Orthros

Phase-Aware Heterogeneous Attention for Efficient Transformers

Paper: Yuhong Chou, Zehao Liu, Yuqi Pan, Qian Liu,
Xianwei Chen, Zhongzhu Zhou, and Jibin Wu

Reading seminar | arXiv:2610.13735v1
12 November 2026

Build memory locally; read it globally only when predicting

Core idea

A base stream builds full layerwise KV with sliding-window attention. Output-specific tokens query that KV globally.

- Prefill: one global query per layer, at the last prompt position.
- Decode: a base/output token pair per generated step.
- Strongest gains occur with long inputs and short outputs.

arXiv v1, 9 October 2026. Accepted at NeurIPS 2026 per coauthor page and arXiv comments. Official code not found as of 12 November 2026; third-party core-mechanism implementation.

Paper, Abstract and Secs. 1/3, PDF pp. 1–7. Third-party implementation is a diagnostic, not full experimental reproduction.

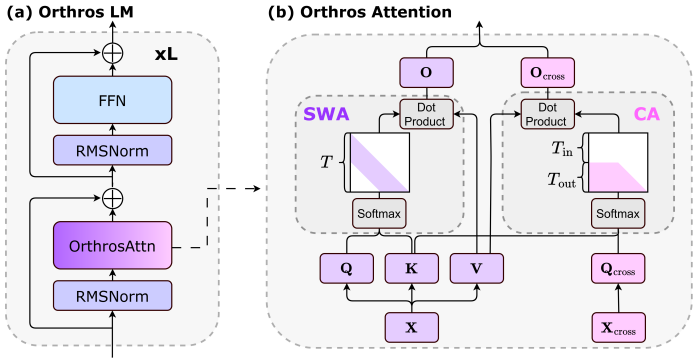
Phase roles offer another way to allocate expensive attention

Full attention	Global reads at every prompt position produce quadratic prefill work.
Efficient attention	Local windows or compressed states reduce work but can impair detailed recall.
Fixed layer hybrid	Retains some global layers, so a quadratic term remains.
Orthros	Keeps full historical KV; reduces the number of global queries during prefill.

The design targets context assimilation versus output prediction, within each layer.

Paper, Secs. 1–2, PDF pp. 1–4. Fixed-width/window asymptotic comparison.

The streams share weights and same-layer KV



- Base X provides Q, K, V; output X_{cross} provides global queries.
- Both streams share output projection, normalization, and tokenwise FFNs.

Paper, Fig. 1(a)–(b), PDF p. 5; Algorithms 1–2, PDF p. 6. SWA: local; CA: full causal cross-attention.

Lightweight query transforms specialize two shared paths

$$k_t = x_t W_K, \qquad v_t = x_t W_V$$
$$q_t = \alpha \odot (x_t W_Q) + \beta, \qquad q'_t = \alpha' \odot (x'_t W_Q) + \beta'$$

- $x_t, x'_t \in \mathbb{R}^{1 \times d_{\text{model}}}$; $q, k, v \in \mathbb{R}^{1 \times d}$ in the single-head notation.
- Shared large projections; separate learnable scale-offset vectors.
- Only the base stream generates K/V. The output stream does not write cache entries.
- T_{out} counts prediction positions, starting at the final prompt token.

Paper, Sec. 3.1, PDF p. 4; Algorithm 1, PDF p. 6. Prime denotes output-specific representations.

The streams differ in their causal attention domains

$$\mathcal{W}_t = \{\max(1, t - w + 1), \ldots, t\}$$

$$o_t = \mathrm{softmax}_{j \in \mathcal{W}_t}\left(\frac{q_t k_j^\top}{\sqrt{d}}\right) V_{\mathcal{W}_t}$$

$$o'_t = \mathrm{softmax}_{j \le t}\left(\frac{q'_t k_j^\top}{\sqrt{d}}\right) V_{1:t}$$

- Local output for all base positions; global output only where predictions are required.
- Default base window $w = 128$; old KV remains available to global queries.
- Final logits and language-model loss use the output stream.

Paper, Eqs. (4)–(5), PDF p. 5; Algorithms 1–2, PDF p. 6. All reads remain causal.

Output states never affect the memory-building path

Consistency invariant

Base states and their KV depend only on base states. Skipping unused output copies cannot change the stored KV.

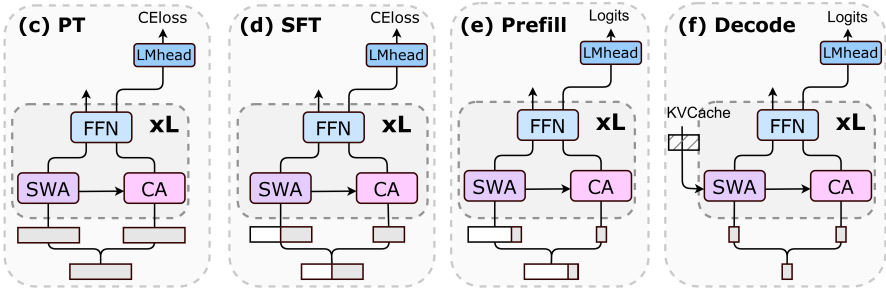
$$[A, B, C] \xrightarrow{\text{base path}} \{K_A, K_B, K_C; V_A, V_B, V_C\}$$

$$C' \xrightarrow{\text{global read}} D, \qquad (D, D') \xrightarrow{\text{next step}} E$$

- The prompt needs only C' to predict the first output D.
- At the next step, base D updates cache and D' predicts E.
- This explains how training can use many output copies while prefill uses one.

Teaching example derived from Paper Secs. 3.1–3.2, PDF pp. 4–7; not an original experiment.

One computation graph supports four processing regimes



- PT: duplicate every position. SFT: duplicate response-prediction positions.
- Prefill: duplicate the last prompt token. Decode: one new token and its copy.
- Duplicated embeddings diverge into two representations through the layers.

Paper, Fig. 1(c)–(f), PDF p. 5; Sec. 3.2, PDF pp. 6–7. Decode also reads historical KV.

Prefill is linear when only one output query is needed

$$C = \mathcal{O}(TwD) + \mathcal{O}(T_{\text{out}}TD) + \mathcal{O}((T + T_{\text{out}})D^2)$$

- T: base length; D: model width; w: fixed window.
- Prefill, $T_{\text{out}} = 1$: linear in T at fixed width and window.
- $T_{\text{out}} = o(T)$: subquadratic attention; $T_{\text{out}} = \Theta(T)$: quadratic.
- Pretraining duplicates all tokens and retains a quadratic global-read term.

Paper, Sec. 3.3, PDF p. 7. A small fixed output/input ratio alone does not imply subquadratic asymptotics.

The savings concern query work, while KV stays complete

- Every layer retains all historical base K/V for output retrieval.
- Local SWA reads do not imply KV eviction.
- PT and decode process extra output-specific states through shared FFNs and projections.
- At low batch, shared weight reads may hide some extra arithmetic; behavior changes with the serving regime.

Trade-off

More training/output-path computation in exchange for cheaper input processing and global history access.

Paper, Secs. 3.3 and 5, PDF pp. 7, 12. No fixed-size cache compression is introduced.

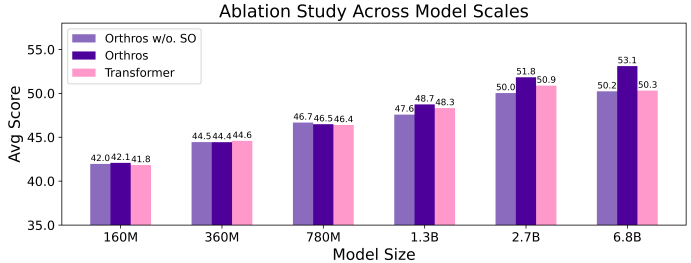
Separate the quality experiments and their token budgets

Experiment	Scale	Budget / protocol
Scaling + SO ablation	160M–6.8B	10B tokens; length 2048
From scratch	1.3B	100B FineWeb-Edu tokens
Conversion	Llama3.1/Qwen3-8B	80B tokens; length 32768
128K NIAH	Converted Qwen3-8B	Additional 20B ProLong tokens

- Orthros window: 128. The scratch-training SWA baseline uses 512.
- Equal original-token counts do not equalize training FLOPs across the double-stream design.

Paper, Secs. 4.1–4.4, PDF pp. 8–10; App. A, PDF pp. 19–20.

Scale-offset gains are strongest in the larger tested models



- All scales see 10B tokens; the appendix reports six-task averages.
- At 6.8B: SO 53.10, w/o SO 50.23, Transformer 50.31.
- At 360M/780M, SO does not beat w/o SO; the improvement is not uniform.

Paper, Fig. 2, PDF p. 8; precise values in Table 7, PDF p. 20. These are not the seven-task 100B-token scores.

Scratch training preserves competitive common-sense performance

Model	Transformer	Orthros	GDN	GLA	RetNet	SWA
Avg. ↑	53.41	53.98	53.29	51.90	51.96	53.22

- 1.3B, 100B tokens; seven common-sense tasks.
- Orthros gains 0.57 percentage points in the mean.
- OBQA improves: 40.20 vs. 37.80; PIQA is lower: 71.82 vs. 72.91.
- This supports model quality at matched data, with extra training computation.

Paper, Table 1 and Sec. 4.2, PDF p. 9. Scores follow each table column's accuracy/normalized-accuracy convention.

Conversion should be compared with continued pretraining

Eight-task average ↑	Base	Base CPT	Orthros Conversion
Llama3.1-8B	65.47	67.00	67.26
Qwen3-8B	67.87	67.55	68.37

- CPT and conversion use the same 80B original tokens.
- Conversion gains vs. CPT: +0.26 and +0.82 percentage points.
- MMLU is slightly below CPT for both families.
- Token-matched controls isolate extra data, but not extra compute from token duplication.

Paper, Table 2, PDF p. 9; training details in Table 6, PDF p. 20. Conversion requires training, not just inference-time replacement.

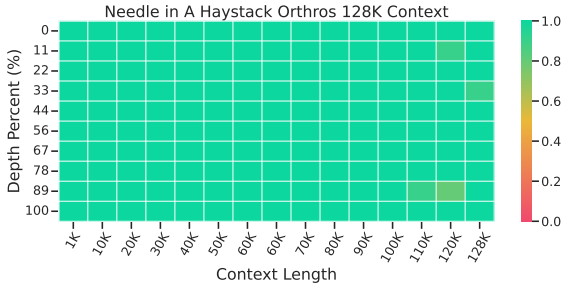
Global queries recover much of the recall lost by efficient baselines

Model	Transformer	Orthros	GDN	GLA	RetNet	SWA
Avg. ↑	33.9	34.4	28.4	27.1	25.7	28.8

- Six recall-intensive tasks, 1.3B scratch-trained models.
- FDA: 24.1 vs. 16.4; SWDE: 39.9 vs. 36.7, against Transformer.
- SQuAD is lower: 38.7 vs. 42.6; the average hides task differences.
- NQ denotes NarrativeQA here, not Natural Questions.

Paper, Table 3 and Sec. 4.4, PDF p. 10. Appendix A mentions 8B inconsistently; the main text and table specify 1.3B.

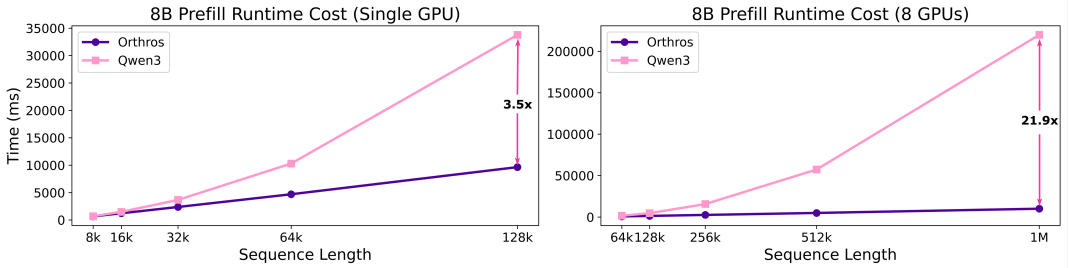
The converted model retrieves needles across 128K context



- Horizontal: context length. Vertical: insertion depth. Color: accuracy.
- Qwen3-8B conversion, followed by 20B tokens of long-context training.
- Near-perfect needle retrieval supports long-range access in this test.

Paper, Fig. 3 and Sec. 4.4, PDF p. 10. This differs from the 1.3B recall-task comparison.

The 21.9x headline is an eight-GPU prefill result



- Left: one GPU, up to 128K; 3.5× marked at 128K.
- Right: eight GPUs with sequence parallelism, up to 1M; 21.9× marked at 1M.
- FLA-style kernels; vertical axes in milliseconds. The serving experiment uses a separate setup.

Paper, Fig. 4 and Sec. 4.5, PDF p. 11; App. A, PDF p. 19. Both models are 8B-scale.

End-to-end gains shrink as output generation grows

Input / output	Transformer E2E (seconds)	Orthros E2E (seconds)	TTFT gain	E2E gain
128K / 128	84.911	24.647	4.54×	3.45×
127K / 1K	137.715	76.345	4.40×	1.80×
120K / 8K	548.357	488.830	4.46×	1.12×

- Single H100, SGLang, Triton attention, batch size 2.
- First setup: TTFT drops from 77.249 s to 17.016 s.
- Similar prefill savings contribute less once decoding dominates total time.

Paper, Table 4, PDF p. 11. Gains calculated from reported values; input/output lengths follow the table headers.

Decode throughput stays comparable in the tested small batch

Input / output	Output throughput		Decode-only throughput	
	Transformer	Orthros	Transformer	Orthros
128K / 128	3.0	10.4	33.7	33.3
127K / 1K	14.9	26.8	33.2	33.7
120K / 8K	29.9	33.5	34.1	34.6

- All rates in tokens/s, aggregated over batch=2.
- Output throughput includes prefill; decode-only throughput excludes it.
- Shared weight loads may hide extra arithmetic at small batch; larger-batch behavior is unestablished.

Paper, Table 4, PDF p. 11; system interpretation in Secs. 3.3/5, PDF pp. 7, 12.

Phase heterogeneity can also be combined with layer hybrids

Input / output	Linear Hybrid E2E	Orthros-H E2E	Gain
128K / 128	28.130 s	12.771 s	2.20×
127K / 1K	46.308 s	30.724 s	1.51×
120K / 8K	189.302 s	173.887 s	1.09×

- Orthros-H uses a 1:3 mixture of full/linear layers, with a GDN variant.
- Compare it within the hybrid pair; standard Orthros retains full global reads during decoding.
- This table reports speed; full matching quality results for Orthros-H are not supplied.

Paper, Sec. 4.6 and Table 4, PDF p. 11. Single H100, SGLang, batch=2; gains calculated from reported values.

The design trades training work for cheaper input processing

- **Training:** PT duplicates all token states; long output/training attention remains quadratic.
- **Memory:** full historical KV grows with context length.
- **MoE:** split token pairs can trigger different expert reads and break I/O sharing.
- **Deployment:** batch size, KV bandwidth, and output length change the bottleneck.

The target workload is prefill-heavy serving with strong global recall requirements.

Paper, Sec. 5, PDF p. 12. Agentic RL and broader serving configurations are future work.

A dependency separation makes the efficiency mechanism possible

1. Build memory through a local base stream at every layer.
2. Query the full history through output-specific states only where needed.
3. Keep output states out of the memory-building dependency chain.
4. Measure quality, training compute, cache cost, and serving latency together.

Global access is preserved; the number of prefill global queries is reduced.

Interpretation of Paper Secs. 3–5, PDF pp. 4–12.

References and implementation status

- Chou, Liu, Pan, Liu, Chen, Zhou, and Wu.
  Orthros: Phase-Aware Heterogeneous Attention for Efficient Transformers.
  arXiv:2610.13735v1, 9 October 2026.
- NeurIPS 2026 acceptance per coauthor publication list and arXiv comments.
- Official code not found as of 12 November 2026.
  Third-party core mechanism / scope description.

Figures come from the authors' v1 source/PDF. Calculated ratios are labeled. This deck is a reading seminar, not an author presentation.

Publication and code checked 12 November 2026. Formal proceedings publication was not independently confirmed.

Backup: duplicated positions keep their original causal meaning

$$Z = \mathrm{Concat}[X_{1:T}, X_{T-T_{\mathrm{out}}+1:T}]$$

- Appending copies is a tensor layout choice; each output copy keeps its original prediction position.
- For a query at original position t, Eq. (5) allows only keys $j \leq t$.
- Prefill copies the final prompt token, which predicts the first response token.
- At decode, current input length is one, but global key length includes all historical base tokens.

The causal mask must respect these positions rather than the physical appended indices.

Paper, Eqs. (4)–(5), PDF p. 5; Algorithms 1–2 and Sec. 3.2, PDF pp. 6–7. Implementation implication derived from the equations.

Backup: long generation restores a quadratic read term

For prompt length P and response length R:

$$\sum_{r=1}^{R}(P+r) = RP + \frac{R(R+1)}{2}$$

- This counts historical keys read by successive global output queries, up to endpoint conventions.
- Prefill adds a final-prompt global query of length about P.
- Fixed R: cost grows linearly with P at fixed model/window size.
- Large R: the response's own quadratic global-read term remains.

Teaching expansion of Paper Sec. 3.3, PDF p. 7; endpoint constants omitted. No timing claim is inferred from FLOPs alone.

Backup: training protocols need distinct budgets

	Scaling / ablation	8B conversion / CPT
Original tokens	10B	80B
Sequence length	2048	32768
Global token batch	0.25M	8M
Steps	40K	10K
Learning rate	3×10^{-4}	3×10^{-5}
Warmup steps	1500	500
Weight decay	.05	.1
Seed	42	42

The 1.3B scratch comparison separately specifies 100B tokens and a 2M-token batch. All use H100 80GB training hardware.

Paper, Tables 5–6, PDF pp. 19–20; Sec. 4.2, PDF p. 9. Original tokens exclude duplicated token-state processing.

Backup: precise scale-offset results do not form a uniform ranking

Size	Transformer	Orthros w/o SO	Orthros w/ SO
160M	41.83	41.96	42.08
360M	44.58	44.45	44.44
780M	46.39	46.67	46.48
1.3B	48.32	47.58	48.74
2.7B	50.87	50.03	51.83
6.8B	50.31	50.23	53.10

- Six-task averages at 10B tokens; Social-IQa is absent from this table.
- Table 1 uses seven tasks at 100B tokens. Keep these protocols separate.

Paper, Table 7, PDF p. 20, and Table 1, PDF p. 9. Metrics follow each table's column labels.