded* variant that limits aggregation to a fixed top level $\ell_{\text{top}} < \ell_{\max}$. Attention to levels $0, \ldots, \ell_{\text{top}} - 1$ retains the window size w, while attention to level ℓ_{top} is causal but unbounded to preserve access to distant context. This variant therefore trades the logarithmic scaling of standard MANTA for a fixed aggregation resolution over distant context. The asymptotic KV-cache growth coefficient is reduced by a factor of $c^{\ell_{\text{top}}}$. See Section A.4 for more details.

Fig. 5a compares NIAH and RULER performance for different choices of the top aggregation level. Lowering this level retains more of the history at a finer resolution and brings performance closer to the dense-attention baseline.

MANTA with alternating attention schemes. We also consider a *hybrid* variant that interleaves MANTA layers with dense causal attention layers. This follows the use of hybrid attention architectures that combine mechanisms with different KV-cache requirements (Lenz et al., 2025; Team et al., 2026; DeepSeek-AI, 2026a). We place a dense attention layer at the end of the model and introduce additional dense layers at regular intervals, counting backward from the final layer. All remaining layers use MANTA attention. Placing dense attention towards the end of the model has a higher impact than at the beginning (Puigdemont & Cevher, 2026).

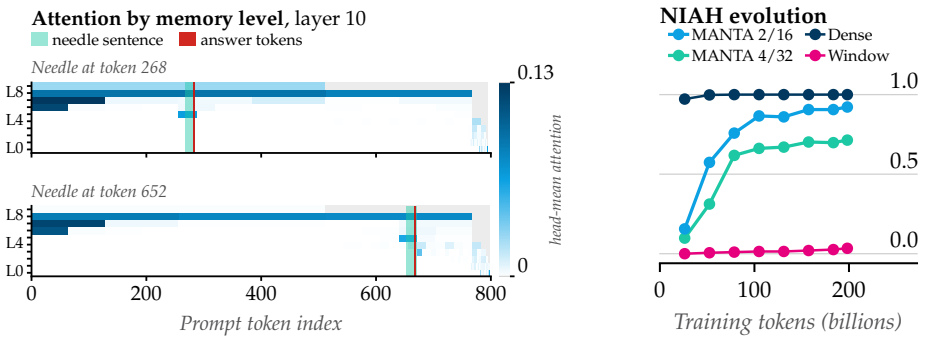


Figure 4: Left: Averaged attention of all heads for layer 10 when predicting the next token of a NIAH task. The green box indicates the needle sentence and the red line the actual needle. We see stronger attention on memory tokens that contain the needle and the use of memory tokens as attention sinks. Right: Evolution of `niah_single_1` results at 4K sequence length during training.

Fig. 5b compares RULER and NIAH performance across different frequencies of dense attention. Adding even a single dense layer improves NIAH performance, while increasing the number of dense layers brings performance closer to the fully dense baseline. The frequency of dense layers therefore provides a further trade-off between retrieval performance and KV-cache size.

MANTA as an autoencoder. MANTA's information routing through memory tokens allows us to train it in the spirit of an autoencoder, despite it being a decoder-only model. For that, we duplicate and pad the input sequence $\boldsymbol{x}_0, \boldsymbol{x}_1, \ldots, \boldsymbol{x}_{S-1}$ with the respective memory tokens. Then we train MANTA on the sequence

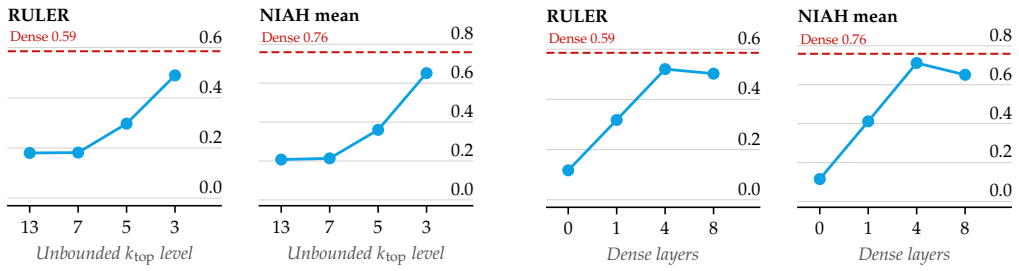
$$\underbrace{\texttt{[mem tokens]} \,|\, \boldsymbol{x}_0 \,|\, \boldsymbol{x}_1 \,|\, \ldots \,|\, \boldsymbol{x}_{S-1}}_{\text{Encoder side}} \,|\, \underbrace{\texttt{[mem tokens]} \,|\, \texttt{BOS} \,|\, \boldsymbol{x}_0 \,|\, \ldots \,|\, \boldsymbol{x}_{S-2}}_{\text{Decoder side}}$$

with the standard language modeling objective applied to the second copy of the sequence. We modify the attention mask such that decoder tokens may only attend to encoder tokens of the top k memory levels as illustrated in Fig 6. Thus, $k = 1$ exposes only the top level memory token of the encoded sequence. Fig. 6 illustrates the mask for $k = 2$. The m exposed tokens take the role of the autoencoder's bottleneck.

We finetune Llama 3.2 1B as a MANTA autoencoder with $c = 2$ and window size $w = 16$. We use the training and validation splits of WikiText-103 Raw v1 (Merity et al. (2016)), partitioning each document into chunks of $\{128, 256, 512\}$ tokens. Shorter samples are discarded. To train the autoencoder, we train for one epoch using a global batch size of 64, AdamW with $\beta_1 = 0.9$, $\beta_2 = 0.95$, $\epsilon = 10^{-8}$ and weight decay 0.1. We use a linear warmup for 25 steps and then a constant learning rate of 3×10^{-5} at sequence length 128, 10^{-4} at 256, and 3×10^{-4} at 512. We further use gradient norm clipping at 1.0. The autoencoder training is implemented without positional encoding.

For evaluation, we use greedy sampling to reconstruct iteratively a sequence $\widehat{x}_0, \ldots, \widehat{x}_{S-1}$ with $\widehat{x}_i$ maximizing $P(\cdot \mid \texttt{BOS}, \widehat{x}_0, \ldots, \widehat{x}_{i-1})$. Strikingly, even with only three visible memory tokens, documents of length 128 can be reconstructed 55% of the time. Further reconstruction percentages for varying numbers of visible memory tokens are provided in the appendix in Table 7. In addition, we use BLEU scores to measure precision of n-grams (Papineni et al., 2002) and BERT scores to quantify semantic similarity (Zhang et al., 2020). In the right panel of Fig. 6 we report the results. While we observe the expected decrease in reconstruction performance with sequence lengths, we confirm here that by exposing only a small number of memory tokens, a faithful compressed representation can be generated.

In addition, we compare MANTA to 500x Compressor, see Li et al. (2025b), an autoencoder with separate encoder and decoder initialized from the same pretrained model. The LoRA-finetuned encoder, see Hu et al. (2022) for LoRA, compresses the input sequence into a fixed number m of memory tokens. Their key and value vectors are passed to the frozen decoder. For a fair comparison, for our MANTA autoencoder we LoRA-finetune Llama 3 8B Instruct with the 500x compressor settings: $\text{rank} = 64$, $\alpha = 32$ and dropout probability 0.05. We match the parameters of MANTA (c, w, k) and 500x compressor (m) so that the decoders receive the same number of memory tokens.



(a) Unbounded top level. (b) Hybrid attention schemes.

Figure 5: RULER and NIAH performance of MANTA variants. (a) Unbounded causal attention at different top aggregation levels. Lower top levels retain distant context at finer resolution but require a larger KV cache. All models use $c = 2$, $w = 16$. (b) Alternating attention schemes with different numbers of dense causal attention layers. The MANTA config is $c = 4$, $w = 16$.

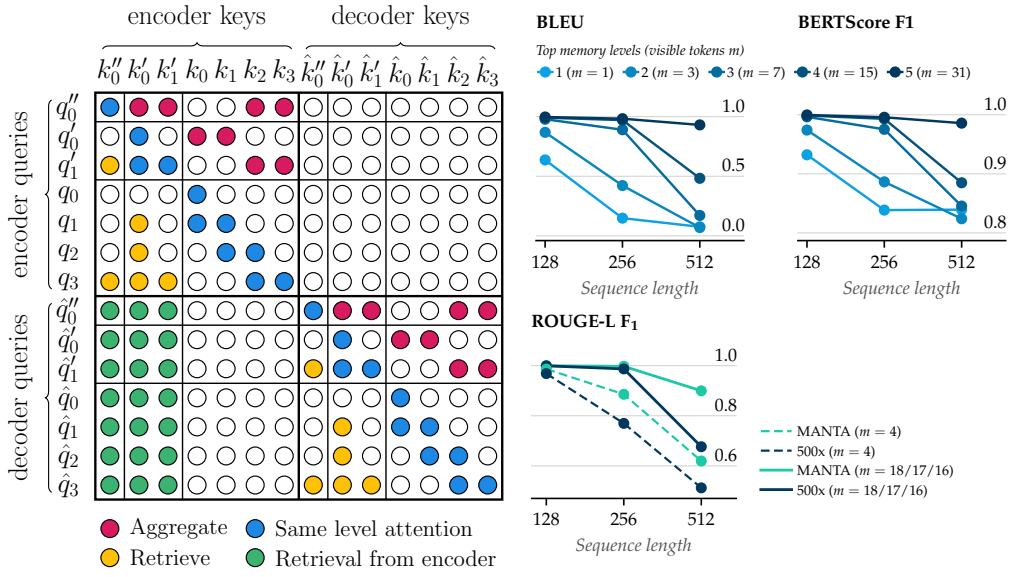


Figure 6: Left: MANTA autoencoder mask: for decoding, MANTA has only access to the top $k = 2$ memory levels of the encoder of the first copy of the sequence. Top right: BLEU and BERT scores after full finetuning of a MANTA autoencoder ($c = 2$, $w = 16$) with varying number of top k visible memory levels. Bottom right: ROUGE-L F1 score of LoRA-finetuned MANTA compared to the 500x compressor model. We use the following MANTA configurations: $m = 4$ with $c = 32/64/128$, $w = 64/128/256$ and $m = 18/17/16$ with $c = 8/16/32$, $w = 16$.

We finetune again on WikiText using a global batch size of 48, warm up for 25 steps up and then a constant learning rate of 3×10^{-4}. The other autoencoder settings remain the same. MANTA is compared to 500x in the lower right panel of Fig. 6 using the ROUGE-L F1 score introduced by Lin (2004) which is based on longest correctly reconstructed subsequences. Under matched settings, MANTA outperforms the 500x compressor model.

MANTA as a byte-level model. We also consider a MANTA variant that restricts middle layers to operate only on memory tokens, as illustrated in Fig. 7. The Transformer layers are divided into three consecutive groups of sizes $[n_1^{\text{in}}, n_2^{\text{mid}}, n_3^{\text{out}}]$. The first n_1^{in} and last n_3^{out} layers process both original and memory tokens using the usual MANTA attention mask. In the middle n_2^{mid} layers, only memory tokens at levels $\ell \geq 1$ are processed by attention and the MLP; the original-token representations pass through unchanged. Compared with processing the full augmented sequence, this reduces the number of active tokens in the middle layers by approximately a factor of c. During cached autoregressive inference, these layers are evaluated only when new memory tokens become available, once every c generated original tokens.

This architecture allows MANTA to operate directly on bytes while processing shorter sequences of aggregated representations in the middle layers similar to MegaByte (Yu et al., 2023) . The first memory level acts as a sequence of learned *quasi-tokens*, each corresponding to a group of c bytes. The UTF-8 encoding of our training dataset contains approximately 4.48 bytes per token on average. Choosing $c = 4$ therefore gives the first memory level a sequence length comparable to that of the tokenized input, while retaining the higher memory levels for further aggregation.

To evaluate this approach, we train a byte-level MANTA model with $c = 4$, $w = 16$ and $[n_1^{\text{in}}, n_2^{\text{mid}}, n_3^{\text{out}}] = [3, 10, 3]$ on 100B token-equivalent bytes. Ten layers operate exclusively on memory tokens, while the first three and last three layers also process the original bytes. We compare this configuration with dense byte- and equivalent token-level baselines in Table 2. The variant with unbounded causal attention at $\ell_{\text{top}} = 2$ achieves performance comparable to dense attention over all bytes, while requiring a smaller KV cache and at a comparable generation throughput. With matched training data size, learned byte-level aggregation outperforms the token-level baseline with a precomputed tokenizer, although this comparison does not control for training compute.

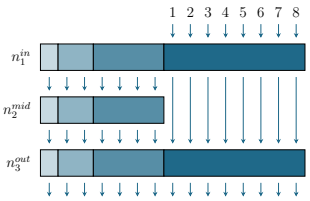


Figure 7: MANTA with memory-only layers. Original token representations bypass the middle layers, memory tokens are processed.

Table 2: Language modeling performance of byte-level models and a token-level baseline. Bits per byte is evaluated on PG-19 (Rae et al. (2019)) and like tokens/s, evaluated at a sequence length of 8K bytes.

Variant	KV cache (MiB)	#Cached (all layers)	Tokens Second	Bits per Byte	ARC-Easy ± 0.01	Hella Swag ± 0.005	Wino Grande ± 0.01
Dense (bytes)	256.00	131,072	14.49	1.087	0.497	0.443	0.537
MANTA $[3, 10, 3]$, $\ell_{\text{top}}=2$	16.69	8,544	29.14	1.122	0.471	0.426	0.537
MANTA $[3, 10, 3]$	2.50	1,280	25.79	1.177	0.442	0.381	0.523
MANTA $4/16$	2.81	1,440	14.23	1.078	0.511	0.444	0.534
Dense (tokens)	57.16	29,264	70.05	1.140	0.498	0.399	0.504

6 Discussion

In this work, we have introduced MANTA, a Multi-level Aggregation Transformer that augments the Transformer with a hierarchy of memory tokens representing increasingly large spans of past context. With a bounded window at each level, this hierarchy yields logarithmic KV-cache growth with sequence length. Timestamp-based masking preserves causality and extends window attention to multiple temporal scales. Information is aggregated locally at each level and retrieved from coarser levels when needed. The same Transformer performs aggregation, retrieval, and generation, keeping the architecture simple. During autoregressive decoding, only the most recent w key-value pairs at each level need to be retained, substantially reducing the KV cache.

We have empirically demonstrated that MANTA can substantially accelerate generation at long context lengths, achieving a speedup of up to $10\times$ at 8K context. Furthermore, we have shown that the hierarchical aggregation mechanism can preserve and retrieve information beyond the reach of local window attention. On NIAH tasks, attention concentrated on memory tokens whose represented spans contained the relevant information, providing qualitative evidence that retrieval proceeded through the hierarchy as intended.

MANTA also supports several architectural variants. By limiting aggregation to a fixed top level and using unbounded causal attention at that level, distant context could be retained at a finer resolution while still reducing the KV cache. With a modified attention mask, the highest-level memory tokens can serve as an autoencoder bottleneck that enables near-lossless compression. Finally, restricting intermediate layers to memory tokens allowed MANTA to operate directly on bytes, replacing conventional tokenization while performing most computation on aggregated representations.

However, these benefits came with a performance trade-off. Compared with dense attention, MANTA generally achieved slightly higher language-modeling loss and lower downstream benchmark scores when trained on the same number of tokens. The discrepancy was more pronounced for long-context performance: on RULER, MANTA did not fully preserve fine-grained information across the 8K context. Nevertheless, the autoencoder experiments showed that, with a dedicated compression objective, a small number of memory tokens can retain substantial information about the original sequence.

In practice, optimizing the architecture of a language model requires balancing memory and compute costs during training and inference against available resources. MANTA introduces an additional degree of freedom in this trade-off: modest architectural complexity can substantially reduce KV-cache requirements and improve decoding efficiency at long context lengths. Hybridizing MANTA with even a single dense-attention layer improved long-context retrieval while retaining much of the KV-cache reduction. These results suggest that MANTA could serve as a useful component in hybrid architectures for efficient long-context language modeling.

Future work could explore broader uses of multi-level aggregation. For example, high-level memory representations could be stored as persistent, composable memory and retrieved in a manner analogous to retrieval-augmented generation, but in a processed and compressed form. Relaxing the causality constraint could extend MANTA beyond language to large-scale, multiscale data such as microscopy images or three-dimensional weather fields. More broadly, MANTA provides a framework for learning hierarchical representations of high-dimensional data across multiple scales.

AI USE STATEMENT

In this work, we used generative AI tools to propose and refine hypotheses, design and provide feedback on research methodology and experiments, implement research methods, assist with translation, and interpret research results. We also used generative AI tools for finding related work, brainstorming and research ideation, drafting sections of the paper, editing and polishing the writing, and creating or editing software code. We did not use generative AI tools to generate synthetic datasets, develop theoretical models or conceptual frameworks, formulate mathematical claims, provide critical ingredients for or assist with mathematical proofs, clean or reformat datasets, or conduct qualitative or thematic data analysis. We have reviewed all AI-assisted work. Relevant content in related work identified by AI was thoroughly read and evaluated by the authors. Research ideas suggested by LLMs were checked for soundness and originality through manual literature review. AI-assisted code was examined, tested for correctness, and reviewed by at least two authors, with changes tracked using version control. AI-drafted content was manually revised. We take responsibility for the final content of this work, including text, claims or artifacts produced with the aid of generative AI.

REPRODUCIBILITY STATEMENT

We provide an anonymous repository containing the source code, dataset preparation scripts and instructions required to reproduce our experiments: https://anonymous.4open.science/r/MANTA-B1B7. The proposed method and experimental setup are described in the paper. The repository will be made public upon acceptance.

REFERENCES

Sandhini Agarwal, Lama Ahmad, Jason Ai, Sam Altman, Andy Applebaum, Edwin Arbus, Rahul K Arora, Yu Bai, Bowen Baker, Haiming Bao, et al. gpt-oss-120b & gpt-oss-20b model card. *arXiv preprint arXiv:2508.10925*, 2025.

Joshua Ainslie, James Lee-Thorp, Michiel De Jong, Yury Zemlyanskiy, Federico Lebrón, and Sumit Sanghai. Gqa: Training generalized multi-query transformer models from multi-head checkpoints. In *Proceedings of the 2023 conference on empirical methods in natural language processing*, pp. 4895–4901, 2023.

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. 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, Bangkok, Thailand, August 2024. Association for Computational Linguistics. doi: 10.18653/v1/2024.acl-long.172. URL https://aclanthology.org/2024.acl-long.172.

Iz Beltagy, Matthew E. Peters, and Arman Cohan. Longformer: The long-document transformer, 2020. URL https://arxiv.org/abs/2004.05150.

Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, et al. Language models are few-shot learners. *Advances in Neural Information Processing Systems*, 33:1877–1901, 2020.

Aydar Bulatov, Yuri Kuratov, and Mikhail S. Burtsev. Recurrent Memory Transformer. In *Advances in Neural Information Processing Systems*, volume 35, pp. 11079–11091, 2022. URL https://proceedings.neurips.cc/paper_files/paper/2022/hash/47e288629a6996a17ce50b90a056a0e1-Abstract-Conference.html.

Yinpeng Chen, DeLesley Hutchins, Aren Jansen, Andrey Zhmoginov, David Racz, and Jesper Andersen. MELODI: Exploring Memory Compression for Long Contexts. In *The Thirteenth International Conference on Learning Representations*. OpenReview.net, 2025. URL https://openreview.net/forum?id=TvGPP8i18S.

Alexis Chevalier, Alexander Wettig, Anirudh Ajith, and Danqi Chen. Adapting language models to compress contexts. In Houda Bouamor, Juan Pino, and Kalika Bali (eds.), *Proceedings of the 2023*

Conference on Empirical Methods in Natural Language Processing, pp. 3829–3846, Singapore, dec 2023. Association for Computational Linguistics. doi: 10.18653/v1/2023.emnlp-main.232. URL https://aclanthology.org/2023.emnlp-main.232/.

Rewon Child, Scott Gray, Alec Radford, and Ilya Sutskever. Generating Long Sequences with Sparse Transformers, apr 2019. URL http://arxiv.org/abs/1904.10509. arXiv:1904.10509 [cs].

Krzysztof Marcin Choromanski, Valerii Likhosherstov, David Dohan, Xingyou Song, Andreea Gane, Tamas Sarlos, Peter Hawkins, Jared Quincy Davis, Afroz Mohiuddin, Lukasz Kaiser, David Benjamin Belanger, Lucy J. Colwell, and Adrian Weller. Rethinking attention with performers. In *International Conference on Learning Representations*, 2021. URL https://openreview.net/forum?id=Ua6zuk0WRH.

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, 2018. URL https://arxiv.org/abs/1803.05457.

Zihang Dai, Zhilin Yang, Yiming Yang, Jaime Carbonell, Quoc Le, and Ruslan Salakhutdinov. Transformer-XL: Attentive language models beyond a fixed-length context. In Anna Korhonen, David Traum, and Lluís Màrquez (eds.), *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pp. 2978–2988, Florence, Italy, jul 2019. Association for Computational Linguistics. doi: 10.18653/v1/P19-1285. URL https://aclanthology.org/P19-1285/.

DeepSeek-AI. DeepSeek-V4: Towards highly efficient million-token context intelligence, 2026a. URL https://arxiv.org/abs/2606.19348.

DeepSeek-AI. DeepSeek-V4.1-Flash: Pushing the limits of KV cache compression, 2026b. URL https://arxiv.org/abs/2609.19969.

DeepSeek-AI, Aixin Liu, Bei Feng, Bin Wang, Bingxuan Wang, Bo Liu, Chenggang Zhao, Chengqi Dengr, Chong Ruan, Damai Dai, Daya Guo, Dejian Yang, Deli Chen, Dongjie Ji, Erhang Li, Fangyun Lin, Fuli Luo, Guangbo Hao, Guanting Chen, Guowei Li, H. Zhang, Hanwei Xu, Hao Yang, Haowei Zhang, Honghui Ding, Huajian Xin, Huazuo Gao, Hui Li, Hui Qu, J. L. Cai, Jian Liang, Jianzhong Guo, Jiaqi Ni, Jiashi Li, Jin Chen, Jingyang Yuan, Junjie Qiu, Junxiao Song, Kai Dong, Kaige Gao, Kang Guan, Lean Wang, Lecong Zhang, Lei Xu, Leyi Xia, Liang Zhao, Liyue Zhang, Meng Li, Miaojun Wang, Mingchuan Zhang, Minghua Zhang, Minghui Tang, Mingming Li, Ning Tian, Panpan Huang, Peiyi Wang, Peng Zhang, Qihao Zhu, Qinyu Chen, Qiushi Du, R. J. Chen, R. L. Jin, Ruiqi Ge, Ruizhe Pan, Runxin Xu, Ruyi Chen, S. S. Li, Shanghao Lu, Shangyan Zhou, Shanhuang Chen, Shaoqing Wu, Shengfeng Ye, Shirong Ma, Shiyu Wang, Shuang Zhou, Shuiping Yu, Shunfeng Zhou, Size Zheng, T. Wang, Tian Pei, Tian Yuan, Tianyu Sun, W. L. Xiao, Wangding Zeng, Wei An, Wen Liu, Wenfeng Liang, Wenjun Gao, Wentao Zhang, X. Q. Li, Xiangyue Jin, Xianzu Wang, Xiao Bi, Xiaodong Liu, Xiaohan Wang, Xiaojin Shen, Xiaokang Chen, Xiaosha Chen, Xiaotao Nie, Xiaowen Sun, Xiaoxiang Wang, Xin Liu, Xin Xie, Xingkai Yu, Xinnan Song, Xinyi Zhou, Xinyu Yang, Xuan Lu, Xuecheng Su, Y. Wu, Y. K. Li, Y. X. Wei, Y. X. Zhu, Yanhong Xu, Yanping Huang, Yao Li, Yao Zhao, Yaofeng Sun, Yaohui Li, Yaohui Wang, Yi Zheng, Yichao Zhang, Yiliang Xiong, Yilong Zhao, Ying He, Ying Tang, Yishi Piao, Yixin Dong, Yixuan Tan, Yiyuan Liu, Yongji Wang, Yongqiang Guo, Yuchen Zhu, Yuduan Wang, Yuheng Zou, Yukun Zha, Yunxian Ma, Yuting Yan, Yuxiang You, Yuxuan Liu, Z. Z. Ren, Zehui Ren, Zhangli Sha, Zhe Fu, Zhen Huang, Zhen Zhang, Zhenda Xie, Zhewen Hao, Zhihong Shao, Zhiniu Wen, Zhipeng Xu, Zhongyu Zhang, Zhuoshu Li, Zihan Wang, Zihui Gu, Zilin Li, and Ziwei Xie. Deepseek-v2: A strong, economical, and efficient mixture-of-experts language model, 2024. URL https://arxiv.org/abs/2405.04434.

Chenlong Deng, Zhisong Zhang, Kelong Mao, Shuaiyi Li, Tianqing Fang, Hongming Zhang, Haitao Mi, Dong Yu, and Zhicheng Dou. Unigist: Towards general and hardware-aligned sequence-level long context compression. In D. Belgrave, C. Zhang, H. Lin, R. Pascanu, P. Koniusz, M. Ghassemi, and N. Chen (eds.), *Advances in Neural Information Processing Systems*, volume 38, pp. 19775–19797. Curran Associates, Inc., 2025. URL https://proceedings.neurips.cc/paper_files/paper/2025/file/1c93b738747776f9d3fdd077a5a07114-Paper-Conference.pdf.

Juechu Dong, Boyuan Feng, Driss Guessous, Yanbo Liang, and Horace He. Flex Attention: a Programming Model for Generating Optimized Attention Kernels. In *Proceedings of the 8th Conference on Machine Learning and Systems*, 2025.

Tao Ge, Hu Jing, Lei Wang, Xun Wang, Si-Qing Chen, and Furu Wei. In-context autoencoder for context compression in a large language model. In *The Twelfth International Conference on Learning Representations*, 2024. URL https://openreview.net/forum?id=uREj4ZuGJE.

Albert Gu and Tri Dao. Mamba: Linear-Time Sequence Modeling with Selective State Spaces. In *First Conference on Language Modeling*, 2024. URL https://openreview.net/forum?id=tEYskw1VY2.

Cheng-Ping Hsieh, Simeng Sun, Samuel Kriman, Shantanu Acharya, Dima Rekesh, Fei Jia, Yang Zhang, and Boris Ginsburg. RULER: What's the Real Context Size of Your Long-Context Language Models? In *First Conference on Language Modeling*, 2024. URL https://openreview.net/forum?id=kIoBbc76Sy.

Edward J. Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, and Weizhu Chen. LoRA: Low-rank adaptation of large language models. In *International Conference on Learning Representations*, 2022. URL https://openreview.net/forum?id=nZeVKeeFYf9.

Sukjun Hwang, Brandon Wang, and Albert Gu. Dynamic chunking for end-to-end hierarchical sequence modeling. In *The Fourteenth International Conference on Learning Representations*, 2026. URL https://iclr.cc/virtual/2026/poster/10008794.

Angelos Katharopoulos, Apoorv Vyas, Nikolaos Pappas, and François Fleuret. Transformers are RNNs: Fast autoregressive transformers with linear attention. In *Proceedings of the 37th International Conference on Machine Learning*, volume 119 of *Proceedings of Machine Learning Research*, pp. 5156–5165. PMLR, 2020. URL https://proceedings.mlr.press/v119/katharopoulos20a.html.

Kimi Team. Kimi Linear: An expressive, efficient attention architecture, 2025. URL https://arxiv.org/abs/2510.26692.

Kimi Team. Kimi K3: Open frontier intelligence, 2026. URL https://arxiv.org/abs/2607.24653.

Xunhao Lai, Weiqi Xu, Yufeng Yang, Qiaorui Chen, Yang Xu, Lunbin Zeng, Xiaolong Li, Haohai Sun, Haichao Zhu, Vito Zhang, Jinkai Hu, Jiayao Li, Rui Gao, Zekun Li, Songquan Zhu, Jingkai Zhou, and Pengyu Zhao. Minimax sparse attention, 2026. URL https://arxiv.org/abs/2606.13392.

Barak Lenz, Opher Lieber, Alan Arazi, Amir Bergman, Avshalom Manevich, Barak Peleg, Ben Aviram, Chen Almagor, Clara Fridman, Dan Padnos, Daniel Gissin, Daniel Jannai, Dor Muhlgay, Dor Zimberg, Edden Gerber, Elad Dolev, Eran Krakovsky, Erez Sa, Erez Schwartz, Gal Cohen, Gal Shachaf, Haim Rozenblum, Hofit Bata, Ido Blass, Inbal Magar, Itay Dalmedigos, Jhonathan Osin, Julie Fadlon, Maria Rozman, Matan Danos, Michael Gokhman, Mor Zusman, Naama Gidron, Nir Ratner, Noam Gat, Noam Rozen, Oded Fried, Ohad Leshno, Omer Antverg, Omri Abend, Or Dagan, Orit Cohavi, Raz Alon, Ro'i Belson, Roi Cohen, Rom Gilad, Roman Glozman, Shahar Lev, Shai Shalev-Shwartz, Shaked Meirom, Tal Delbari, Tal Ness, Tomer Asida, Tom Ben Gal, Tom Braude, Uriya Pumerantz, Joshua Cohen, Yonatan Belinkov, Yuval Globerson, Yuval Levy, and Yoav Shoham. Jamba: Hybrid transformer-mamba language models. In Y. Yue, A. Garg, N. Peng, F. Sha, and R. Yu (eds.), *International Conference on Learning Representations*, volume 2025, pp. 67959–67984, 2025. URL https://proceedings.iclr.cc/paper_files/paper/2025/file/a9ed43fa31dc8b4a7d7a673d713dcb5f-Paper-Conference.pdf.

Yangning Li, Shaoshen Chen, Yinghui Li, Yankai Chen, Hai-Tao Zheng, Hui Wang, Wenhao Jiang, and Philip S Yu. Admtree: Compressing lengthy context with adaptive semantic trees. In D. Belgrave, C. Zhang, H. Lin, R. Pascanu, P. Koniusz, M. Ghassemi, and N. Chen (eds.), *Advances in Neural Information Processing Systems*, volume 38, pp. 40389–40415. Curran Associates, Inc., 2025a. URL https://proceedings.neurips.cc/paper_files/paper/2025/file/39a397e1721555ddfa83cea2de225760-Paper-Conference.pdf.

Yuhong Li, Yingbing Huang, Bowen Yang, Bharat Venkitesh, Acyr Locatelli, Hanchen Ye, Tianle Cai, Patrick Lewis, and Deming Chen. Snapkv: Llm knows what you are looking for before generation. In A. Globerson, L. Mackey, D. Belgrave, A. Fan, U. Paquet, J. Tomczak, and C. Zhang (eds.), *Advances in Neural Information Processing Systems*, volume 37, pp. 22947–22970. Curran Associates, Inc., 2024. doi: 10.52202/079017-0722. URL https://proceedings.neurips.cc/paper_files/paper/2024/file/28ab418242603e0f7323e54185d19bde-Paper-Conference.pdf.

Zongqian Li, Yixuan Su, and Nigel Collier. 500xCompressor: Generalized prompt compression for large language models. In Wanxiang Che, Joyce Nabende, Ekaterina Shutova, and Mohammad Taher Pilehvar (eds.), *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 25081–25091, Vienna, Austria, jul 2025b. Association for Computational Linguistics. ISBN 979-8-89176-251-0. doi: 10.18653/v1/2025.acl-long.1219. URL https://aclanthology.org/2025.acl-long.1219/.

Chin-Yew Lin. ROUGE: A package for automatic evaluation of summaries. In *Text Summarization Branches Out*, pp. 74–81, Barcelona, Spain, 2004. Association for Computational Linguistics. URL https://aclanthology.org/W04-1013/.

Zhixuan Lin, Evgenii Nikishin, Xu He, and Aaron Courville. Forgetting transformer: Softmax attention with a forget gate. In Y. Yue, A. Garg, N. Peng, F. Sha, and R. Yu (eds.), *International Conference on Learning Representations*, volume 2025, pp. 69704–69738, 2025. URL https://proceedings.iclr.cc/paper_files/paper/2025/file/add3d389197ad2267f660ad060ef61f4-Paper-Conference.pdf.

Zirui Liu, Jiayi Yuan, Hongye Jin, Shaochen Zhong, Zhaozhuo Xu, Vladimir Braverman, Beidi Chen, and Xia Hu. KIVI: A tuning-free asymmetric 2bit quantization for KV cache. In *Proceedings of the 41st International Conference on Machine Learning*, volume 235 of *Proceedings of Machine Learning Research*, pp. 32332–32344. PMLR, 2024. URL https://proceedings.mlr.press/v235/liu24bz.html.

Stephen Merity, Caiming Xiong, James Bradbury, and Richard Socher. Pointer sentinel mixture models. *arXiv preprint arXiv:1609.07843*, 2016.

Jesse Mu, Xiang Li, and Noah Goodman. Learning to compress prompts with gist tokens. In A. Oh, T. Naumann, A. Globerson, K. Saenko, M. Hardt, and S. Levine (eds.), *Advances in Neural Information Processing Systems*, volume 36, pp. 19327–19352. Curran Associates, Inc., 2023. URL https://proceedings.neurips.cc/paper_files/paper/2023/file/3d77c6dcc7f143aa2154e7f4d5e22d68-Paper-Conference.pdf.

Piotr Nawrot, Szymon Tworkowski, Michał Tyrolski, Lukasz Kaiser, Yuhuai Wu, Christian Szegedy, and Henryk Michalewski. Hierarchical transformers are more efficient language models. In Marine Carpuat, Marie-Catherine de Marneffe, and Ivan Vladimir Meza Ruiz (eds.), *Findings of the Association for Computational Linguistics: NAACL 2022*, pp. 1559–1571, Seattle, United States, jul 2022. Association for Computational Linguistics. doi: 10.18653/v1/2022.findings-naacl.117. URL https://aclanthology.org/2022.findings-naacl.117/.

Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. BLEU: a method for automatic evaluation of machine translation. In *Proceedings of the 40th Annual Meeting of the Association for Computational Linguistics*, pp. 311–318, Philadelphia, Pennsylvania, USA, 2002. Association for Computational Linguistics. doi: 10.3115/1073083.1073135. URL https://aclanthology.org/P02-1040/.

Pol Puigdemont and Volkan Cevher. Probing hybrid language models for in-context recall. In *Mechanistic Interpretability Workshop at ICML 2026*, 2026. URL https://openreview.net/forum?id=Kifvz6d6Rt.

Jack W Rae, Anna Potapenko, Siddhant M Jayakumar, and Timothy P Lillicrap. Compressive transformers for long-range sequence modelling. *arXiv preprint arXiv:1911.05507*, 2019.

Jack W. Rae, Anna Potapenko, Siddhant M. Jayakumar, Chloe Hillier, and Timothy P. Lillicrap. Compressive transformers for long-range sequence modelling. In *International Conference on Learning Representations*, 2020. URL https://openreview.net/forum?id=SylKikSYDH.

Keisuke Sakaguchi, Ronan Le Bras, Chandra Bhagavatula, and Yejin Choi. Winogrande: an adversarial winograd schema challenge at scale. *Commun. ACM*, 64(9):99–106, August 2021. ISSN 0001-0782. doi: 10.1145/3474381. URL https://doi.org/10.1145/3474381.

Imanol Schlag, Kazuki Irie, and Jürgen Schmidhuber. Linear transformers are secretly fast weight programmers. In *Proceedings of the 38th International Conference on Machine Learning*, volume 139 of *Proceedings of Machine Learning Research*, pp. 9355–9366. PMLR, 2021. URL https://proceedings.mlr.press/v139/schlag21a.html.

Noam Shazeer. Fast transformer decoding: One write-head is all you need, 2019. URL https://arxiv.org/abs/1911.02150.

Jianlin Su, Yu Lu, Shengfeng Pan, Ahmed Murtadha, Bo Wen, and Yunfeng Liu. RoFormer: Enhanced Transformer with Rotary Position Embedding. *Neurocomputing*, 568:127063, 2024. doi: 10.1016/j.neucom.2023.127063. URL https://doi.org/10.1016/j.neucom.2023.127063.

Kimi Team, Tongtong Bai, Yifan Bai, Yiping Bao, Jianfeng Cai, Xinyuan Cai, Peizhou Cao, Yuxuan Cao, Ziwei Chai, Y Charles, et al. Kimi k3: Open frontier intelligence. *arXiv preprint arXiv:2607.24653*, 2026.

Team Olmo, Allyson Ettinger, Amanda Bertsch, Bailey Kuehl, David Graham, David Heineman, Dirk Groeneveld, Faeze Brahman, Finbarr Timbers, Hamish Ivison, Jacob Morrison, Jake Poznanski, Kyle Lo, Luca Soldaini, Matt Jordan, Mayee Chen, Michael Noukhovitch, Nathan Lambert, Pete Walsh, Pradeep Dasigi, Robert Berry, Saumya Malik, Saurabh Shah, Scott Geng, Shane Arora, Shashank Gupta, Taira Anderson, Teng Xiao, Tyler Murray, Tyler Romero, Victoria Graf, Akari Asai, Akshita Bhagia, Alexander Wettig, Alisa Liu, Aman Rangapur, Chloe Anastasiades, Costa Huang, Dustin Schwenk, Harsh Trivedi, Ian Magnusson, Jaron Lochner, Jiacheng Liu, Lester James V. Miranda, Maarten Sap, Malia Morgan, Michael Schmitz, Michal Guerquin, Michael Wilson, Regan Huff, Ronan Le Bras, Rui Xin, Rulin Shao, Sam Skjonsberg, Shannon Zejiang Shen, Shuyue Stella Li, Tucker Wilde, Valentina Pyatkin, Will Merrill, Yapei Chang, Yuling Gu, Zhiyuan Zeng, Ashish Sabharwal, Luke Zettlemoyer, Pang Wei Koh, Ali Farhadi, Noah A. Smith, and Hannaneh Hajishirzi. Olmo 3, 2026. URL https://arxiv.org/abs/2512.13961.

Hugo Touvron, Louis Martin, Kevin Stone, et al. Llama 2: Open foundation and fine-tuned chat models. *arXiv preprint arXiv:2307.09288*, 2023. doi: 10.48550/arXiv.2307.09288. URL https://arxiv.org/abs/2307.09288.

Guangxuan Xiao, Yuandong Tian, Beidi Chen, Song Han, and Mike Lewis. Efficient streaming language models with attention sinks. In B. Kim, Y. Yue, S. Chaudhuri, K. Fragkiadaki, M. Khan, and Y. Sun (eds.), *International Conference on Learning Representations*, volume 2024, pp. 21875–21895, 2024. URL https://proceedings.iclr.cc/paper_files/paper/2024/file/5e5fd18f863cbe6d8ae392a93fd271c9-Paper-Conference.pdf.

Anyi Xu, Bangcai Lin, Bing Xue, Bingxuan Wang, Bingzheng Xu, Bochao Wu, Bowei Zhang, Chaofan Lin, Chen Dong, Chenchen Ling, et al. Deepseek-v4: Towards highly efficient million-token context intelligence. *arXiv preprint arXiv:2606.19348*, 2026.

Zihao Ye, Qipeng Guo, Quan Gan, Xipeng Qiu, and Zheng Zhang. BP-Transformer: Modelling Long-Range Context via Binary Partitioning, nov 2019. URL http://arxiv.org/abs/1911.04070. arXiv:1911.04070 [cs].

Lili Yu, Dániel Simig, Colin Flaherty, Armen Aghajanyan, Luke Zettlemoyer, and Mike Lewis. MEGABYTE: Predicting Million-byte Sequences with Multiscale Transformers. In *Advances in Neural Information Processing Systems*, volume 36, 2023. URL https://proceedings.neurips.cc/paper_files/paper/2023/hash/f8f78f8043f35890181a824e53a57134-Abstract-Conference.html.

Jingyang Yuan, Huazuo Gao, Damai Dai, Junyu Luo, Liang Zhao, Zhengyan Zhang, Zhenda Xie, Yuxing Wei, Lean Wang, Zhiping Xiao, Yuqing Wang, Chong Ruan, Ming Zhang, Wenfeng Liang, and Wangding Zeng. Native sparse attention: Hardware-aligned and natively trainable sparse attention. In Wanxiang Che, Joyce Nabende, Ekaterina Shutova, and Mohammad Taher Pilehvar (eds.), *Proceedings of the 63rd Annual Meeting of the Association for Computational*

Linguistics (Volume 1: Long Papers), pp. 23078–23097, Vienna, Austria, jul 2025. Association for Computational Linguistics. ISBN 979-8-89176-251-0. doi: 10.18653/v1/2025.acl-long.1126. URL https://aclanthology.org/2025.acl-long.1126/.

Manzil Zaheer, Guru Guruganesh, Kumar Avinava Dubey, Joshua Ainslie, Chris Alberti, Santiago Ontanon, Philip Pham, Anirudh Ravula, Qifan Wang, Li Yang, and Amr Ahmed. Big bird: Transformers for longer sequences. In H. Larochelle, M. Ranzato, R. Hadsell, M.F. Balcan, and H. Lin (eds.), *Advances in Neural Information Processing Systems*, volume 33, pp. 17283–17297. Curran Associates, Inc., 2020. URL https://proceedings.neurips.cc/paper_files/paper/2020/file/c8512d142a2d849725f31a9a7a361ab9-Paper.pdf.

Rowan Zellers, Ari Holtzman, Yonatan Bisk, Ali Farhadi, and Yejin Choi. HellaSwag: Can a machine really finish your sentence? In Anna Korhonen, David Traum, and Lluís Màrquez (eds.), *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pp. 4791–4800, Florence, Italy, July 2019. Association for Computational Linguistics. doi: 10.18653/v1/P19-1472. URL https://aclanthology.org/P19-1472/.

Peitian Zhang, Zheng Liu, Shitao Xiao, Ninglu Shao, Qiwei Ye, and Zhicheng Dou. Long context compression with activation beacon. In Y. Yue, A. Garg, N. Peng, F. Sha, and R. Yu (eds.), *International Conference on Learning Representations*, volume 2025, pp. 101932–101948, 2025. URL https://proceedings.iclr.cc/paper_files/paper/2025/file/fc797c61eb18f7b3ce9a74af0ef0d876-Paper-Conference.pdf.

Tianyi Zhang, Varsha Kishore, Felix Wu, Kilian Q. Weinberger, and Yoav Artzi. BERTScore: Evaluating text generation with BERT. In *International Conference on Learning Representations*, 2020. URL https://openreview.net/forum?id=SkeHuCVFDr.

A Appendix

A.1 MANTA RoPE

The positional encoding RoPE (Su et al., 2024) assumes the hidden dimension d to be even. Suppose $\boldsymbol{x}_m \in \mathbb{R}^d = (\mathbb{R}^2)^{d/2}$ denotes a hidden representation corresponding to the token at position m. Then the RoPE map $R^d_{m,\theta}$ rotates for chosen angles θ_i for $i = 1, \dots, \frac{d}{2}$ the entries of $\boldsymbol{x}$ in the respective copy of $\mathbb{R}^2$ by $m\theta_i$. For standard RoPE one uses $\theta_i = 10\,000^{-2i/d}$, whereas Llama 3's frequency-aware scaled RoPE rescales the angles for large i. A key feature of RoPE is that when applied to query and key vectors the scalar products in the attention mechanism do not depend on their absolute position in the sequence but only their relative distance. This can be observed from the fact for all positions m, n one has

$$\langle R^d_{m,\theta}(\boldsymbol{q}), R^d_{n,\theta}(\boldsymbol{k})\rangle = \langle R^d_{n,\theta}(R^d_{m-n,\theta}(\boldsymbol{q})), R^d_{n,\theta}(\boldsymbol{k})\rangle = \langle R^d_{m-n,\theta}(\boldsymbol{q}), \boldsymbol{k}\rangle.$$

To adapt RoPE for MANTA we need to compare positions on different aggregation levels. By extending the usual position function $p(\boldsymbol{x}^{(0)}_m) = m$ to the higher levels we can set

$$R^d_{m,\theta,\ell}(\boldsymbol{x}^{(\ell)}_m) = R^d_{p(\boldsymbol{x}^{(\ell)}_m),\theta}(\boldsymbol{x}^{(\ell)}_m).$$

Then MANTA RoPE does not depend on absolute positions, but only on relative positions.

There are several plausible choices for p as the memory token at position m on level ℓ represents original tokens positioned in the interval $[mc^\ell, (m+1)c^\ell - 1]$. One might choose, for example, the start position, the end position or the center position of these intervals. We compare these options to using no positional encoding using the training setup from Section 4, see Fig. 8 for their loss curves and Table 3 final loss.

Although the differences are small we observed consistently across different settings that the start position performs slightly better. This is why we chose this as the default MANTA rope. As the end positions of the intervals equal the time stamps from equation 1, one may argue that MANTA already obtains positional information through its attention mask.

Table 3: Final loss for MANTA RoPE variants

Variant	start position	center position	end position	no RoPE
Final loss	2.11	2.13	2.13	2.21

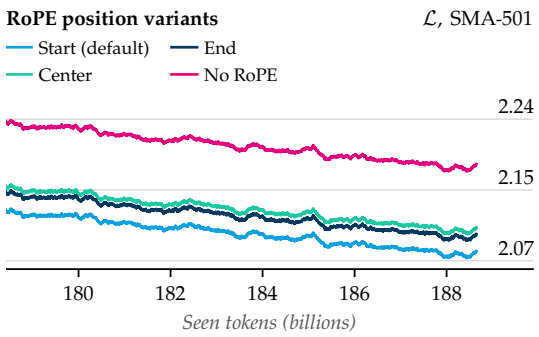


Figure 8: Loss of RoPE variants smoothed with a centered rolling mean over 501 steps

A.2 Memory-bandwidth model for autoregressive decoding

We consider a simple roofline model to illustrate how KV-cache size affects autoregressive decoding throughput. Let B denote the batch size, S the sequence length, M the size of the model parameters in bytes, K the KV-cache size in bytes per token and sequence, and β the available memory bandwidth. Ignoring intermediate activations and other implementation overheads, one decoding step produces B tokens and requires approximately

$$M + BSK$$

bytes to be read from device memory. In the memory-bandwidth-bound regime, the aggregate token throughput is therefore approximately

$$T(B, S) \approx \frac{B\beta}{M + BSK}. \tag{3}$$

When $M \gg BSK$, memory traffic is dominated by the model parameters, and increasing B amortizes the cost of loading the parameters across sequences. Conversely, when $BSK \gg M$, KV-cache accesses dominate and

$$T(B, S) \approx \frac{\beta}{SK},$$

which is independent of batch size. Thus, increasing the batch size eventually ceases to improve aggregate throughput once KV-cache traffic dominates.

MANTA reduces K by retaining only a fixed number of key-value pairs at each hierarchy level. This shifts the crossover to larger batch sizes and sequence lengths, providing a potential throughput benefit in addition to the reduction in memory capacity required for the KV cache. The model is intentionally idealized; realized speedups additionally depend on hardware utilization, attention kernels, and other implementation details.

A.3 Estimate of FLOPs during training

Compared to a usual Transformer, MANTA has an increased sequence length due to the memory tokens whereas attention is cheaper due to MANTA's multi-scale window approach. To compute the number of FLOPs for attention one can estimate the number of active entries in the MANTA mask as follows. First, consider a fixed $\boldsymbol{q}_j^{(i)}$ belonging to level i. Then for each level o and $\boldsymbol{k}_\ell^{(o)}$ one has

$$0 \le t(\boldsymbol{q}_j^{(i)}) - t(\boldsymbol{k}_l^{(o)}) < w \cdot c^o \quad \Longleftrightarrow \quad (j+1)c^{i-o} - w < \ell \le (j+1)c^{i-o}.$$

Consequently, for every level o, there are at most w tokens attending to $\boldsymbol{q}_j^{(i)}$. Therefore, we obtain the following bound for the active entries of the MANTA mask

$$\sum_{i=0}^{\ell_{\max}} \sum_{j=0}^{c^i-1} \sum_{o=0}^{\ell_{\max}} w = w \cdot (\ell_{\max}+1) \cdot \frac{c^{\ell_{\max}+1}-1}{c-1} \leq w \cdot \frac{c}{c-1}(\log_c S + 1)S$$

In Table 4 we give an estimate for the FLOPs executed for one layer in the forward pass for batch size 1 and sequence length S for a Llama 3.2 architecture as backbone. In Fig. 9 the computational costs are plotted for different sequence lengths for the Llama 3.2 1B model. Although MANTA adds overhead for shorter sequences due to the additional memory tokens processed by the MLPs, the computational savings from its log-linear attention outweigh this overhead for longer sequences.

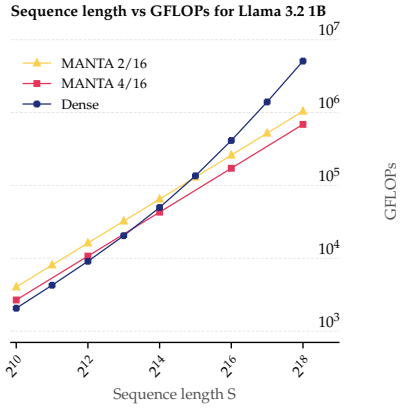


Figure 9: Computational cost (GFLOPs) for one forward path as a function of context size S on a log-log scale.

Table 4: Estimate of FLOPs for the forward pass of one Llama 3.2 Transformer layer for batch size 1, input sequence length S, hidden dimension d, MLP hidden dimension d_{MLP}, number of query heads n_q, number of key/value heads n_{kv} and $r = n_{kv}/n_q$. Here, additions, multiplications and scalar operations are counted as one FLOP and MACs as two FLOPs.

Component	Llama 3.2	Llama 3.2 MANTA
RMSNorm attention	$4Sd$	$4\frac{c}{c-1}Sd$
$\boldsymbol{Q}, \boldsymbol{K}, \boldsymbol{V}$ computation	$2Sd^2(1+2r)$	$2\frac{c}{c-1}Sd^2(1+2r)$
RoPE	$3Sd(1+r)$	$3\frac{c}{c-1}Sd(1+r)$
$\boldsymbol{QK}^\top$ attention scores	$d(S+1)S$	$2dw\frac{c}{c-1}(\log_c S+1)S$
softmax	$2n_q(S+1)S$	$4n_q w\frac{c}{c-1}(\log_c S+1)S$
multiplication by $\boldsymbol{V}$	$d(S+1)S$	$2dw\frac{c}{c-1}(\log_c S+1)S$
output projection	$2Sd^2$	$2\frac{c}{c-1}Sd^2$
RMSNorm MLP	$4Sd$	$4\frac{c}{c-1}Sd$
linear layers MLP (SwiGLU)	$6Sd_{\text{MLP}}d$	$6\frac{c}{c-1}Sd_{\text{MLP}}d$
SiLU activations + gating	$6Sd_{\text{MLP}}$	$6\frac{c}{c-1}Sd_{\text{MLP}}$
Asymptotics for $S \to \infty$	$\mathcal{O}(S^2)$	$\mathcal{O}(S \log S)$

A.4 Unbounded MANTA

Standard MANTA progressively adds aggregation levels as the sequence grows, retaining a bounded window at each level. In the *unbounded* variant aggregation is limited to a fixed top level $\ell_{\text{top}} < \ell_{\max}$. Attention to levels $0, \ldots, \ell_{\text{top}}-1$ retains the window size w, while attention to level ℓ_{top} is causal but unbounded to preserve access to distant context.

Then, each top-level memory token corresponds to a span of $c^{\ell_{\text{top}}}$ original tokens. For a sequence of length S, the top level therefore contains $\lfloor S/c^{\ell_{\text{top}}} \rfloor$ tokens, while each lower level retains at most w key-value pairs. The number of cached key-value pairs per attention layer consequently scales as $\mathcal{O}\big(w\ell_{\text{top}} + S/c^{\ell_{\text{top}}}\big)$. For a fixed top level, KV-cache growth is linear in sequence length, but the sequence-dependent term is reduced by a factor of $c^{\ell_{\text{top}}}$ relative to dense attention.

A.5 Additional language modeling results

In Table 5 and Table 6 we provide further benchmark results.

Table 5: Detailed language modeling results for further MANTA variants

Attention (c/w)	KV cache (MiB)	KV-cached states/layer	Memory tokens @8k	ARC-Easy ± 0.010	Hella Swag ± 0.005	Wino Grande ± 0.014	Long Bench ± 0.003	RULER @8k	NIAH single 1 @8k
Dense	256.00	8192	0	0.532	0.449	0.552	0.149	0.585	1.000
Window ($-/16$)	0.50	16	0	0.533	0.412	0.530	0.067	0.033	0.022
MANTA ($2/16$)	5.47	175	8191	0.533	0.428	0.550	0.114	0.180	0.738
MANTA ($2/32$)	9.97	319	8191	0.547	0.440	0.525	0.119	0.207	0.778
MANTA ($4/16$)	2.81	90	2730	0.521	0.417	0.536	0.103	0.117	0.472
MANTA ($4/32$)	5.31	170	2730	0.547	0.434	0.532	0.119	0.176	0.554
MANTA ($4/64$)	9.31	298	2730	0.522	0.426	0.526	0.121	0.186	0.354
MANTA ($8/16$)	2.06	66	1170	0.520	0.411	0.529	0.097	0.084	0.236
MANTA ($8/32$)	3.56	114	1170	0.505	0.421	0.550	0.103	0.116	0.400
MANTA ($8/64$)	6.56	210	1170	0.520	0.419	0.513	0.112	0.126	0.320
MANTA ($2/16$), $\ell_{\text{top}} = 3$	33.50	1072	7168	0.526	0.448	0.550	0.150	0.489	1.000
MANTA ($2/16$), $\ell_{\text{top}} = 5$	10.50	336	7936	0.546	0.446	0.537	0.126	0.296	0.950
MANTA ($2/16$), $\ell_{\text{top}} = 7$	5.50	176	8128	0.510	0.425	0.540	0.110	0.181	0.920
MANTA ($4/16$), dense every 2	129.41	90/8192	2730	0.527	0.454	0.546	0.141	0.502	1.000
MANTA ($4/16$), dense every 5	66.11	90/8192	2730	0.535	0.438	0.532	0.142	0.520	1.000
MANTA ($4/16$), dense every 16	18.64	90/8192	2730	0.513	0.435	0.528	0.119	0.317	0.920

Table 6: LongBench results by task

LongBench task group	Dense	Window (16)	MANTA ($2/16$)	MANTA ($4/32$)
Code Completion	0.161 ± 0.004	0.103 ± 0.004	0.150 ± 0.005	0.121 ± 0.005
Few-shot Learning	0.382 ± 0.013	0.077 ± 0.007	0.236 ± 0.012	0.272 ± 0.012
Multi-Document QA	0.066 ± 0.003	0.059 ± 0.003	0.071 ± 0.003	0.067 ± 0.003
Single-Document QA	0.073 ± 0.004	0.044 ± 0.002	0.073 ± 0.004	0.077 ± 0.004
Summarization	0.152 ± 0.003	0.096 ± 0.002	0.118 ± 0.002	0.124 ± 0.002
Synthetic Tasks	0.041 ± 0.005	0.036 ± 0.005	0.037 ± 0.005	0.036 ± 0.004

A.6 Additional autoencoder results

Table 7 reports the number of correctly reconstructed documents for different sequence lengths and numbers of visible levels.

Table 7: Correctly reconstructed validation samples for the MANTA ($2/16$) autoencoder. The Total number of reconstructed samples is reported in the parentheses.

Top k levels visible	Visible memory tokens	Length 128	Length 256	Length 512
1	1	23% (456/1952)	0% (0/960)	0% (0/464)
2	3	55% (1068/1952)	6% (55/960)	0% (0/464)
3	7	82% (1597/1952)	34% (328/960)	0% (0/464)
4	15	89% (1745/1952)	60% (572/960)	2% (8/464)
5	31	95% (1854/1952)	74% (708/960)	13% (58/464)