Lost in the Middle: How Language Models Use Long Contexts

Nelson Liu et al., 2023

Today I'll summarize "Lost in the Middle," which investigates how large language models (LLMs) actually use long input contexts. The authors design controlled experiments to test whether models can find and use relevant information when its position varies within the prompt. Their central finding: performance is highest when the answer is at the very beginning or very end of the context, and it drops when the answer sits in the middle—hence "lost in the middle."

—— Tianzhuang Xiong

Motivation

- LLMs can accept very long contexts (4K–100K tokens).
- But: Do they robustly use information anywhere in that context?
- Many real apps (RAG, chat history, long docs) depend on this.

We often assume “bigger window → better reasoning,” especially in retrieval-augmented generation or long document analysis. But if a model is sensitive to where information appears, then simply stuffing more text into the prompt can hurt rather than help. This paper probes that assumption with targeted tests.

Core Research Questions

- How does position of relevant info affect accuracy?
- Does context length change this sensitivity?
- How do architectures (decoder-only vs encoder-decoder) behave?
- Do query-aware prompt layouts help?
- What about instruction fine-tuning?

The authors measure not just raw accuracy but trends as they move relevant passages around and make contexts longer. They also compare architectural families, test prompt formatting tricks, and evaluate models with and without instruction tuning to see what's fundamental versus what can be fixed by prompting

Experimental Tasks (Overview)

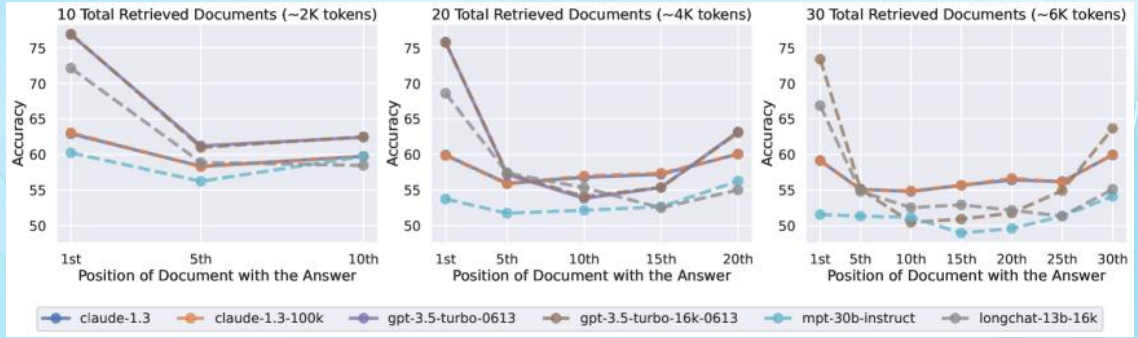
- Multi-Document QA: one relevant passage among distractors.
- Synthetic Key–Value Retrieval: JSON key–value pairs; return v for key k.
- Systematically vary:
  - Context length (e.g., number of docs/pairs)
  - Position of the relevant item (start / middle / end)

Two complementary tasks: a realistic multi-document QA setting where exactly one passage contains the answer, and a minimal, controlled key–value lookup task. In both, they can cleanly move the relevant item across positions to observe performance curves.

Multi-Document QA: Setup

- Questions from NaturalQuestions-Open; passages from Wikipedia.
- Exactly one passage has the answer; others are distractors.
- Distractors are topically relevant (retrieved) but incorrect.
- Evaluate accuracy vs. position of the correct passage.

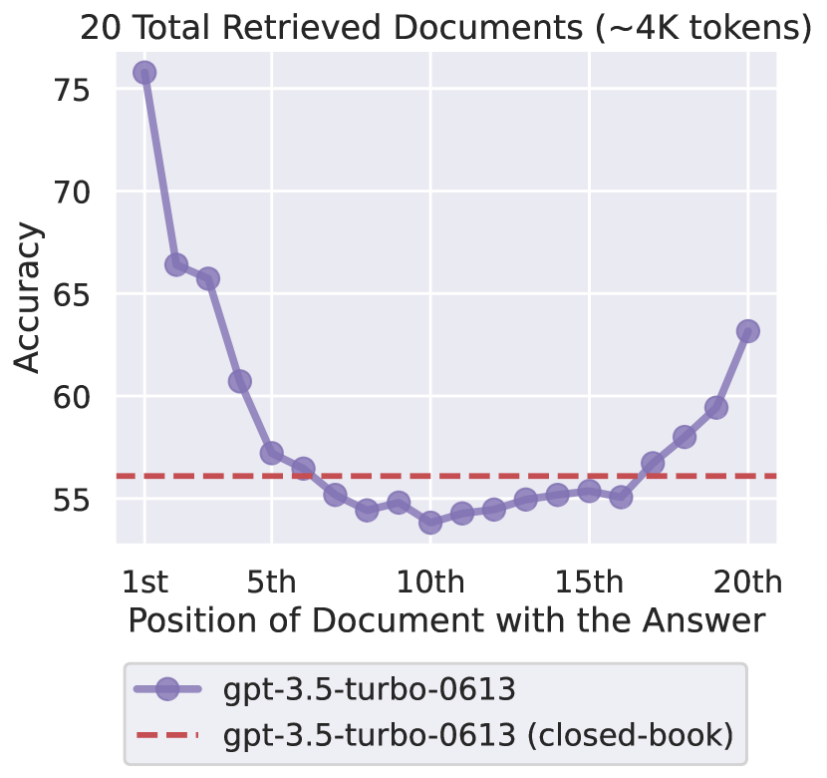
For each question, they place the answer passage at the beginning, middle, or end of the input, with high-quality distractors elsewhere. This simulates a retriever + LLM pipeline where the reader must use the provided evidence—not its parametric memory—to answer.



Key Finding #1 (U-Shaped Curve)

- Accuracy is highest when evidence is at the start or end.
- Accuracy drops when evidence is in the middle (“lost in the middle”).
- Seen across several models (open & closed)

Results show a distinctive U-shaped curve: primacy bias at the beginning and recency bias at the end. In the middle, models often underperform even a closed-book baseline, meaning the long context can hurt performance if the crucial text sits mid-prompt. This happens across multiple model families.



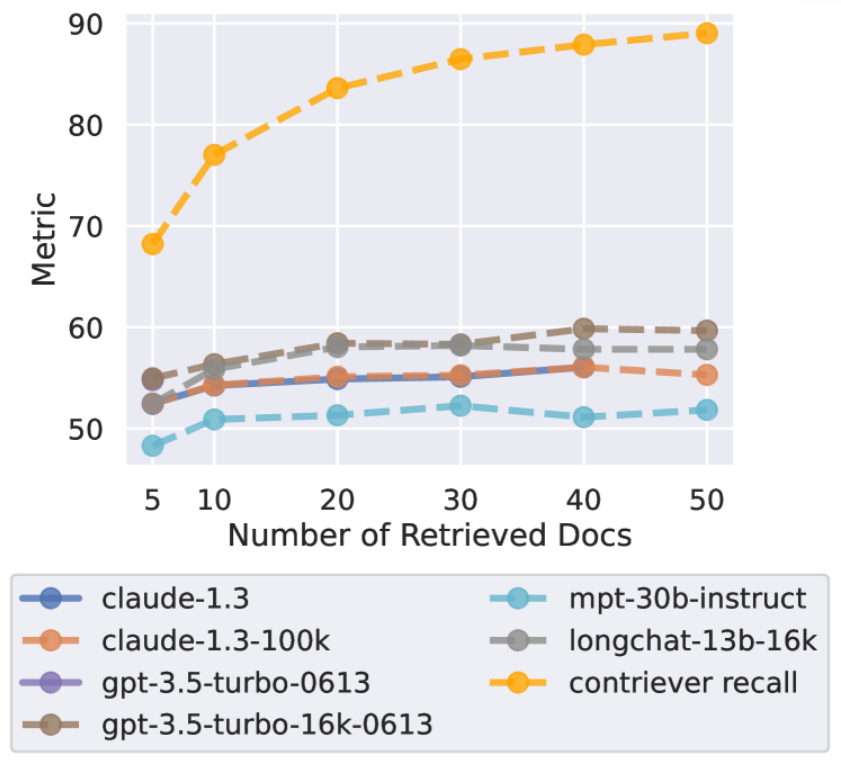
Extended Context ≠ Robust Use

- "Long-context" variants don't automatically fix the problem.
- Models with larger windows can still struggle to use mid-context info.
- Adding more documents saturates performance quickly.

Extending the context window is not a silver bullet.

The authors observe that, for open-domain QA with retriever-reader pipelines, model performance saturates

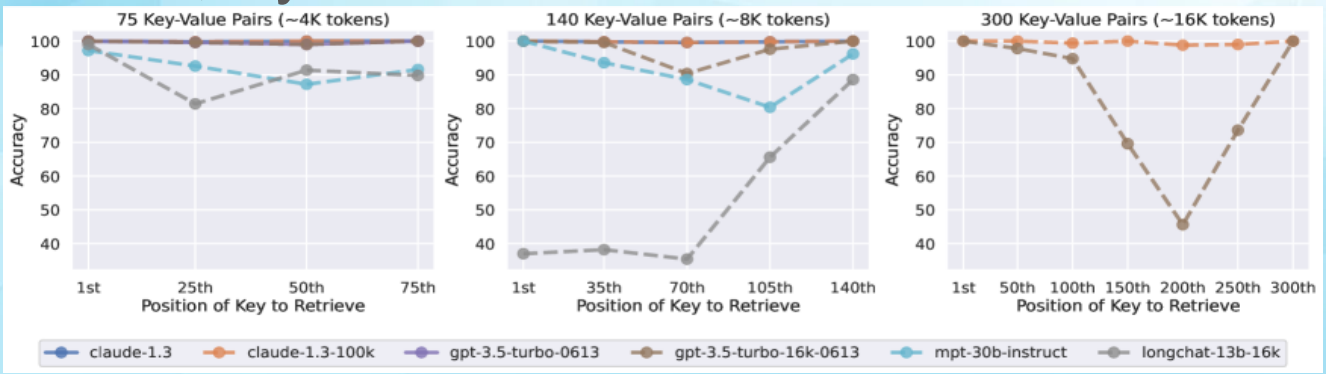
well before retriever recall does: going from 20 to 50 retrieved docs yields only marginal gains, implying the reader fails to benefit from extra material.



Synthetic Key–Value Task

- Minimal test of exact retrieval (k→v).
- Some models still show U-shape: middle positions are hardest.
- However, query-aware formatting can help a lot here.

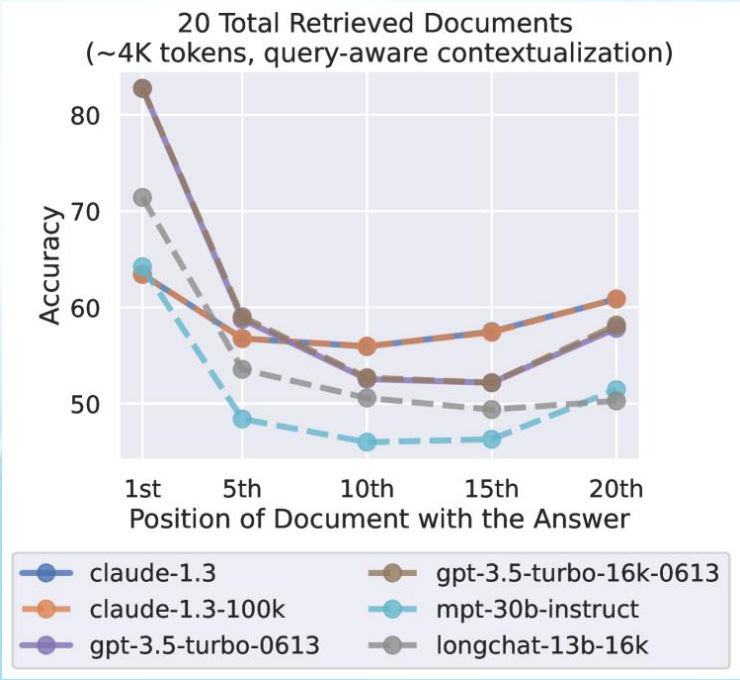
Even in a very structured setting—just matching a key to its value—some models falter when the pair sits mid-context. This indicates the phenomenon isn't only about natural language messiness; it can appear in clean, synthetic formats too.



Architecture Matter (But...)

- Encoder–decoder models are more position-robust within their training-time sequence length.
- When tested beyond that length, they also show U-shapes.

Encoder–decoder models, which encode the whole input before decoding, are less sensitive to position—but only up to the context lengths they were trained on. Once you push them beyond those lengths, the same U-shaped degradation emerges, highlighting fundamental long-range limitations.



Query-Aware Contextualization

- Place the query both before and after the documents/pairs.
- Near-perfect on synthetic key–value retrieval.
- Minimal change to trends in multi-doc QA.

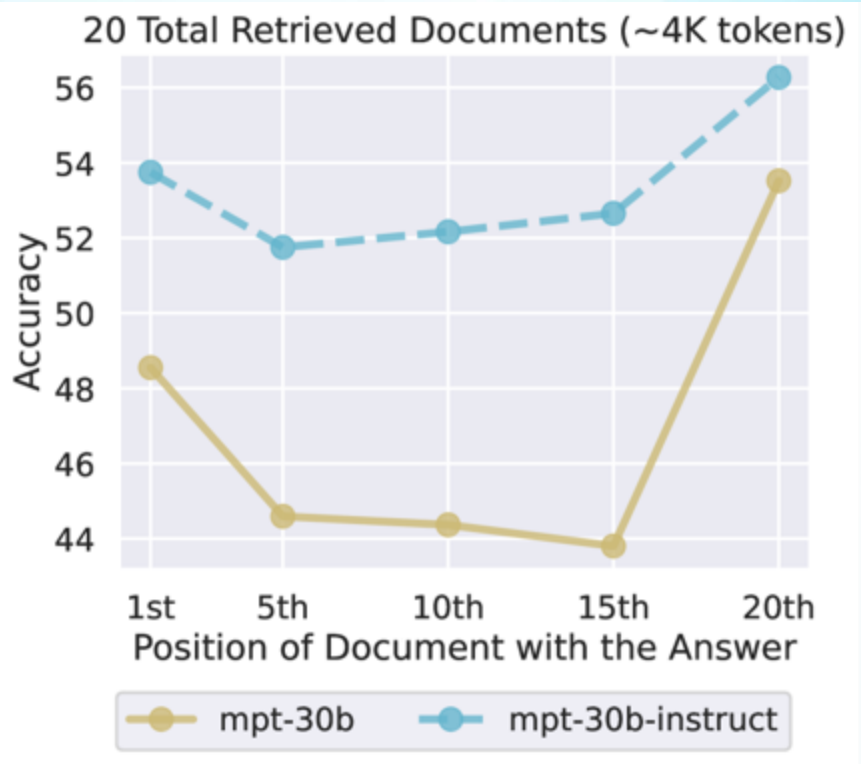
A practical prompting trick—repeat the question at both ends—dramatically helps the key–value task but doesn’t fundamentally change the position-sensitivity

trends in multi-document QA. In realistic reading scenarios, better formatting helps a bit, but it doesn’t solve the core problem.

Instruction Tuning is Not the Fix

- Even base models (no instruction tuning) show U-shapes.
- Suggests a fundamental limitation in attention/position generalization.

Because the U-shape appears even without instruction tuning, the issue seems rooted in how models represent and attend over long sequences, not simply their willingness to follow instructions. Tuning improves helpfulness; it doesn't magically grant robust long-range retrieval.



Practical Implications

- RAG pipelines: Re-rank so the best evidence is first (or last).
- Prompt design: Put summaries/quotes near the query; optionally repeat the query at the end.
- Trim irrelevant text; avoid unnecessary length.
- Prefer chunking/sliding windows or hierarchical aggregation.
- Don’t assume longer window ⇒ better accuracy.

If you control the prompt, bias it so key evidence is early (or echoed late). Pre-summarize, highlight, or quote the most relevant spans. Keep contexts lean. For documents you don’t control, use retrieval + re-ranking to bring the best passages forward. Consider hierarchical or multi-step reading, rather than a single monolithic prompt.

Why “U-Shaped”? (Intuition)

- Attention dilution: more tokens = harder to focus on the right span.
- Positional encoding limits: extrapolation beyond training length is fragile.
- Boundary effects: starts/ends get disproportionate attention.
- Cognitive analogy: primacy/recency effects in memory.

The paper discusses empirical trends rather than mechanistic proofs, but these factors are consistent with prior understanding: attention budgets dilute with length, and many position encoding schemes struggle when extrapolating. Start/end tokens also benefit from structural prominence during generation.

Limitations & Scope

- Benchmarks emphasize English, specific models, and tasks.
- Multi-doc QA uses one relevant passage by design.
- Results don’t claim impossibility—just current model behavior.

As with any controlled study, external validity has bounds. Still, the consistency of the U-shape across tasks and models suggests a robust phenomenon worth addressing in both research and practice.

Takeaways & Looking Forward

- Key message: Models are not position-robust over long contexts; they favor beginnings and ends.
- Engineering guidance: curate/re-rank, front-load evidence, repeat the query, and keep the prompt concise.
- Research directions: better long-range attention, position encodings with reliable extrapolation, training curricula for long contexts, and aggregation/reading strategies.

To claim a model can truly use long contexts, we should test not only maximum window size but position robustness: minimal spread between best- and worst-case evidence positions. Future work could combine improved architectures with training regimes and interface designs that emphasize robust retrieval across the entire prompt.

LongNet: Scaling Transformers to 1,000,000,000 Tokens

Oct 14, 2025

Yusheng Tan

The outline of today's presentation

- Motivation
- Proposed Method
- Engineering Implementation
- Results
- Take away

The outline of today's presentation

- Motivation
- Proposed Method
- Engineering Implementation
- Results
- Take away

Motivation / Problem

Scaling sequence length is the last frontier in neural network scaling.
While depth and width have been successfully scaled (e.g., deep networks, MoE models), sequence length remains fundamentally constrained by computational limits.

Existing long-sequence models face a trade-off between efficiency and expressivity.
RNNs and state-space models handle long sequences efficiently but lack the expressivity of Transformers, while standard Transformers are powerful but computationally infeasible for long contexts.

Quadratic attention cost severely limits Transformer scalability.
The $O(N^2)$ complexity of self-attention makes it impossible to train or infer with extremely long contexts, restricting the ability to model long-range dependencies.

Goal

Greatly extend the Transformer's context length (to extremely long sequences) in order to better model long-range dependencies and multi-sample or multi-segment prompts.

Break free from the O(N^2) bottleneck of standard attention by introducing sparse *dilated attention*, which reduces computation to near-linear complexity while preserving efficient information flow (logarithmic depth).

Enable practical training on ultra-long contexts by designing distributed training along the sequence dimension, making long-context learning feasible on hardware/communication budgets and compatible with optimizations such as FlashAttention.

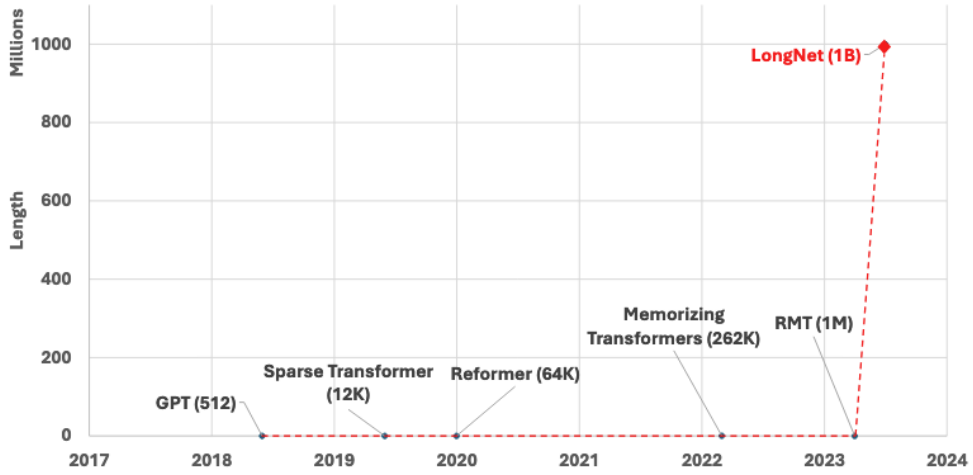


Figure 1: Trend of Transformer sequence lengths over time.

The outline of today's presentation

- Motivation
- Proposed Method
- Engineering Implementation
- Results
- Take away

Proposed Method

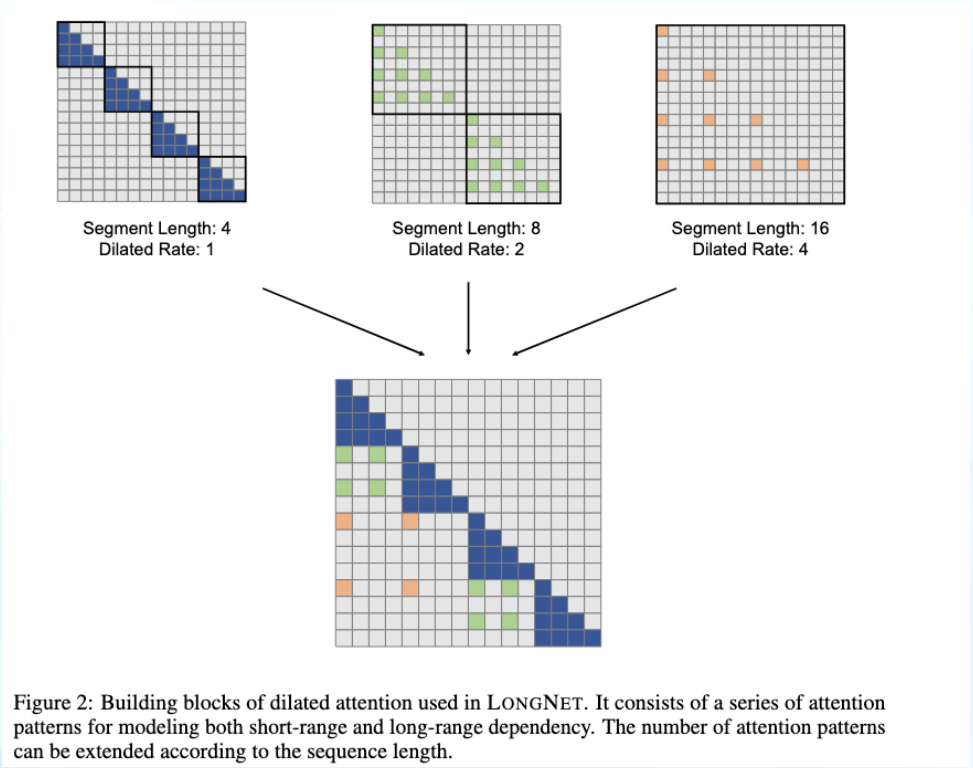


Figure 2: Building blocks of dilated attention used in LONGNET. It consists of a series of attention patterns for modeling both short-range and long-range dependency. The number of attention patterns can be extended according to the sequence length.

Figure explanation:

LongNet introduces *dilated attention* to efficiently capture dependencies across different ranges.

Each segment has a different dilation rate, enabling both short-range and long-range interactions.

Combining these patterns forms a scalable attention mechanism that extends with sequence length.

Dilated attention with multiple heads

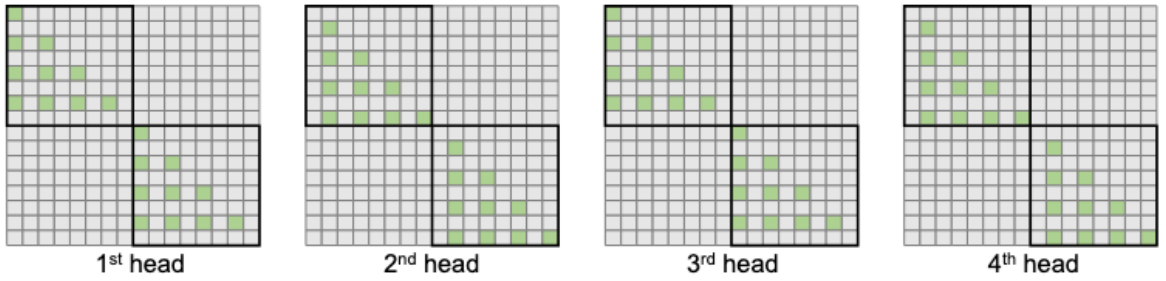


Figure 3: Dilated attention with multiple heads. The attention patterns differ among heads by shifting the position successively.

The core of LongNet lies in combining multi-scale dilated attention with multi-head attention.

Across different scales (with varying dilation rates and segment lengths), the model captures dependencies ranging from local to global contexts.
Within each scale, multiple attention heads are applied.

This hierarchical design allows LongNet to achieve both multi-scale perception and multi-view contextual modeling, while maintaining linear computational complexity for long-sequence processing.

Computational Complexity and Token Dependency

Given dilated attention with a segment size and dilation rate of (r, w), each query-key-value pair is sparsified from $(Q, K, V) \in \mathbb{R}^{N \times d}$ to $(Q, K, V) \in \mathbb{R}^{\frac{w}{r} \times d}$, so the flops of the attention computation are estimated as:

$$FLOPs = \frac{2N}{w}(\frac{w}{r})^2 d = \frac{2Nwd}{r^2} \tag{16}$$

We further extend it to dilated attention with multiple segment sizes and dilation rates. The flops can be written as:

$$FLOPs = 2Nd \sum_{i=1}^{k} \frac{w_i}{r_i^2} \tag{17}$$

With the segment sizes and dilation rates in Equation (11) and Equation (12), the flops are given by

$$FLOPs = 2w_0 Nd \sum_{i=0}^{k-1} \frac{1}{\alpha^i} \leq \frac{2\alpha}{\alpha - 1} w_0 Nd \quad (\alpha > 1) \tag{18}$$

where w_0 is a predefined constant and α is the common ratio for geometric sequences w and r. Therefore, the computation complexity of dilated attention is approximate to $\mathcal{O}(Nd)$.

The outline of today's presentation

- Motivation
- Proposed Method
- Engineering Implementation
- Take away

Engineering Implementation

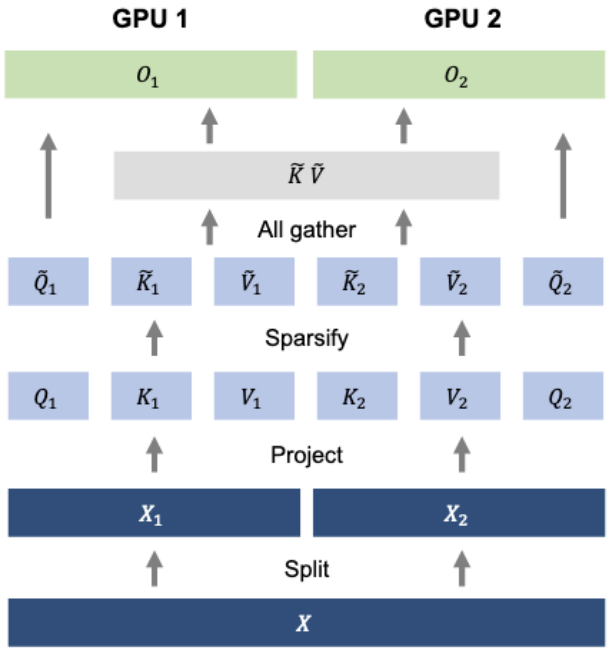


Figure 4: Distributed training of LONGNET on two GPU devices. It parallelizes the training by partitioning the sequence dimension. The computation and communication costs are nearly constant as the number of devices grows.

The figure shows distributed training of LongNet on two GPUs, where the sequence is partitioned across devices and only sparsified key–value pairs are communicated.

This approach easily scales to more GPUs with nearly constant computation and communication costs.

The outline of today's presentation

- Motivation
- Proposed Method
- Engineering Implementation
- Results
- Take away

Model	Length	Batch	Github 2K	Github 8K	Github 32K
Transformer [VSP+17]	2K	256	4.24	5.07	11.29
Sparse Transformer [CGRS19]	8K	64	4.39	3.35	8.79
LONGNET (ours)			4.23	3.24	3.36
Sparse Transformer [CGRS19]	16K	32	4.85	3.73	19.77
LONGNET (ours)			4.27	3.26	3.31
Sparse Transformer [CGRS19]	32K	16	5.15	4.00	3.64
LONGNET (ours)			4.37	3.33	3.01

Table 2: Perplexity of language models for LONGNET and the baselines.

In the experiments, *torchscale* was used as the base library. After replacing the attention layer, a 12-layer model with hidden size 768 was trained and examined.

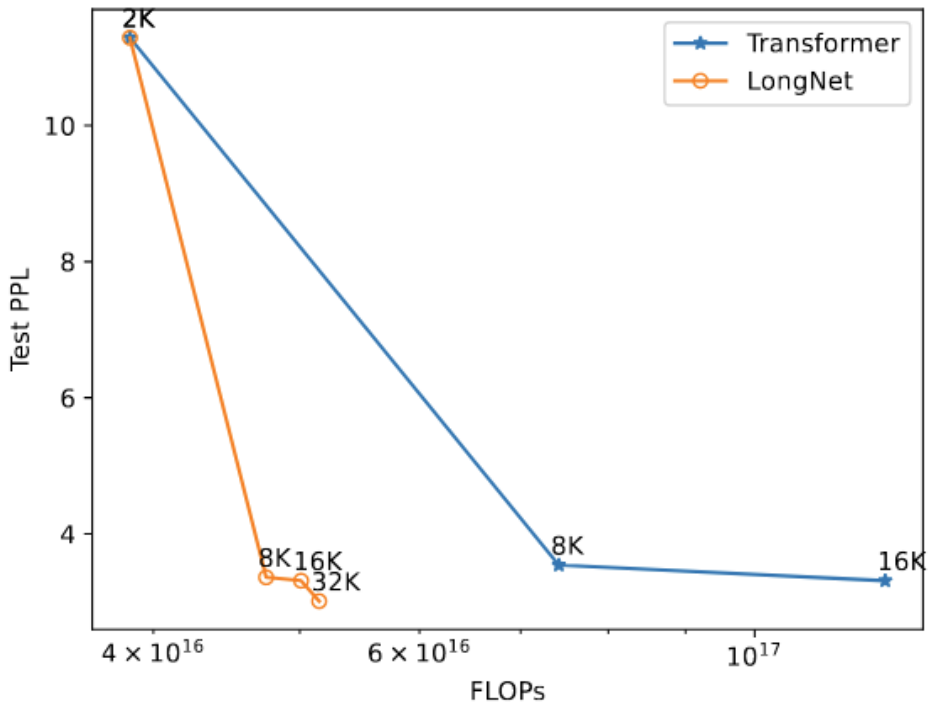


Figure 6: Test perplexity of LONGNET and dense Transformers using different sequence lengths during training. LONGNET outperforms dense Transformers with a lower perplexity and a significantly smaller amount of computation.

In addition to the final perplexity results, the paper also compares the computation required by Transformer and LongNet when processing texts of different lengths.

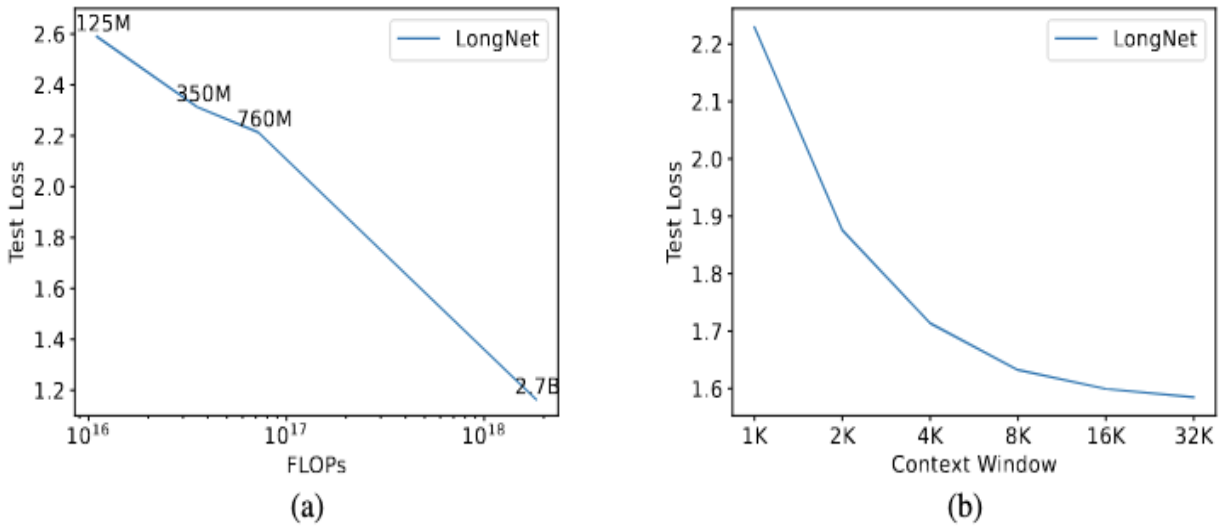


Figure 7: **Left:** Test loss of LONGNET with an increasing model size. The scaling curve follows a similar law to the vanilla Transformers. **Right:** Test loss of LONGNET using different context windows. A longer context window yields better language modeling.

- As the number of parameters increases, the model's perplexity continuously decreases, showing that LongNet has strong scalability.

- As the context window grows larger, the model's performance keeps improving, indicating that LongNet has better comprehension ability for long texts.

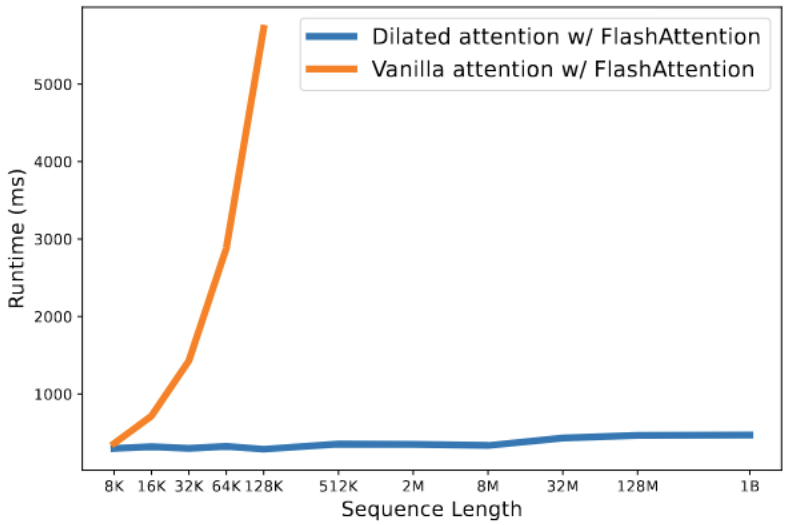


Figure 5: Runtime of our dilated attention and vanilla attention. Both are equipped with FlashAttention [DFE+22].

The results directly demonstrate LongNet's ability to handle long-text processing. Compared with the rapid growth of vanilla Transformers, Dilated Attention shows almost no significant change.

Conclusion — Three Takeaways

- **Scalable Context Expansion**
  LongNet extends Transformer context length to billions of tokens, enabling effective modeling of long-range dependencies and multi-segment prompts.

- **Efficient Sparse Attention**
  Dilated Attention reduces the quadratic cost of standard attention to near-linear while maintaining information flow, making ultra-long context training feasible.

- **Strong Performance & Practicality**
  LongNet shows competitive or improved perplexity across benchmarks, scales efficiently with model size and context window, and remains compatible with distributed training and optimizations like FlashAttention.

RoFormer: Enhanced Transformer with Rotary Position Embedding

Sayee Sreenivas G B

Problem

Transformers are position-agnostic; we must inject position info.

Prior schemes: absolute (sinusoidal/learned) vs relative (bias terms).

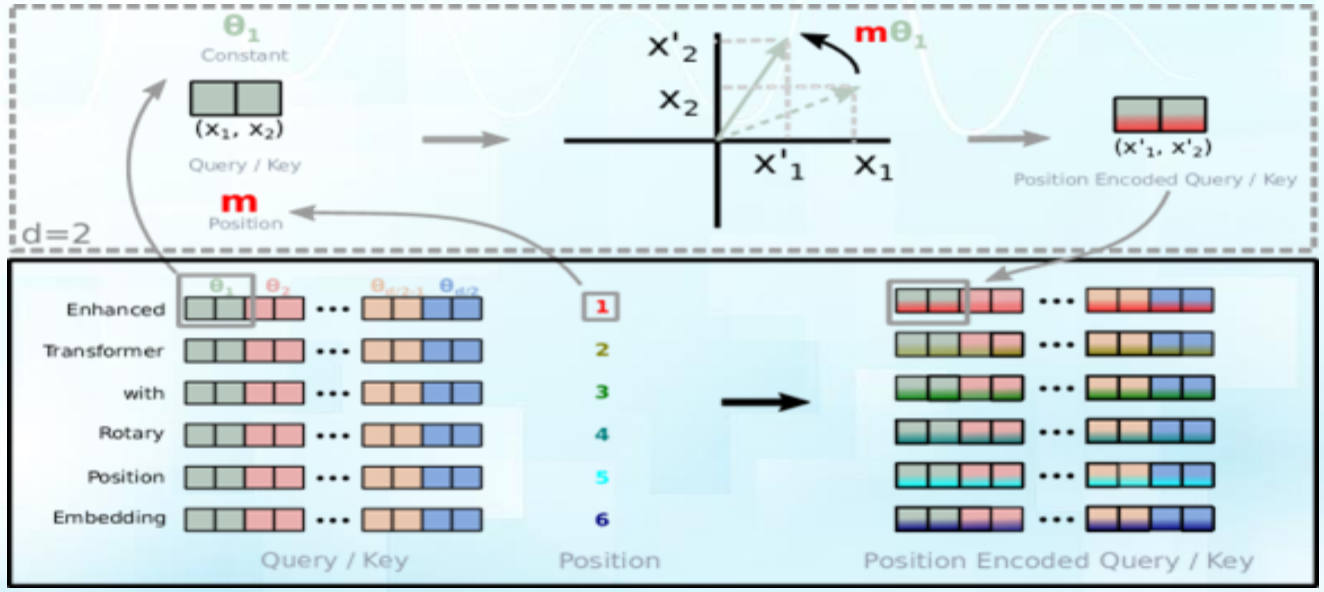
Goal of RoPE: encode absolute index as rotations so that attention scores depend only on relative offsets ($m-n$).

Introduction

RoFormer: Rotary Position Embedding (RoPE):

- A new positional encoding.
- Multiplicative, lightweight, and theory-backed.
- RoFormer (Transformer + RoPE).
- RoPE multiplies positions into Q/K as rotations.

Implementation of RoPE



Left (Q/K blocks): Channel pairs $(2i, 2i+1)$ form tiny 2-D planes; each pair uses its own θ_i.

Middle (Position → Angle): Token index m → angle $m\theta_i$ per pair (vector is rotated on that 2-D plane).

Right (Position-encoded Q/K): Apply rotations to **Q** at m and **K** at n; **V is unchanged**.

In attention, rotations **subtract** → score depends on $(n-m)$.

Proposed Approach

Formulation - (Goal of RoPE)

Problem: Inject positions so attention depends on **relative offset** only.

We want encoders f_q, f_k so the inner product of query/key at positions m, n satisfies

$$\langle f_q(\mathbf{x}_m, m),\ f_k(\mathbf{x}_n, n)\rangle\ =\ g(\mathbf{x}_m, \mathbf{x}_n,\ m-n) \quad \text{(Eq. 11)}$$

Interpretation: attention score should use **content** $(\mathbf{x}_m, \mathbf{x}_n)$ and **relative distance** $(m-n)$, not the absolute indices separately.

Target: find a position encoding that **builds relative dependence directly into** the dot-product, without extra bias terms.

Absolute Position Encoding (Additive)

General additive form:

$$f_t(\mathbf{x}_i, i) = W_t\big(\mathbf{x}_i + \mathbf{p}_i\big),\ t \in \{q, k, v\}.$$

Fixed sinusoidal (Vaswani'17):

$$p_{i,2t} = \sin\big(i/10000^{2t/d}\big),\quad p_{i,2t+1} = \cos\big(i/10000^{2t/d}\big).$$

Learned absolute tables (BERT/RoBERTa): $\mathbf{p}_i$ trainable up to max length L.

Relative Position Embedding

Shaw et al.'18 (clipped relative embeddings):

$$f_q(\mathbf{x}_m) = W_q\mathbf{x}_m,\ f_k(\mathbf{x}_n, n) = W_k(\mathbf{x}_n + \tilde{\mathbf{p}}_r^{(k)}),\ f_v(\cdot) = W_v(\mathbf{x}_n + \tilde{\mathbf{p}}_r^{(v)}),$$
$$r = \mathrm{clip}(m-n, r_{\min}, r_{\max}).$$

Transformer-XL (Dai'19) decomposition of logit:

$$\mathbf{q}_m^\top \mathbf{k}_n = \mathbf{x}_m^\top W_q^\top W_k \mathbf{x}_n + \mathbf{x}_m^\top W_q^\top \tilde{W}_k \tilde{\mathbf{p}}_{m-n} + \mathbf{u}^\top W_k \mathbf{x}_n + \mathbf{v}^\top \tilde{W}_k \tilde{\mathbf{p}}_{m-n}.$$

T5 (Raffel'20) learned relative bias:

$\mathbf{q}_m^\top \mathbf{k}_n + b_{m,n}$ (bucketed relative offsets).

DeBERTa (He'20), Ke'20, Huang'20: variants mixing content & relative terms.

Properties of RoPE

Long-term decay: $\theta_i = 10000^{-2i/d}$.

With RoPE, the attention logit is a sum of oscillators whose phases separate as |m−n| grows ⇒ destructive interference ⇒ the maximum of | q_m^⊤ . decays with distance (see Fig. 2).

Compatibility with linear attention

General attention (per token m) (Eq. 17):

$$\mathrm{Attn}(\mathbf{Q},\mathbf{K},\mathbf{V})_m = \frac{\sum_n \mathrm{sim}(\mathbf{q}_m,\mathbf{k}_n)\,\mathbf{v}_n}{\sum_n \mathrm{sim}(\mathbf{q}_m,\mathbf{k}_n)}.$$

Linearized form (Eq. 18):

$$\mathrm{Attn}_m = \frac{\sum_n \phi(\mathbf{q}_m)^\top \psi(\mathbf{k}_n)\,\mathbf{v}_n}{\sum_n \phi(\mathbf{q}_m)^\top \psi(\mathbf{k}_n)}$$, with non-negative feature maps ϕ, ψ.

RoPE plug-in (Eq. 19): rotate the **feature-mapped** queries/keys (rotations are orthogonal → **norm-preserving**):

$$\mathrm{Attn}_m = \frac{\sum_n (R^{(d)}_{\Theta,m}\phi(\mathbf{q}_m))^\top (R^{(d)}_{\Theta,n}\psi(\mathbf{k}_n))\,\mathbf{v}_n}{\sum_n \phi(\mathbf{q}_m)^\top \psi(\mathbf{k}_n)}.$$

Long-Term Decay of RoPE

Group features into 2-D pairs → RoPE logit is a sum of oscillators

$$\mathbf{q}_m^\top \mathbf{k}_n = \mathrm{Re}\Big[\sum_{i=0}^{d/2-1} h_i\, e^{i\,\Delta\theta_i}\Big] \quad \text{(Eq. 35)}$$

Apply **Abel transform** (summation by parts):

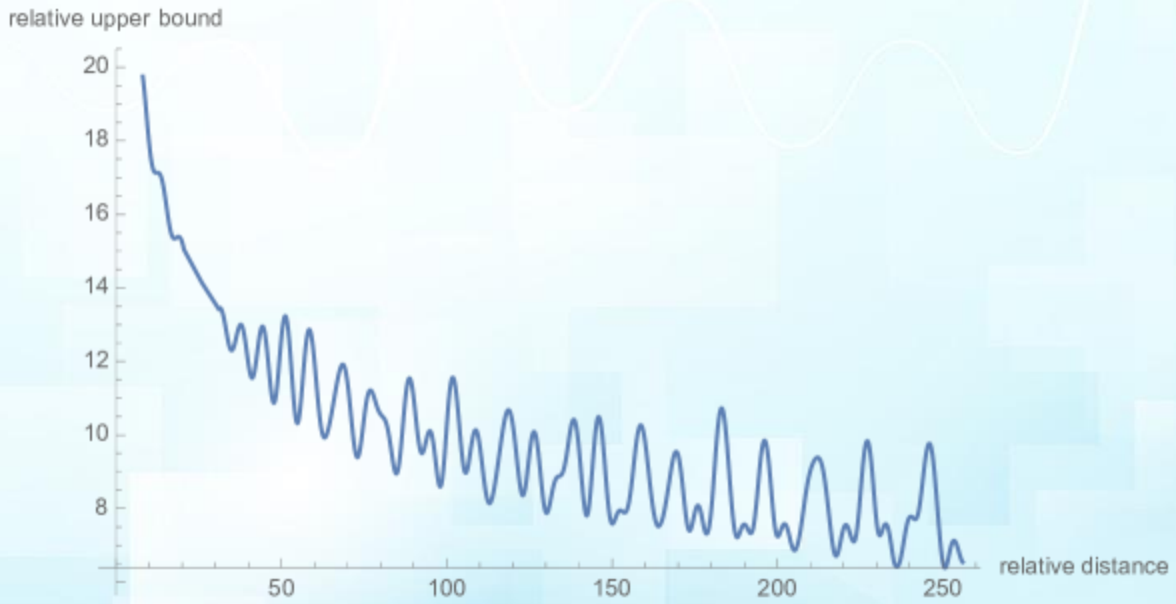
$$\sum_i h_i e^{i\Delta\theta_i} = -\sum_i S_{i+1}\,(h_{i+1}-h_i) \quad \text{with } S_j = \sum_{t=0}^{j-1} e^{i\Delta\theta_t} \quad \text{(Eq. 36)}$$

Triangle inequality bound:

$$\Big|\sum_i h_i e^{i\Delta\theta_i}\Big| \le \big(\max_i |h_{i+1}-h_i|\big) \sum_i |S_{i+1}| \quad \text{(Eq. 37)}$$

RoPE turns positions into phases; as tokens get farther apart, cross-band phase mismatch causes destructive interference, naturally reducing long-range attention.

Long-Term Decay of RoPE - Figure



RoPE naturally down-weights distant tokens but its maximum influence declines with distance, which tends to stabilize training and improve long-context behavior.

Experiments & Evaluation Results

Tasks (by category)

- **Pretrain:** BERT-base on **BookCorpus+Wiki** (MLM)
- **Downstream: GLUE** (MRPC, SST-2, QNLI, STS-B, QQP, MNLI)
- **Linear Attn: Performer** on **Enwik8**

Table 1: The proposed RoFormer gives better BLEU scores compared to its baseline alternative Vaswani et al. [2017] on the WMT 2014 English-to-German translation taskBojar et al. [2014].

Model	BLEU
Transformer-baseVaswani et al. [2017]	27.3
RoFormer	**27.5**

On WMT14 En→De, **RoFormer** slightly outperforms **Transformer-base—27.5 vs 27.3 BLEU** under the same setup—showing a modest but consistent gain from RoPE.

Experiments — Pretraining & GLUE

Pretraining loss curves (Figure: BERT/RoFormer • Performer w/ & w/o RoPE)

- **BERT vs RoFormer (MLM):** RoFormer converges **faster** and to a **lower loss** under the same recipe → swapping sinusoid → **RoPE** helps optimization.
- **Performer (linear attention):** Adding **RoPE** keeps linear complexity and yields **lower loss** across steps → confirms **compatibility** + **benefit** for linear attention.

GLUE downstream (Table: RoFormer vs BERT-base)

- **Wins (3/6): MRPC** 88.9→**89.5** (F1), **STS-B** 85.8→**87.0** (Spearman), **QQP** 71.2→**86.4** (F1).
- **Drops: SST-2** 93.5→90.7, **QNLI** 90.5→88.0, **MNLI m/mm** 84.6/83.4→80.2/79.8.
- **Takeaway:** RoPE gives **clear gains** on **semantic similarity/paraphrase** tasks and improves **optimization speed**, but is **not uniformly better** across all GLUE tasks.

Performer with RoPE

Setup

- Dataset: **Enwik8** (char LM)
- Model: **12-layer Performer**, **d=768**, **12 heads**
- Train: **LR 1e-4**, **bs 128**, **max seq 1024** (same for both)

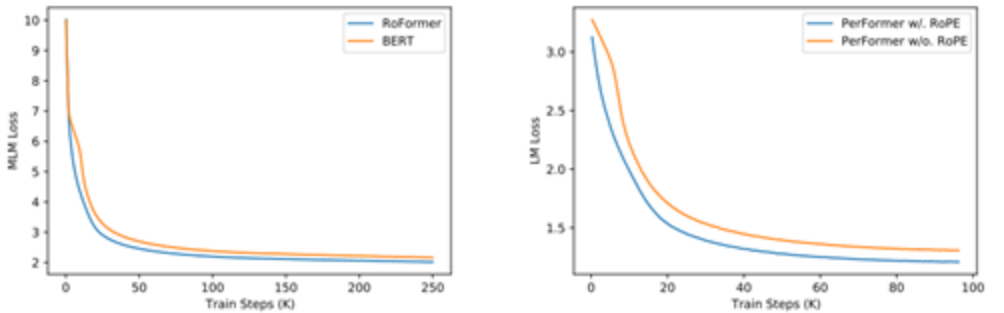


Figure 3: Evaluation of RoPE in language modeling pre-training. **Left**: training loss for BERT and RoFormer. **Right**: training loss for PerFormer with and without RoPE.

Takeaway

RoPE is **compatible with linear attention** and improves optimization on Enwik8.

Chinese Long-Text Evaluation

Table 3: Cross-comparison between our RoFormer and other pre-trained models on Chinese data. 'abs' and 'rel' annotates absolute position embedding and relative position embedding, respectively.

Model	BERTDevlin et al. [2019]	WoBERTSu [2020]	NEZHAWei et al. [2019]	RoFormer
Tokenization level	char	word	char	word
Position embedding	abs.	abs.	rel.	RoPE

Table 5: Experiment results on CAIL2019-SCM task. Numbers in the first column denote the maximum cut-off sequence length. The results are presented in terms of percent accuracy.

Model	Validation	Test
BERT-512	64.13%	67.77%
WoBERT-512	64.07%	68.10%
RoFormer-512	64.13%	68.29%
RoFormer-1024	**66.07%**	**69.79%**

Limitations

RoPE often trains faster than additive PEs, but a full theoretical reason is still open.

Long-term decay is proved, yet why RoPE outperforms peers on long documents isn’t fully explained.

Still a Transformer—pre training requires significant hardware.

Conclusion

Encode absolute position as rotations of Q/K, making attention scores depend only on relative offsets.

Sequence-length flexibility, natural long-range decay, and compatibility with linear attention.

Faster MLM convergence, small but consistent MT gains, mixed-but-positive GLUE results, improved Performer training, and better long-doc accuracy.

RULER: What's the Real Context Size of Your Long-Context Language Models?

Oct 14, 2025

The outline of today's presentation

- Motivation
- Intro to RULER Benchmark
- Experimental Setup
- Results
- Limitation
- Takeaways

The outline of today's presentation

- Motivation
- Intro to RULER Benchmark
- Experimental Setup
- Results
- Limitation
- Takeaways

Motivation

Current long-context evaluations are overly simplistic.
Most works rely on *needle-in-a-haystack (NIAH)* tests, which only measure retrieval ability — whether the model can find a small piece of information from a long distractor text — but this reflects only superficial context usage rather than true long-context understanding.

Inconsistent and incomplete evaluation practices.
Existing benchmarks are used inconsistently across studies (different setups, data, and lengths), making it difficult to compare models fairly or systematically assess progress in long-context learning.

Need for richer, more diagnostic tasks.
Long-context understanding involves more than retrieval — it includes reasoning across multiple context spans, aggregating dispersed information, and following multi-hop dependencies.
A new benchmark should explicitly test these capabilities.

Empirical evidence of performance degradation.
Despite large claimed context windows (e.g., 32K–200K tokens), existing LMs **degrade sharply** as context length or task complexity increases, showing that **long-context scaling is far from solved**.

The outline of today's presentation

- Motivation
- Intro to RULER Benchmark
- Experimental Setup
- Results
- Limitation
- Takeaways

Intro to RULER Benchmark

1.Retrieval

2.Multi-hop Tracing

3.Aggregation

4.Question Answering

Benchmark & Task	Avg Len	Type	Diverse Tasks	Min. Parametric Knowledge	Controllable Context
ZeroSCROLLS	~10k	realistic	✓	✗	✗
L-Eval	~8k	realistic	✓	✗	✗
BAMBOO	~16k	realistic	✓	✓	✗
LongBench	~8k	hybrid	✓	✗	✗
LooGLE	~20k	hybrid	✓	✓	✗
InfiniteBench	~200k	hybrid	✓	✓	✗
Needle-in-a-haystack (NIAH)	any	synthetic	✗	✓	✓
Passkey / Line / KV Retrieval	any	synthetic	✗	✓	✓
RULER (Ours)	any	synthetic	✓	✓	✓

Aggregation

In the common word extraction task (CWE), words are sampled from discrete uniform distributions, with the number of common words fixed while the number of uncommon words increases with the sequence length. In the frequent words extraction task (FWE), words are sampled from Zeta distribution.

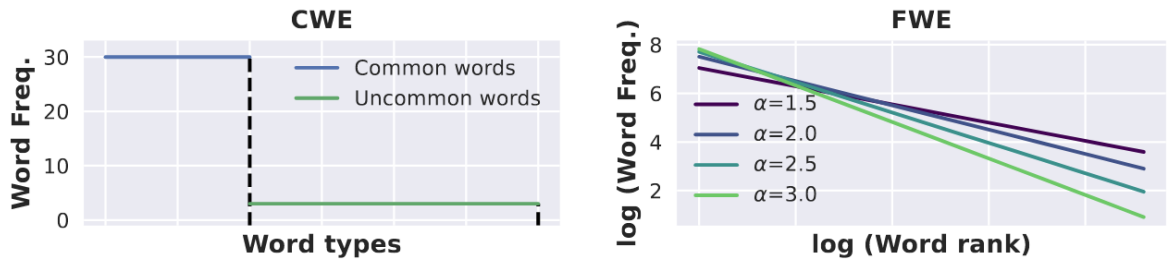


Figure 1: In aggregation tasks, we sample words from a vocabulary following the two distributions above. The common words extraction (CWE) samples from uniform distributions. In the frequent words extraction (FWE), the frequency of each word is determined by its rank in the vocabulary and the parameter α of Zeta distribution.

The outline of today's presentation

- Motivation
- Intro to RULER Benchmark
- Experimental Setup
- Results
- Limitation
- Takeaways

Experimental Setup

17 long-context LLMs, including 15 open-source models and two closed-source model (Gemini-1.5-Pro and GPT-4), covering diverse model sizes (7B to 8x22B with MoE architecture) and claimed context lengths (32K to 1M).

Model	Aligned	Size	Context Length	Huggingface (Wolf et al., 2019) / API
GPT-4 (OpenAI: Josh Achiam et al., 2023)	✓	-	128K	`gpt-4-1106-preview`
Gemini-1.5 (Reid et al., 2024)	✓	-	1M	`gemini-1.5-pro`
Llama3.1 (Meta.AI, 2024b)	✓	70B	128K	meta-llama/Meta-Llama-3.1-70B-Instruct
Llama3.1 (Meta.AI, 2024b)	✓	8B	128K	meta-llama/Meta-Llama-3.1-8B-Instruct
Command-R-plus (Cohere, 2024)	✓	104B	128K	CohereForAI/c4ai-command-r-plus
Qwen2 (Yang et al., 2024)	✓	72B	128K	Qwen/Qwen2-72B-Instruct
Yi (Young et al., 2024)	✓	34B	200K	01-ai/Yi-34B-200K
Mixtral-8x22B (Jiang et al., 2024)	✓	39B/141B	32K	mistralai/Mixtral-8x22B-Instruct-v0.1
Mistral-v0.2 (Mistral.AI, 2023)	✓	7B	32K	mistralai/Mistral-7B-Instruct-v0.2
GLM4 (GLM et al., 2024)	✓	9B	1M	THUDM/glm-4-9b-chat-1m
GradientAI/Llama3 (Meta.AI, 2024a)	✓	70B	1M	gradientai/Llama-3-70B-Instruct-Gradient-1048k
Phi3-medium (Abdin et al., 2024)	✓	14B	128K	microsoft/Phi-3-medium-128k-instruct
LWM (Liu et al., 2024a)	✓	7B	1M	LargeWorldModel/LWM-Text-Chat-1M
DBRX (Databricks, 2024)	✓	36B/132B	1M	databricks/dbrx-instruct
Together (Together.AI, 2023b)	✓	7B	32K	togethercomputer/Llama-2-7B-32K-Instruct
LongChat (Li et al., 2023a)	✓	7B	32K	lmsys/longchat-7b-v1.5-32k
LongAlpaca (Chen et al., 2024)	✓	13B	32K	Yukang/LongAlpaca-13B
Mixtral-base (Jiang et al., 2024)	✗	8x7B	32K	mistralai/Mixtral-8x7B-v0.1
Mistral-base (Mistral.AI, 2023)	✗	7B	32K	alpindale/Mistral-7B-v0.2-hf
LWM-base (Liu et al., 2024a)	✗	7B	1M	LargeWorldModel/LWM-Text-1M
LongLoRA-base (Chen et al., 2024)	✗	7B	100K	Yukang/Llama-2-7b-longlora-100k-ft
Yarn-base(Peng et al., 2024)	✗	7B	128K	NousResearch/Yarn-Llama-2-7b-128k
Together-base (Together.AI, 2023a)	✗	7B	32K	togethercomputer/Llama-2-7B-32K
Jamba-base (AI21, 2024)	✗	52B	256K	ai21labs/Jamba-v0.1
Llama2 (chat) (Touvron et al., 2023)	✓	7B	4K	meta-llama/Llama-2-7b-chat-hf
Llama2 (base) (Touvron et al., 2023)	✗	7B	4K	meta-llama/Llama-2-7b-hf
Yi series (Young et al., 2024)	✓	6B,9B	200K	01-ai/Yi-(6B,9B)-200K
LWM series (Liu et al., 2024a)	✓	7B	128K,256K,512K	LargeWorldModel/LWM-Text-Chat-(128K,256K,512K)
LWM-base series (Liu et al., 2024a)	✗	7B	32K,128K,256K,512K	LargeWorldModel/LWM-Text-(32K,128K,256K,512K)
Mamba (Gu & Dao, 2023)	✗	2.8B	2K	state-spaces/mamba-2.8b-slimpj
RWKV (Peng et al., 2023)	✗	7B	4K	RWKV/v5-Eagle-7B-HF

Table 4: Information of evaluated and analyzed models in RULER.

The outline of today's presentation

- Motivation
- Intro to RULER Benchmark
- Experimental Setup
- Results
- Limitation
- Takeaways

Main Results

Models	Claimed Length	Effective Length	4K	8K	16K	32K	64K	128K	Avg.	wAvg. (inc)	wAvg. (dec)
Llama2 (7B)	4K	-	85.6								
Gemini-1.5-Pro	1M	>128K	96.7	95.8	96.0	95.9	95.9	94.4	95.8	95.5 (1st)	96.1 (1st)
GPT-4	128K	64K	96.6	96.3	95.2	93.2	87.0	81.2	91.6	89.0 (2nd)	94.1 (2nd)
Llama3.1 (70B)	128K	64K	96.5	95.8	95.4	94.8	88.4	66.6	89.6	85.5 (4th)	93.7 (3rd)
Qwen2 (72B)	128K	32K	96.9	96.1	94.9	94.1	79.8	53.7	85.9	79.6 (9th)	92.3 (4th)
Command-R-plus (104B)	128K	32K	95.6	95.2	94.2	92.0	84.3	63.1	87.4	82.7 (7th)	92.1 (5th)
GLM4 (9B)	1M	64K	94.7	92.8	92.1	89.9	86.7	83.1	89.9	88.0 (3rd)	91.7 (6th)
Llama3.1 (8B)	128K	32K	95.5	93.8	91.6	87.4	84.7	77.0	88.3	85.4 (5th)	91.3 (7th)
GradientAI/Llama3 (70B)	1M	16K	95.1	94.4	90.8	85.4	80.9	72.1	86.5	82.6 (8th)	90.3 (8th)
Mixtral-8x22B (39B/141B)	64K	32K	95.6	94.9	93.4	90.9	84.7	31.7	81.9	73.5 (11th)	90.3 (9th)
Yi (34B)	200K	32K	93.3	92.2	91.3	87.5	83.2	77.3	87.5	84.8 (6th)	90.1 (10th)
Phi3-medium (14B)	128K	32K	93.3	93.2	91.1	86.8	78.6	46.1	81.5	74.8 (10th)	88.3 (11th)
Mistral-v0.2 (7B)	32K	16K	93.6	91.2	87.2	75.4	49.0	13.8	68.4	55.6 (13th)	81.2 (12th)
LWM (7B)	1M	<4K	82.3	78.4	73.7	69.1	68.1	65.0	72.8	69.9 (12th)	75.7 (13th)
DBRX (36B/132B)	32K	8K	95.1	93.8	83.6	63.1	2.4	0.0	56.3	38.0 (14th)	74.7 (14th)
Together (7B)	32K	4K	88.2	81.1	69.4	63.0	0.0	0.0	50.3	33.8 (15th)	66.7 (15th)
LongChat (7B)	32K	<4K	84.7	79.9	70.8	59.3	0.0	0.0	49.1	33.1 (16th)	65.2 (16th)
LongAlpaca (13B)	32K	<4K	60.6	57.0	56.6	43.6	0.0	0.0	36.3	24.7 (17th)	47.9 (17th)

Table 3: Long Context Performance (%) of selected models evaluated at length from 4K to 128K. Each score is computed by averaging accuracy of 13 tasks in RULER. The performance exceeding the Llama2-7B performance at 4K (85.6%) is underlined. The effective context length is the maximum length passing this threshold. Weighted average score (wAvg.) aggregates performance across all context sizes, with the weights linearly increasing (inc) or decreasing (dec) to simulate length distribution of real-world usage. We put the rank of each model in the subscript. More details about the selected models are in Appendix A.

The outline of today's presentation

- Motivation
- Intro to RULER Benchmark
- Experimental Setup
- Results
- Limitation
- Takeaways

Task error analysis

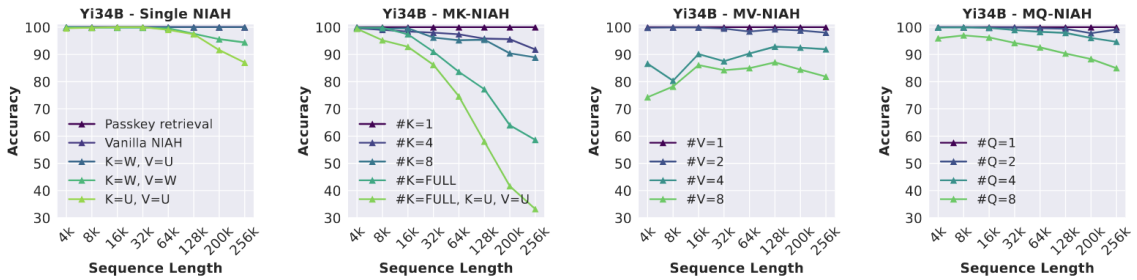


Figure 2: Performance of Yi-34B in the needle-in-a-haystack (NIAH) tasks. By default, we use word-number as the key-value pair and Paul Graham essays as the haystack. Yi is not robust to the change of needle types and degrades with the increasing amount of distractors. (W: words; N: numbers; U: UUIDs; Full: entire haystack).

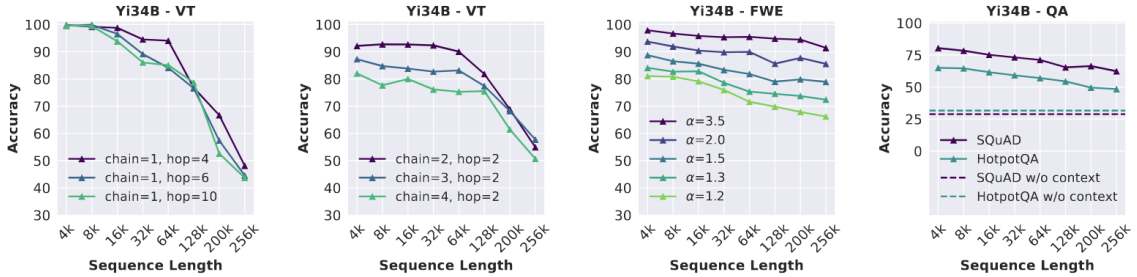


Figure 3: Performance of Yi-34B in variable tracking (VT), frequent words extraction (FWE), and QA tasks across different task complexities. Yi shows large degradation and distinct trends with scaled context size in these non-retrieval tasks, demonstrating the need to evaluate behavior beyond retrieval from context.

Non-robustness to “needle” types.
Failure to ignore distractors.
Return incomplete information.

Unreliable tracking within context.
Failure to accurately aggregate.
Frequent hallucination in long-context QA

Model analysis

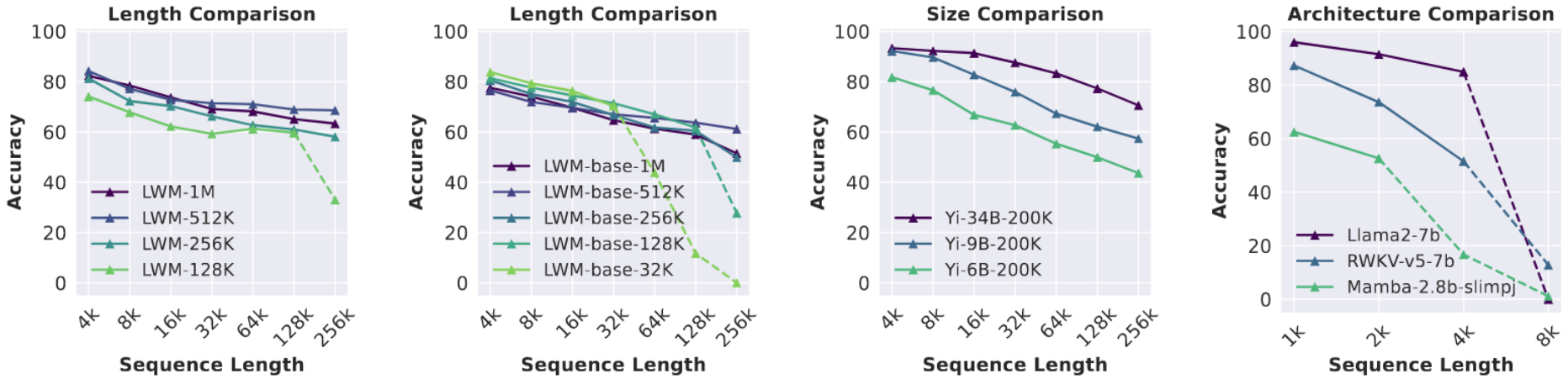


Figure 4: (**Left & middle left**): Comparison of LargeWorldModel (LWM) series trained up to various context sizes with fixed parameter size of 7B. (**Middle right**): Comparison of Yi suite models with different parameter sizes with controlled training context length of 200K. (**Right**): Performance of non-Transformer architectures lags behind the Transformer baseline Llama2-7B by large margin. Length extrapolation is presented with dashed lines.

Limitation

- Lack of position controlling
- Lack of correlation with realistic long-context tasks
- Lack of evaluation on short context.
- Lack of verification of prompt robustness.

Conclusion — Three Takeaways

- **RULER reveals the gap between *reading* and *understanding*.**
  Many models can technically process long inputs but fail to *use* them effectively for reasoning or synthesis.
- **"Readable length" ≠ "Usable length."**
  As context length grows, models often lose focus, suffer from attention dilution, and forget critical information.
- **Longer windows alone do not solve the problem.**
  Architectural and training innovations — such as memory compression, recurrent reasoning, and retrieval-augmented attention — are needed to truly *use* long contexts.

The End