LoLCATs: On Low-Rank Linearizing of Large Language Models

Paper discussion: transfer local functions, then adapt the model

Paper by Michael Zhang, Simran Arora, Rahul Chalamala, Alan Wu,
Benjamin Spector, Aaryan Singhal, Krithik Ramesh, Christopher Ré

arXiv:2410.10254v3 — 5 March 2025 (ICLR 2025)
Reading prepared 28 September 2026 | 25-minute main talk + backup

Low-cost conversion is feasible; capability preservation remains selective.

Independent reading presentation. Source: `https://arxiv.org/abs/2410.10254v3`.

The problem is the cost of converting a strong LLM

What we already have

A pretrained Transformer with useful knowledge and language abilities.

- Full attention: growing KV cache.
- Linear attention: fixed-size recurrent state.
- A direct swap can break model quality.

What LoLCATs asks

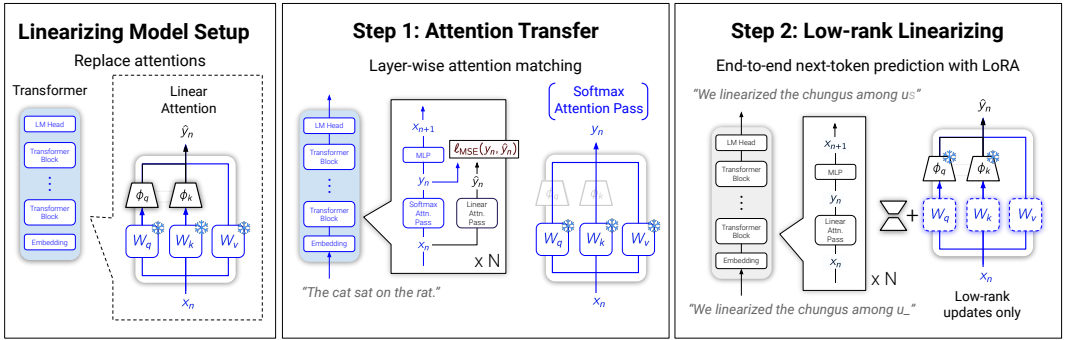
Can a close local approximation make cheap global adaptation sufficient?

- Preserve useful pretrained computation.
- Train few additional parameters.
- Scale conversion up to 405B.

Conversion token counts exclude the original model's pretraining investment.

Paper, Secs. 1–2, PDF pp. 1–4.

First transfer attention; then adapt the whole network



Two different training targets

Stage 1: attention output MSE. Stage 2: next-token cross-entropy.

Original Fig. 1, PDF p. 2; Sec. 3.1, pp. 4–5. Frozen pretrained weights are reused.

Linear attention compresses history into a recurrent state

For one head, $q_n, k_n, v_n \in \mathbb{R}^d$ and $\phi_q, \phi_k : \mathbb{R}^d \to \mathbb{R}^F$:

$$S_n = S_{n-1} + \phi_k(k_n) v_n^\top, \qquad z_n = z_{n-1} + \phi_k(k_n)$$

$$\boxed{\hat{y}_n = \frac{\phi_q(q_n)^\top S_n}{\phi_q(q_n)^\top z_n}}$$

- $S_n \in \mathbb{R}^{F \times d}$ stores key–value associations; $z_n \in \mathbb{R}^F$ normalizes.
- Fixed F, d: $O(Fd)$ attention work per generated token.
- Attention state is independent of history length; model weights remain.

Paper, Eqs. (2)–(3), PDF p. 4. F denotes the final feature dimension in this presentation.

Output matching calibrates layers on teacher inputs

$$\mathcal{L}_{\mathrm{AT}} = \mathrm{mean}_{\text{batch, layer, head, token, channel}} \left(y - \hat{y}\right)^2$$

- Freeze pretrained projections; train feature maps and mixing terms.
- Compute teacher and student attention from the same layer input.
- Pass teacher outputs to subsequent layers during transfer.
- Match $N \times d$ outputs, without materializing all $N \times N$ weights.

What output matching supervises

It encourages functional agreement on sampled inputs. Matching AV does not uniquely identify the attention matrix A.

Paper, Sec. 3.1 and Eq. (5), PDF pp. 4–5. Explicit mean is a clarified notation. Teacher attention still costs compute.

LoRA adjusts the converted model's own computation

$$W' = W + \frac{\alpha}{r} BA, \qquad \mathcal{L}_{\mathrm{LM}} = -\sum_{t} \log P_{\Theta'}(u_{t+1} \mid u_{1:t})$$

- Add LoRA to W_q, W_k, W_v, W_o; freeze original weights and feature maps.
- Default 7B/8B recipe: rank $r = 8$, $\alpha = 16$, dropout 0.
- The student now supplies all hidden states: train against accumulated error.

"Low-rank" refers to the update

The full pretrained matrices remain. This is parameter-efficient adaptation, not a low-rank compression of the entire LLM.

Paper, Sec. 3.1, PDF p. 5; App. A.1 and B.4.1, pp. 18–19, 23–25.

Neither transfer nor LoRA alone is sufficient here

Llama 3 8B, pure linear attention	T2R	Hedgehog
Random swap, before adjustment	1539.39	2448.01
Transfer only, before LoRA	79.33	60.86
No transfer + LoRA	16.05	9.02
Transfer + LoRA	4.11	3.90

- Alpaca validation perplexity ↓; final rows use equal total steps.
- Transfer + LoRA: 2 + 2 epochs; no transfer: 4 LoRA epochs.
- Pure linear attention still struggles on downstream MMLU.

Selected results from Table 2, PDF p. 6. Pure Hedgehog MMLU: Table 3, p. 9.

Keep local softmax; compress only the older history

Let $\mathcal{W}_n$ be the recent window, and c_n its maximum attention logit.

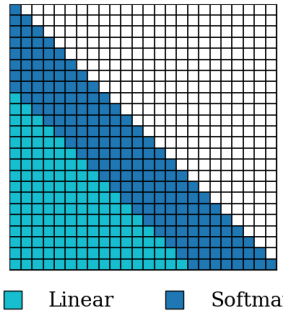
$$U_n = \sum_{i \in \mathcal{W}_n} \gamma e^{q_n^\top k_i / \sqrt{d} - c_n} v_i, \quad D_n = \sum_{i \in \mathcal{W}_n} \gamma e^{q_n^\top k_i / \sqrt{d} - c_n}$$

$$\boxed{\hat{y}_n = \frac{U_n + \phi_q(q_n)^\top S_{n-w}}{D_n + \phi_q(q_n)^\top z_{n-w}}}$$

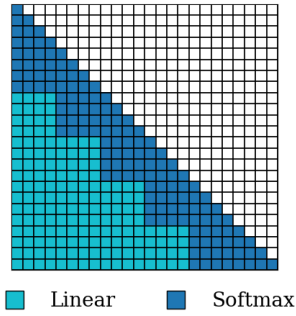
- γ: learned mixing term; S, z: history outside the local window.
- Combine unnormalized contributions, then normalize jointly.
- Evicted local KV entries enter the recurrent state.

Paper, Eq. (6), PDF p. 7, reorganized for presentation. Empty historical state is zero.

The deployed recipe uses a GPU-friendly terraced window



Standard sliding window



TK terraced window

Rows: queries
Columns: keys

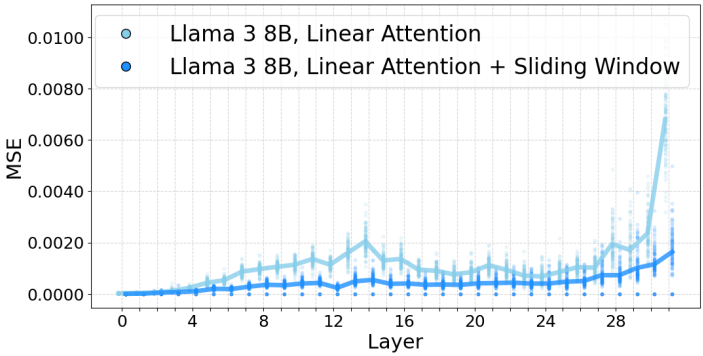
Blue: local softmax
Cyan: linear attention

Default tile size: 64. Local coverage varies with query position.

Apply RoPE *before* Hedgehog features. A 64-wide projection produces 128 features after positive/negative softmax concatenation.

Original Fig. 9, PDF p. 18; Table 1, p. 5; Table 7, p. 19.

A local window reduces the hardest attention errors



- Horizontal axis: layer; vertical axis: attention output MSE ↓.
- The local + linear layer reduces late-layer error substantially.
- Lower MSE correlates with quality; it is not a universal task guarantee.

Original Fig. 7, right panel, PDF p. 8; related entropy analysis: Figs. 5–6, pp. 7–8.

The small-model recipe needs little additional training

7B / 8B setup

- Base Mistral 7B / Llama 3 8B.
- About 50K cleaned Alpaca samples.
- Packed length 1024; global batch 8.
- AdamW; BF16; terrace size 64.

Training budget

- Transfer LR: 10^{-2}.
- LoRA LR: 10^{-4}; rank 8.
- Reported conversion tokens: 40M.
- One 40GB A100: about 6.5 hours.

Budget caveat: Table 7's epoch/token entries conflict with the 40M total stated elsewhere. Exact reproduction requires clarification.

Paper, Sec. 3.2, p. 6; App. A.1, pp. 18–19; App. B.5.1, p. 26.

Zero-shot averages recover; MMLU retains a large gap

Model	5-task avg.	MMLU	6-task avg.
Mistral 7B	74.4	62.4	72.4
SUPRA (100B conversion tokens)	69.9	34.2	64.0
LoLCATs (40M reported)	74.5	51.4	70.7
Llama 3 8B	73.1	66.6	72.0
Mamba2-Llama 3 (20B tokens)	65.6	43.2	61.9
LoLCATs (40M reported)	74.2	52.8	70.7

MMLU gains over these conversion baselines: +17.2 and +9.6 points.
Remaining MMLU gaps to the original models: 11.0 and 13.8 points.

Selected columns of Table 3, PDF p. 9. Scores in percent, higher is better. MMLU is 5-shot; other tasks are zero-shot.

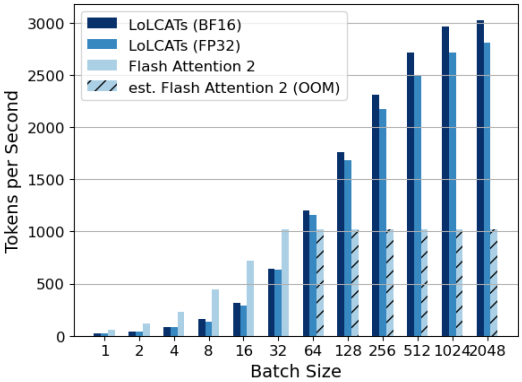
Transfer and the local window work best together

Setting	Hedgehog Avg.	Hedgehog MMLU	T2R Avg.	T2R MMLU
Swap + adjust	44.20	23.80	38.84	23.20
+ transfer	55.32	23.80	54.83	23.10
+ window only	68.78	45.80	39.52	23.80
+ transfer + window	70.66	52.77	68.28	40.70

- Pure linear transfer helps the average, but barely helps MMLU.
- A window alone is insufficient for T2R under this protocol.
- Three-seed LoLCATs MMLU: 52.74 ± 0.64 (mean $\pm$ SD).

Table 5, PDF p. 11; Table 11, p. 22. Llama 3 8B; Avg. includes all six tasks.

Higher feasible batch sizes unlock throughput gains



Llama 3 8B, one 80GB H100.
128-token prompt; 4096 generated tokens.

- FA2 OOM at batch 64.
- LoLCATs: batch 2048, about 3000 tokens/s.
- Hatched FA2 bars are estimates beyond OOM.
- FA2 is faster at small shared batch sizes.

Original Fig. 8, left panel, and Sec. 4.2, PDF p. 10. About 3x peak throughput is not a single-request latency claim.

Attention transfer improves 70B and 405B MMLU

Llama 3.1	Original	No transfer	LoLCATs	Gap closed
8B	66.11	51.44	54.88	23.4%
70B	78.80	28.74	67.70	77.8%
405B	82.98	33.86	72.20	78.1%

$$\text{Gap closed} = \frac{\text{LoLCATs} - \text{No transfer}}{\text{Original} - \text{No transfer}}$$

- MMLU 5-shot accuracy (%). No-transfer baseline uses the same hybrid layer.
- Remaining original-model gaps: 11.10 points (70B), 10.78 (405B).

Table 6, PDF p. 11; baseline protocols in App. A.2–A.3, pp. 18–21. Gap percentages calculated from table values.

Block-wise transfer helps at 405B, with a storage cost

405B validation perplexity	
Joint transfer, 126 layers:	**4.23**
14 blocks of 9 layers:	**3.21**

- Both evaluated after LoRA adjustment.
- Joint and block-wise training are similar at 8B / 70B.

Teacher forcing permits precomputed block inputs.

$$\text{Disk bytes} \approx 2TD\frac{M}{b}$$

T: tokens; D: model width; M: layers; b: layers per block; 16-bit states.

More blocks enable independent training, but require more stored hidden states.

Sec. 3.3.2, PDF pp. 8–9; Table 24, p. 35. Disk storage is a training cost, distinct from inference KV memory.

The 405B run adds attention-map supervision

- Stage 1: 1000× output MSE + 1× attention-map CE.
- Stage 2: LoRA rank 4, alpha 8, dropout 0.5; FP16.
- Data: 20M RedPajama tokens per stage; packed length 1024.
- Transfer: 14 blocks, about 5 hours on one H100 per block.
- LoRA: 16 hours on 24 H100s; about 454 GPU-hours in total.

Interpretation

The 405B result does not establish that output MSE alone suffices at every scale. The baseline also has a different allocation of training stages.

App. A.3 and Table 9, PDF pp. 20–21. GPU-hours derived as $14 \times 5 + 24 \times 16$; preparation/I/O not separately quantified.

Retrieval is not automatically preserved by conversion

8K passkey retrieval, Llama 3 8B	Exact match
LoLCATs, packed Alpaca conversion data	**0% in every position bin**
LoLCATs, dedicated passkey conversion data	100% in every position bin

- Dedicated recipe: 10K examples, approximately 8192 tokens each.
- Task-trained LoLCATs still fails at 10240 tokens in the reported plot.
- This demonstrates learnability of a narrow task, not default long-context retention.

Table 21, PDF p. 28; Fig. 10, p. 29. Data length and amount differ from the default conversion recipe.

Data choices trade general scores for targeted abilities

Llama 3 8B conversion data	5-task avg.	MMLU
Alpaca	74.2	52.8
Random RedPajama	72.1	55.2
RedPajama, high effective sequence length	70.9	56.5

- Packing short samples does not create genuine long-range dependencies.
- Adding multiple-choice CQA to Alpaca: MMLU 52.8 $\rightarrow$ 54.5.
- CQA alone: 43.9. Targeted format alone is insufficient.

Tables 19–20, PDF p. 27; Table 22, p. 30. Effective sequence length is an attention-distance statistic.

The evidence supports efficient conversion within limits

1. **Quality:** near-recovery on five common tasks; persistent MMLU gaps and data-sensitive retrieval.
2. **Compute:** small additional training cost; original pretraining is inherited.
3. **Inference:** strong batch-capacity benefits; speed depends on batch, workload, and implementation.
4. **Scale:** 405B conversion works with block training and an extended distillation objective.

Synthesis of Tables 3, 6, 9, 21; Fig. 8; paper limitations, PDF pp. 11–12.

Transfer the function before adapting the network

Main lesson

Reducing the architectural mismatch locally can make a small global parameter update surprisingly effective.

- Local softmax + recurrent history is easier to transfer than pure linear attention.
- Conversion data should exercise the capabilities we need to retain.
- Judge the converted model on quality, memory, and the actual serving workload.

Discussion: Can normalized layer losses reduce the need for cached block inputs at 405B?

Reader synthesis and open question. Full paper: `https://arxiv.org/abs/2410.10254v3`. Main talk ends here.

Backup: evaluation metrics and comparison roles

- PiQA, ARC-easy, Winogrande: zero-shot accuracy.
- ARC-challenge, HellaSwag: zero-shot length-normalized accuracy.
- MMLU: 5-shot accuracy. Six-task average includes it; five-task average excludes it.
- Prior conversion numbers are partly cited from original papers; not all uniformly rerun.
- Comparisons to models pretrained from scratch inherit different data and training investments.

405B small improvements over the original model on non-MMLU tasks lack reported multi-seed uncertainty.

Tables 3–6, PDF pp. 9–11; multi-seed evidence is limited to Table 11, p. 22.

Backup: small LoRA updates can already be enough

Rank	5-task avg.	MMLU
4	74.5	50.1
8	74.2	52.8
128	73.0	53.4
256	72.7	52.1
Full FT	73.7	52.1

Five-task average, rank 8:

- W_q only: 72.68
- W_k only: 72.29
- W_v only: 74.19
- W_o only: 74.09
- All projections: 74.24

Values and output projections recover much of the measured quality. Projection ablation does not report MMLU.

Tables 16–17, PDF pp. 25–26. Full FT uses a different LR. Inconsistent six-task averages are not reproduced.

Backup: full-attention hybrids recover more MMLU

Llama 3 8B architecture	Conversion tokens	MMLU
Original full attention	0	66.6
First 16 layers remain full attention	20M	65.8
All layers converted	40M reported	52.8

- The 50% hybrid uses attention transfer only, without Stage 2 LoRA.
- Remaining full-attention layers still require a growing KV cache.
- At 1B scale: LoLCATs also nearly recovers five-task averages, while MMLU declines (Llama 3.2 1B: 31.9 $\rightarrow$ 27.3).

Tables 12–15, PDF pp. 23–24.

Backup: the first transfer steps have high value

Transfer steps	0	200	1800
Alpaca validation PPL ↓	3.211	2.802	2.776
RedPajama validation PPL ↓	61.305	10.459	9.358
Five-task average ↑	56.86	73.68	73.66

- Each run uses 2000 subsequent LoRA steps.
- More transfer can improve PPL without monotonic task-score gains.
- A separate larger-RedPajama study also finds this distinction.

Table 23 and Figs. 12–13, PDF p. 31. Llama 3 8B.

Backup: training dynamics are suggestive, not causal proof

- Fig. 14: LoRA can further reduce attention-output MSE, especially where the initial mismatch is large.
- Diagnostic exception: those LoRA updates optimize MSE directly, not the default next-token objective.
- Figs. 15–17: with next-token training, transferred models' value/output updates look more like the softmax model's updates.
- Selected attention heatmaps show recovery of some distant interactions, but do not prove exact attention-weight recovery.

App. B.7.1–B.7.2, PDF pp. 32–34; App. E.2, pp. 50–58 (examples inspected selectively).

Backup: the prefill kernel exploits GPU locality

1. Load 64-token query/key/value tiles.
2. One warpgroup handles local attention and memory transfers; another handles linear attention and state updates.
3. Use tensor-core operations and keep recurrent state in registers.
4. Combine branch contributions, normalize, and write outputs.

This is a prefill implementation. Algorithmic complexity alone does not establish end-to-end speed.

App. C.2, Algorithm 3, PDF pp. 43–44. Schematic summary; not executable kernel code.

Backup: several paper details need careful reproduction

- Token budget: Table 7 implies 60M by its epoch entries; the main text reports 40M.
- MMLU relative gains in Sec. 4.1 do not match Table 3. Use the verified +17.2 / +9.6 percentage-point gains.
- Some appendix averages disagree with the listed task scores.
- MSE, block-loss, and ESL equations contain index ambiguities.
- Appendix C code is illustrative and includes incomplete or erroneous details; it should not be executed verbatim.

PDF pp. 5, 8–9, 19, 25–26, 29, 37–42. Detailed checks and adopted interpretations are in the Chinese notes.

Backup: source material and a practical replication order

Michael Zhang et al. LoLCATs: On Low-Rank Linearizing of Large Language Models. arXiv:2410.10254v3, 2025.

`https://arxiv.org/abs/2410.10254v3`
Author-linked code: `https://github.com/HazyResearch/lolcats`

1. Fix the model, tokenizer, attention variant, and evaluation settings.
2. Reproduce the two-stage ablation before scaling up.
3. Test MMLU, held-out PPL, and retrieval separately.
4. Audit realized token counts, masks, causal state updates, and LoRA scaling.

This reading uses the matching author TeX and PDF. No model training or current repository audit was performed.