MLP Memory: learn retrieval before deployment

A Retriever-Pretrained Memory for Large Language Models

Rubin Wei, Jiaqi Cao, Jiarui Wang, Jushi Kai, Qipeng Guo, Bowen Zhou, Zhouhan Lin

arXiv:2508.01832v4 · Research group discussion · 20–30 min

Core idea

Distill token-level retrieval distributions into a separate MLP, then mix its probabilities with a frozen language model.

Published at ICLR 2026. Official code and checkpoints available.

Status checked 2026-11-11. This presentation analyzes arXiv v4.

Knowledge access has deployment trade-offs

Approach	Knowledge access	Main cost
RAG	Retrieved documents	Search + longer context
kNN-LM	Retrieved token labels	Datastore + token search
CPT / LoRA	Adapted model weights	Potential task degradation
MLP Memory	Separate parametric branch	Offline distillation + extra MLP

- Move retrieval work into training; retain a direct forward pass at inference.
- Long-term memory here means corpus knowledge stored in parameters.

Sec. 1–2, PDF pp. 2–4.

The teacher is a token-level retriever

$$(\mathcal{K}, \mathcal{V}) = \{(f(c_t), w_t) : (c_t, w_t) \in \mathcal{D}\}$$

$$p_{kNN}(y \mid c) \propto \sum_{(k_i, v_i) \in \mathcal{N}} \mathbb{I}[y = v_i] \exp[-d(k_i, f(c))]$$

- c_t: prefix; w_t: next token; $f(c_t) \in \mathbb{R}^d$: context vector.
- Retrieve nearby context vectors, then aggregate their next-token labels.
- Sparse vocabulary distribution supplies alternatives beyond one observed token.

Sec. 2, Eqs. (2)–(3), PDF p. 4. Normalization is implicit in the paper.

A separate MLP replaces search at inference

Frozen backbone: encode the prefix

$$c_t \longrightarrow \text{Transformer} \longrightarrow (h_t,\ p_{LM})$$

Memory branch: predict retrieval-like probabilities

$$h_t \longrightarrow \text{stacked MLPs} \longrightarrow p_{MLP}$$

$$p_{final} = \lambda p_{MLP} + (1 - \lambda) p_{LM}$$

Teaching schematic based on Fig. 4, PDF p. 5. We follow Eq. (7)'s memory-weight convention; Fig. 4 reverses it.

No token mixing is needed inside the memory

$$\mathcal{M} : \mathbb{R}^d \longrightarrow \mathbb{R}^{|V|}$$

- Each query vector already summarizes the prefix through the Transformer.
- Stack feed-forward, residual and normalization operations; output a vocabulary distribution.
- Llama2 / Mistral: 8 layers; hidden size 4096; intermediate size 11008 / 14336.
- Initialize from the corresponding backbone MLP modules. Roughly 1B parameters, excluding embeddings.

Sec. 3.1 and Appendix I, PDF pp. 5–6, 26–27; Table 11.

Build the retrieval targets once, offline

1. Run the backbone over the training corpus; store context vectors and next tokens.
2. Retrieve neighbors for each training context, excluding the query itself.
3. Cache vector–distribution pairs for memory training.
4. Optimize the memory branch; deploy without the datastore.

- Self-exclusion prevents trivial lookup of the training example.
- Deployment savings do not remove the offline indexing and search costs.

Sec. 3.2, PDF p. 6.

Hybrid supervision combines retrieval and truth

$$\mathcal{L}_{KL} = \mathrm{KL}(p_{kNN} \parallel p_{MLP}),$$
$$\mathcal{L}_{CE} = -\log p_{MLP}(w_t \mid c_t),$$
$$\mathcal{L} = \alpha\mathcal{L}_{KL} + (1-\alpha)\mathcal{L}_{CE}, \quad \alpha = 0.4.$$

- KL transfers the similarity-weighted distribution of plausible continuations.
- CE anchors the prediction to the observed next token.
- For a fixed teacher, this equals cross-entropy against $\alpha p_{kNN} + (1-\alpha)\delta_{w_t}$, up to a constant.

Sec. 3.2, Eqs. (4)–(6), PDF p. 6. Final bullet: teaching derivation.

Mix probabilities after softmax

$$p_{final}(w_t \mid c_t) = \lambda p_{MLP}(w_t \mid c_t) + (1 - \lambda) p_{LM}(w_t \mid c_t)$$

- λ weights memory at inference; α weights KL during training.
- Tune λ on each task's validation split. Probability mixing differs from logit averaging.
- Run the backbone plus the memory MLP; no document access or nearest-neighbor search.
- Differentiability permits joint optimization; main experiments train the components separately.

Sec. 3.3, Eq. (7), PDF p. 7; Fig. 4 reverses the weight convention.

Evaluation separates five questions

Question	Setting / metric
Scaling	GPT-2 + WikiText / Web; PPL $\downarrow$
Factual QA	Two 7B backbones; five benchmark scores $\uparrow$
General NLP	Mistral; nine classification / entailment tasks
Hallucination detection	HaluEval accuracy $\uparrow$
Design choices	Loss weight, input layer, teacher and capacity

- QA uses Wikipedia-2021; general NLP uses a separate mixed-domain corpus.
- EM for NQ, WebQA, TriviaQA, HotpotQA; MC1/MC2/MC3 average for TruthfulQA.

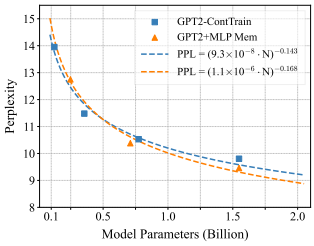
Sec. 4 and Appendix A, PDF pp. 7, 18.

Budgets and baselines shape the comparison

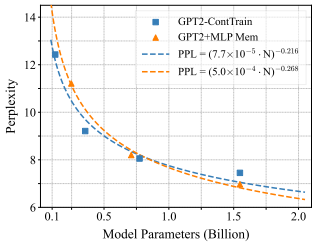
- 32 A800 80GB GPUs; AdamW; memory learning rate 4×10^{-4}; $\alpha = 0.4$.
- Paper states a training budget equivalent to one epoch of a 7B model. Offline retrieval costs are not itemized.
- RAG: BGE, top-5 documents. kNN-LM: $\lambda = 0.1$, temperature 10.
- LoRA: Q/K/V and MLP layers, rank chosen to match memory parameters. CPT updates all weights.
- Memory interpolation is tuned per task; the kNN baseline uses a fixed interpolation weight.

Sec. 4, PDF p. 7; Appendix A, Table 4, PDF p. 18.

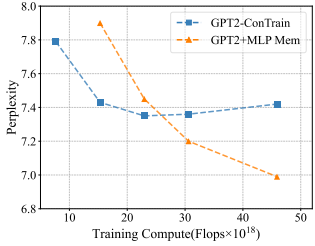
Scaling gains refer to fitted exponents



(a) Scaling law on WikiText-103



(b) Scaling law on Web



(c) Training compute Scaling Law

Dataset	$\lvert\gamma_d\rvert$	$\lvert\gamma_m\rvert$	Relative increase
WikiText-103	0.143	0.168	17.5%
Web	0.216	0.268	24.1%

$PPL = (\beta N)^{\gamma}$; N includes backbone and memory parameters.

Sec. 5.1, Fig. 5, Eqs. (8)–(9), PDF p. 8. These are not PPL reductions.

More compute continues to help the memory model

- WikiText-103: about 100M tokens. Web mixture: about 600M tokens.
- GPT-2 baselines span 124M–1.5B parameters; augmented models count both components.
- At the GPT2-xl scale, the Web compute curve keeps declining for the augmented model.
- Continued decoder training flattens and then rises in the observed range.
- Interpretation: retrieval imitation supplies a different learning signal; extrapolation beyond the measured scale remains open.

Sec. 5.1, Fig. 5(c), PDF pp. 8–9.

Average QA scores improve on both backbones

Method	Llama2-7B	Mistral-7B
Base	32.81	32.12
RAG	32.94	33.38
*k*NN-LM	33.16	32.57
CPT	29.66	30.53
LoRA	31.34	32.28
MLP Memory	**35.38**	**36.06**

- Relative increase of the average: 7.8% / 12.3%.
- Mistral gains 3.94 absolute points; this is not a pooled EM score.

Sec. 5.2, Table 1, PDF p. 9.

Explicit retrieval still wins on some QA tasks

Mistral task	Base	RAG	Memory
NQ	20.63	22.56	25.20
WebQA	29.28	24.90	37.45
TriviaQA	57.65	54.21	60.99
TruthfulQA	32.09	35.47	32.54
HotpotQA	20.96	29.77	24.14

- Largest memory gains over Base: WebQA +8.17; NQ +4.57.
- RAG leads Memory on TruthfulQA and HotpotQA.
- CPT / LoRA do improve some tasks, contrary to the body's blanket degradation wording.

Table 1, PDF p. 9. Scores follow each benchmark's evaluation metric.

General NLP improves without changing the backbone

Method	Nine-task average
Base	67.86
RAG	70.79
*k*NN-LM	68.50
CPT	70.58
LoRA	67.88
MLP Memory	73.07

- All nine tasks improve over Base; average gain is 5.21 points.
- Examples: CB 69.64 $\rightarrow$ 76.79; RTE 59.57 $\rightarrow$ 64.62.
- A separately trained mixed-domain memory is used for this experiment.

Sec. 5.3, Table 2, PDF p. 10. Domain-conditional PMI scoring.

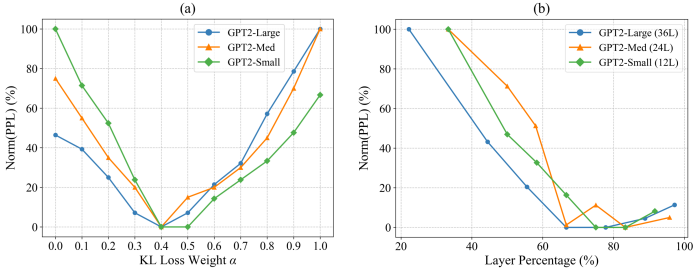
HaluEval improves, but v4 has a column conflict

Main Table 3	Dialogue	QA	Summary
Base	57.18	53.99	50.27
RAG	59.06	65.09	–
Memory	66.86	64.07	52.41

- These are hallucination detection accuracies, not free-generation hallucination rates.
- Appendix Table 5 instead gives best Dialogue = 64.07 and QA = 66.86.
- The best values are swapped across tasks; task-specific gains cannot be reconciled from v4.
- RAG exceeds Memory on QA under the main-table labeling.

Sec. 5.4, Table 3, PDF p. 10; Appendix B, Table 5, PDF p. 19.

Balanced loss and deeper inputs work best



- Best loss balance is near $\alpha = 0.4$ across GPT-2 sizes.
- Input-layer minima lie roughly in the 70–90% depth range; authors summarize this as about 70%.
- Normalized PPL is per curve: values do not compare absolute quality across models.

Sec. 5.5, Fig. 6, PDF p. 11. High alpha means more KL, despite contrary prose.

TTFT and throughput show different speedups

Method	TTFT (ms) ↓	TPS ↑
Base 7B	22	42
RAG top-5	67	24
*k*NN-LM	151	6
Memory + 7B	27	36

- Memory vs RAG: 2.48× TTFT ratio; 1.5× throughput.
- Memory vs *k*NN-LM: 5.59× TTFT ratio; 6× throughput.
- Relative to Base, Memory adds about 23% TTFT and loses about 14% TPS.
- Same Wikipedia corpus; *k*NN uses 4096 → 256 dimensionality reduction.

Fig. 1, PDF p. 1; Sec. 3.3, PDF p. 7. Ratios computed from plotted values.

Distribution evidence suggests a sharper head

Statistic (20K examples)	LM	kNN	Memory
Tokens with nonzero mass	50257	251	50257
Tokens for 80% mass	23	22	13
Tokens for 99% mass	617	126	308

- Memory retains a dense softmax tail but concentrates mass more than the LM.
- At 80% mass, Memory is more concentrated than either comparator.
- Token case studies support rare-entity gains; they do not establish the mechanism across all contexts.

Appendix D/H, Tables 7–8/10, PDF pp. 20–21/26.

Small memory captures much of the measured gain

GPT2-large configuration	WikiText-103 PPL ↓
Base	15.81
+ 8 layers, 221M	11.41
+ 15 layers, 359M	11.35
+ 36 layers, 772M	11.25

- Larger teacher distributions modestly help the small student: 12.91 $\rightarrow$ 12.84.
- Teacher kNN PPL improves 9.43 $\rightarrow$ 9.42 from $k = 1024$ to 2048.
- The latter is a teacher sensitivity test, not retrained memory performance.

Appendices E/F/J; Fig. 8 and Tables 9/13, PDF pp. 22–23/27.

Deploying a stable corpus is the strongest use case

- Useful when offline indexing / distillation can be amortized over many low-latency requests.
- Fixed-size memory inference does not scale with the original datastore size.
- Updating or deleting knowledge requires parameter changes; no online write mechanism is demonstrated.
- The branch is tied to a backbone representation space and vocabulary.
- Explicit retrieval retains source provenance and immediate document updates.

Analysis based on Sec. 3 and reported experiments.

Take-home: retrieval can be a training signal

1. **Mechanism:** compile retrieval distributions into a separate MLP, with mixed soft and hard supervision.
2. **Evidence:** stronger average QA and general NLP scores, plus lower retrieval-related inference latency.
3. **Trade-off:** offline retrieval and training replace online search; corpus updates become a learning problem.
4. **Reading caution:** preserve metric definitions and the v4 HaluEval / interpolation inconsistencies.

References and discussion prompts

- Wei et al. MLP Memory: A Retriever-Pretrained Memory for Large Language Models. arXiv:2508.01832v4.
- ICLR 2026 proceedings; official implementation.

Discussion

- When do lifetime inference savings offset offline retrieval costs?
- Can a gate choose between parametric memory and source-backed retrieval?
- How should updates and provenance work for a distilled memory?

Backup: interpolation strength matters

Appendix Table 5	λ	Best accuracy
Dialogue	0.45	64.07
QA	0.55	66.86
Summarization	0.35	52.41

- At $\lambda = 0.9$, Dialogue = 49.71 and Summary = 46.67, below their baselines.
- At $\lambda = 0.6$, Summary = 49.79, below Base = 50.27.
- A broad claim of uniform robustness over 0.3–0.6 overstates these results.
- These appendix task labels conflict with main Table 3.

Appendix B, Table 5, PDF p. 19.

Backup: FLOPs are not measured end-to-end speed

$$\frac{F_t}{F_m} \approx \frac{4Ld(2d_a + d_f) + 2Ln_{ctx}d_a}{6Ldd_f} = 1 + \frac{n_{ctx}}{12d}$$

$$d_a = d, \qquad d_f = 4d.$$

- L: layers; d: hidden width; d_f: feed-forward width; d_a: attention width.
- Compares matched Transformer and MLP non-embedding computations.
- Does not imply that adding a memory branch makes the original backbone faster.

Appendix C, Eq. (10), Table 6, PDF pp. 19–20.