TURNING DOCUMENTS INTO WEIGHT UPDATES WITH PAIRED MEMORY TOKENS

Anonymous authors
Paper under double-blind review

ABSTRACT

Large language models use new documents through prompting or fine-tuning, requiring either the document in context or additional parameter optimization. We introduce Paired Memory Adaptation (PMA), which learns to convert documents into reusable low-rank weight updates. PMA constructs separate read and write memories inside the frozen language model. Each read memory guides the construction of its paired write memory, and learned projection maps turn their states into update factors. We train the mechanism through context distillation: the adapted model matches a teacher that receives the document in context. New documents then produce updates without per-document gradient optimization. We evaluate fact recall and application in new situations on our Document Acquisition and Retention (DocAR) dataset, with answers assessed by a separate language model. After the same short, 600-step training on Qwen3-4B, PMA outperforms retrained Doc-to-LoRA (D2L) and SHINE on these tests with 50.5% and 19.0% fewer trainable parameters. The share of answers rated fully correct exceeds the unadapted model's by 21.65 percentage points on SQuAD (token F1 and a second judge agree; exact match does not). On most tasks, PMA's performance drops when the update comes from an unrelated document. Removing the learned readout reduces PMA's trainable parameters by 54.4% while retaining useful acquisition, although some public QA scores fall below the unadapted model's. Gains vary across tasks, and updates can disrupt answers to unrelated questions. In separate PMA and D2L experiments, training to preserve those answers reduces this disruption but also reduces gains on document questions. D2L's released checkpoint, trained far longer, beats PMA on nearly all tasks.

1 INTRODUCTION

In-context learning lets a language model use a new document by receiving the text alongside a question. The document occupies part of the context whenever its information is needed. A technical manual, for example, may support many questions over time. Encoding its information into a reusable weight update would let the model answer those questions without the document in its prompt.

The central challenge is to construct that update *before knowing which questions will be asked*. At adaptation time, the mechanism receives only the document. Its update must make the information useful for direct retrieval and for questions that require applying it in a different form. Useful adaptation should also preserve behavior outside the document's scope. The document representation must therefore support both knowledge acquisition and selective changes to future answers.

Learning to generate adaptation parameters offers a way to address this problem without running gradient optimization for every new document. Prior work generates parameters from examples, task descriptions or document context (Phang et al., 2023; Charakorn et al., 2025; Chen et al., 2025). Doc-to-LoRA (D2L) learns to aggregate frozen layerwise document features and predict low-rank adapters (Charakorn et al., 2026). SHINE and CompAs instead learn context representations through a frozen backbone augmented with trainable adapters; SHINE reads them from appended memory tokens (Liu et al., 2026; Jukić et al., 2025). These methods leave an architectural question: how should document representations be constructed for the two factors of a weight update?

We introduce *Paired Memory Adaptation* (PMA), which builds document memories for the complementary *read* and *write* roles of a low-rank update. The read factor determines how the update

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.

responds to later activations; the write factor determines the change added to a layer's output. Separate trainable adapters construct the two memories inside a frozen backbone. We call the read memories keys and the write memories values. Each final key state supplies the input for its paired value computation. A learned *readout* maps these memories into the factors that update a feed-forward projection. Keys supply activations from the projection's input space; values supply states in its output space.

Following Doc-to-LoRA, we train memory construction and readout together through context distillation. The adapted model learns to answer without the document by matching a frozen teacher that receives it in context. After training, a new document produces an update without gradient optimization. The same update can serve multiple questions.

We build our comparison on the released Doc-to-LoRA repository, reusing its training pipeline, data and default AdamW settings to reduce training decisions tailored to PMA. All three methods share the Qwen3-4B-Instruct-2507 backbone, training stream and 600-step budget. We adapt SHINE to this backbone and the same rank-eight feed-forward updates, retaining its memory tokens and Transformer readout at a comparable parameter count, without its pretraining and instruction tuning. We therefore compare the systems as configured, not the released checkpoints; details appear in Section 4 and Appendix B. A separate comparison tests preservation supervision in PMA and Doc-to-LoRA on our Document Acquisition and Retention (DocAR) dataset.

We use a separate language model to assess the answers. At this equal, short training budget, PMA outperforms Doc-to-LoRA and SHINE on DocAR test questions that ask models to recall document information or apply it in new situations. It uses 208.3M trainable parameters, compared with 421.0M for Doc-to-LoRA and 257.3M for SHINE. It also exceeds both comparators on SQuAD and multi-turn MS MARCO, with gains in judged correctness over the frozen base of 21.65 and 5.28 percentage points, respectively. Encoding an unrelated document reduces these gains. SHINE performs better on DROP; HotpotQA shows no clear gain over the base. At this limited training budget, answering with the document in context remains substantially more accurate. Performance at convergence has not been measured. Without training to preserve unrelated answers, PMA disrupts them more than the comparators. Adding this objective reduces disruption at a cost to document learning.

With 95.1M trainable parameters, the memory encoder still learns useful document-specific updates when the readout stays fixed. One training seed is used; a second judge confirms PMA's SQuAD and DocAR leads, while human validation and replication remain open.

Contributions.

- We introduce paired document memories that supply read and write update factors (Section 3).
- We demonstrate acquisition and QA gains at an equal, short training budget, with controls for document use (Sections 5.1–5.2).
- We show that memory activations support acquisition without a learned readout, and analyze architectural and preservation interventions (Sections 5.3–5.4).

2 Related work

Generating adapters from context. An adapter generator learns to produce a model adaptation from examples or a task description. HyperTuning uses a few input–output examples to generate prefixes or LoRA weights (Phang et al., 2023). Text-to-LoRA uses a task description written in natural language (Charakorn et al., 2025). Once trained, these generators can produce an adaptation without running gradient optimization for each new task.

For learning from documents, the generator must produce an update that supports future questions about the text. Doc-to-LoRA gathers document features from a frozen language model and uses a Perceiver and factor heads to generate LoRA adapters through context distillation (Charakorn et al., 2026). Generative Adapter combines transformed token representations through outer products to construct its updates (Chen et al., 2025). These approaches share our goal of making document information available through model parameters.

Learning document memories. SHINE learns the document representation as part of generating an adapter. It adds trainable adapters and memory tokens to a frozen language model. A Transformer combines the memory states across layers and positions, then its output is sliced and reshaped into LoRA factors (Liu et al., 2026). SHINE also reports that outer-product factor arrangements converged much more slowly, and that attention between the two factors' states sped early training but ended worse (Liu et al., 2026, Apps. A.2, E.1). CompAs also learns a context encoder within a frozen backbone. It averages token representations to generate an adapter. Its training objective encourages adapters from different contexts to work together when combined (Jukić et al., 2025).

PMA constructs paired memories using separate adapters for read and write roles. A final key state supplies the input to its paired value computation. The readout uses key activations at the adapted projection's input and value states in its output space, with an independent identity-plus-low-rank map for each role and layer. We study the empirical behavior of this specific memory construction and readout.

Using compressed context. Another way to reuse document information is to replace the original text with a shorter representation. The In-context Autoencoder learns to compress a document into memory slots that are supplied to the model during answering (Ge et al., 2024). HyperLlama turns few-shot examples into reusable soft prefixes, represented by cached attention keys and values (Phang, 2024). In both cases, the model accesses the compressed information through attention. PMA uses its memories to generate weight updates; the memory states are then absent from the question-answering context.

Preserving unrelated behavior. A useful update should change the intended answers while limiting effects elsewhere. Model-editing research studies this requirement as *locality*. MEND learns to turn edit gradients into weight updates and includes a loss that penalizes unwanted changes (Mitchell et al., 2022). ROME edits specified factual associations and tests both generalization to rephrased questions and effects on neighboring facts (Meng et al., 2022).

These methods receive an explicit requested correction. PMA receives a document before its future questions are known. Doc-to-LoRA also tests interference from unrelated contexts and suggests adding irrelevant questions during training (Charakorn et al., 2026, Appendix C.4). Our study measures acquisition alongside retention and tests preservation supervision on the same training data. Preservation means retaining the base model's behavior, which may itself be incorrect.

3 Paired Memory Adaptation

PMA converts a document into a reusable weight update before its future questions are known. During encoding, we add extra input positions called *memory tokens*, as in SHINE (Liu et al., 2026). Their activations are the *memory states*. A key position and its associated value position form a *paired memory*. The *readout* is the learned mapping from these states to the vectors that form the weight update. The collection of updates applied to the model's layers is the *generated adapter*. Figure 1 separates this document-encoding process from subsequent question answering. We use θ for the frozen backbone weights and ϕ for the trainable adaptation parameters. The primary model uses eight memory pairs, separate attention adapters for the two roles, and independent readout maps at each layer. It generates rank-eight updates to the feed-forward output projections in all 36 layers.

3.1 The read and write roles of an update

A weight update must determine both how it responds to an input and what change it contributes to the output. Consider a linear projection $y = Wu$. A rank-one update separates these two roles:

$$\Delta W = ba^\top, \qquad (W + \Delta W)u = Wu + b(a^\top u). \tag{1}$$

The *read vector* a determines the response to activation u. The *write vector* b determines the direction of the added output. The scalar $a^\top u$ controls its signed magnitude. This interpretation connects the associative-memory view of feed-forward layers (Geva et al., 2021; Meng et al., 2022) with read/write constructions in fast-weight models (Schlag et al., 2021).

PMA learns document memories for these complementary roles. A key state supplies the read representation, and its paired value supplies the write representation. Conditioning the value on its

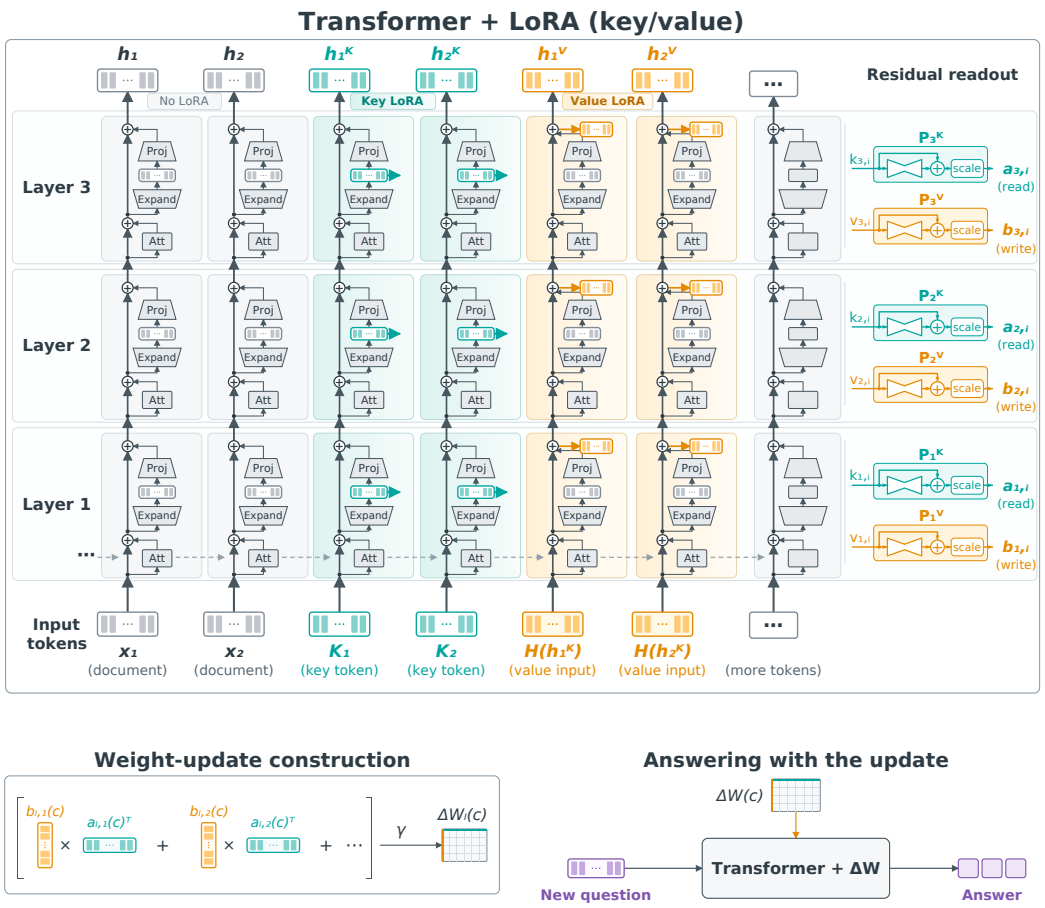

Figure 1: **Paired Memory Adaptation. Top:** document processing is frozen; key and value positions use distinct adapters (teal and amber). Weights are shared across positions within each layer; attention is causal. Three layers and two pairs are illustrative. **Lower left:** read/write outer products form layer updates. **Lower right:** installed updates support questions without document text or memory tokens. H includes final normalization. P_ℓ^K and P_ℓ^V in Equation 3 denote the complete readout maps, including the identity path, low-rank correction (drawn as a bottleneck) and scaling. $\Delta W(c)$ collects the layer-specific updates.

key lets the write representation depend on the document information gathered by that key. The write vector contributes an additive change to the existing computation. Low rank restricts the directions of that change; whether it respects the document's semantic scope depends on what training learns.

As in SHINE, we construct these memories with the backbone they will adapt. This reuses its pretrained transformations and gives memory construction access to the target model's representation spaces. We hypothesize that this makes it easier for the readout to learn useful update factors. Memory positions and question positions can still produce different activation distributions, so their compatibility must be learned.

3.2 Constructing Paired Document Memories

Given a document c, we append r learned key embeddings $e_1^K, \ldots, e_r^K$. Causal attention lets each key read the document and preceding keys. Trainable low-rank adapters modify the attention projections at these memory positions. The document itself is processed with the frozen backbone. We complete the key computation through all decoder layers before constructing the value inputs.

Each key then supplies the input to a paired value position. Let h_i^K be key i's residual state after the final decoder layer. The *conditioning map* H applies the backbone's final normalization followed by a learned transformation:

$$e_i^V = H(h_i^K). \tag{2}$$

The resulting vector is a continuous input embedding for value position i. We process these inputs through the same backbone, reusing the cached document and key attention states. Thus final-layer key information can influence value construction from its first layer onward. Using hidden states as inputs to later computation also appears in latent reasoning (Hao et al., 2024).

The value positions follow the entire key segment and use a separate set of trainable adapters. Each value can attend to the document, all keys and preceding values. The paired conditioning path therefore coexists with communication through attention. Individual pairs are distributed representations; we do not assume that each pair corresponds to one fact.

3.3 Generating and using the weight update

We adapt the feed-forward output projection $W_\ell \in \mathbb{R}^{d_{\text{out}} \times d_{\text{in}}}$ at each selected layer ℓ. For key position i, we take the activation $k_{\ell i} \in \mathbb{R}^{d_{\text{in}}}$ entering this projection. For its paired value position, we take the residual state $v_{\ell i} \in \mathbb{R}^{d_{\text{out}}}$ after the decoder layer. These choices place the two memories in the input and output spaces of the projection being updated.

The readout maps P_ℓ^K and P_ℓ^V in Figure 1 turn the extracted states into a *read factor* $a_{\ell i}$ and *write factor* $b_{\ell i}$:

$$\begin{aligned} a_{\ell i}(c) = P_\ell^K k_{\ell i}(c), \qquad b_{\ell i}(c) = P_\ell^V v_{\ell i}(c), \\ \Delta W_\ell(c) = \gamma \sum_{i=1}^{r} b_{\ell i}(c) a_{\ell i}(c)^\top. \end{aligned} \tag{3}$$

Each projection adds a learned low-rank correction to the extracted memory state and scales the result. The correction is initialized at zero, giving the memory state a direct path to the update factor. It uses fewer parameters than a dense map. The update has rank at most r, the number of memory pairs.

We also evaluate these maps held fixed after initialization (Section 5.3).

The fixed adapter scale γ multiplies the sum once. We apply the update as a LoRA modification (Hu et al., 2022). Dimensions, initialization and scaling details are given in Appendix A.

When answering, the model uses $W_\ell + \Delta W_\ell(c)$. The memory-construction adapters are inactive, and the question's context contains neither the document nor the memory positions. The generated factors can be reused across questions. PMA has no document-independent shared adapter; the study resets the update between documents.

3.4 Learning acquisition and preservation

Training teaches the memories and readout which updates help answer questions about a document. A frozen teacher receives both the document and a question. The adapted student receives the question and its generated update. The acquisition loss distills the teacher's answer distribution into the student along teacher-answer prefixes. As in Doc-to-LoRA, we use stored top-16 teacher probabilities with their original probability mass.

Preservation supervision addresses questions whose answers the document should not change. For those questions, a separate loss measures $\mathrm{KL}(p_\theta \| p_{\phi,c})$ along the frozen base model's answer prefixes, where p_θ is the unadapted model and $p_{\phi,c}$ is the model with the document update. This encourages retention of base behavior, including cases where that behavior is incorrect.

The training objective combines acquisition, preservation and factor regularization:

$$\mathcal{L} = \mathcal{L}_{\text{acq}} + \lambda \mathcal{L}_{\text{pres}} + \eta \mathcal{R}_{\text{factors}}. \tag{4}$$

The term $\mathcal{R}_{\text{factors}}$ penalizes factor magnitudes, with weight η. The primary comparison of PMA, Doc-to-LoRA and SHINE uses acquisition-only training ($\lambda = 0$). A separate study compares $\lambda = 0$ and $\lambda = 1$ for PMA and Doc-to-LoRA using the DocAR training set. Within each method, samples,

order, initialization and regularization are matched. Appendix A.2 specifies the sample weighting and exact loss reductions.

Gradients pass through the adapted model to the readout and memory encoder while the pretrained backbone weights remain fixed.

4 EXPERIMENTAL SETUP

Systems and common training recipe. We implement all three methods in the released Doc-to-LoRA repository (Charakorn et al., 2026), sharing its data loading, packing, distillation loss and training loop. The frozen backbone is Qwen3-4B-Instruct-2507; all methods generate rank-eight updates to the feed-forward output projections in its 36 layers. Table 1 summarizes their differences. D2L retains its Perceiver, factor heads and learned document-independent adapter. SHINE retains its memory tokens and Transformer readout, adapted to the 4B dimensions and common output modules. It uses 39 memory positions and a rank-64 attention Meta-LoRA, keeping its parameter count between PMA's and D2L's, without its pretraining or instruction-tuning stages.

Table 1: **Architectures and trainable parameters.** Counts exclude the frozen 4B backbone and refer to the configurations trained here (Appendices A–B).

Method	Document representation	Readout	Trainable parameters (M)
D2L	Frozen features + Perceiver	Factor heads	421.00
SHINE	39 learned memory positions	M2P Transformer	257.31
PMA	Eight read/write memory pairs	Separate linear maps	208.30

Training data and optimization. We use a 71,902-document subset of D2L's released context-distillation data, drawn primarily from FineWeb-Edu, with additional data from PwC, SQuAD, DROP and ROPES. Documents are paired with questions and cached teacher responses generated with the document in context. In the primary comparison, all three methods are trained from fresh parameters on the same 340,324 question–document presentations (30.2M answer tokens) in the same order over 600 optimizer steps, with the AdamW defaults of D2L's released code: learning rate 4×10^{-5}, 100 warm-up steps and cosine decay. We evaluate the final checkpoint at nominal update strength. We thus measure acquisition under this allowance, not at convergence or equal compute; D2L (Charakorn et al., 2026, App. C.2) and SHINE (Liu et al., 2026, Sec. 5.4) also compare designs at 50% and 40% of their schedules, though ours is far shorter. With its gate started at one or after 1,300 steps, D2L stays close to its 600-step judged results (Appendices B, D.4).

We construct DocAR using model-generated documents and questions, supplemented with edited public passages. It tests fact recall, application of supplied information and preservation of unrelated answers. A separate PMA/D2L study uses the 17,431-document DocAR training set, matching samples, order and initialization within each method to compare acquisition-only training with added preservation supervision.

Tasks and reference conditions. We evaluate D2L's SQuAD, DROP and ROPES panels (497, 492 and 500 questions), 1,000 HotpotQA questions, and 1,000 MS MARCO conversations with 15 questions each (Rajpurkar et al., 2016; Dua et al., 2019; Lin et al., 2019; Charakorn et al., 2026; Yang et al., 2018; Nguyen et al., 2016; Liu et al., 2026). The DocAR test subset contains 200 document bundles from a separate reserved test pool, with probes for *direct acquisition* (recalling supplied facts or instructions), *indirect acquisition* (applying them in new situations), and *preservation*. Near-miss preservation tests related questions outside the intended scope; unrelated preservation tests other topics. To test document use, we also generate an update from a different document that does not answer the question, then ask the original question with the same answer target. References use the frozen base with no document or with the correct document in context. Decoding is greedy. For MS MARCO, the same adapter serves all turns, and each condition retains its own previous answers as history. Selection and generation limits appear in Appendix C.

Scoring and uncertainty. GPT-6 Luna grades one answer at a time, blinded to method identity, using a fixed rubric for each task family. QA correctness measures the fraction of fully correct answers;

DocAR acquisition measures full satisfaction of the probe. Retention measures full preservation of the base answer's substantive claims and the probe's explicit constraints, including baseline mistakes. Partial answers do not count as full passes. Every row of a table is scored on the same questions (the released D2L omits a few unresolved answers; Appendix C.3). Error bars show 95% confidence intervals from 2,000 resamplings of the evaluation source groups. Questions sharing an article, document or conversation are kept together, and each comparison uses the same questions for both methods. The intervals describe uncertainty from the evaluation sample for one training seed and one judge. Appendix C gives selection, scoring and statistical details; Appendix D.1 gives F1 and EM.

5 Results and Analysis

5.1 Answering Questions from Document Updates

Table 2: **QA correctness (%).** Answers rated fully correct, on the same questions for all systems; MQA covers all 15 conversation turns. Adapted systems use seed 42 and 600 steps; *D2L, released* is D2L's published checkpoint (100,000 steps), shown for reference. Bold marks the best 600-step system. Figure 2 reports paired differences; counts are in Appendix C.3.

System	SQuAD	DROP	ROPES	HotpotQA	MQA
Frozen base	21.86	10.54	44.08	25.05	37.96
Document in context	94.02	74.19	85.31	81.21	95.70
D2L, released	74.85	23.23	30.82	29.72	54.26
D2L	24.74	5.59	34.87	24.31	39.06
SHINE	29.28	**16.77**	40.57	24.63	40.35
PMA	**43.51**	13.12	**50.00**	**25.37**	**43.24**
Scored questions	485	465	456	942	14,651

PMA's clearest public gains are on SQuAD and MS MARCO (Table 2). On SQuAD, it reaches 43.51% judged correctness, compared with 24.74% for D2L, 29.28% for SHINE and 21.86% for the base. Across MS MARCO conversations, PMA improves over the base by 5.28 points and over D2L/SHINE by 4.18/2.89 points. All of these paired intervals exclude zero (Figure 2). PMA recovers 41% and 32% of the gains over the base that D2L's released checkpoint reaches on these tasks after 100,000 steps; D2L retrained for 600 steps recovers 5% and 7%.

The remaining tasks show the limits of this result. PMA exceeds both comparators on ROPES, but its gain over the base is less certain: the 95% interval includes zero. SHINE exceeds PMA on DROP by 3.66 points; PMA's gain over the base is also less certain there. HotpotQA shows no clear difference among the three adapted methods or from the base. At this 600-step training budget, the in-context reference remains far ahead, including 94.02% on SQuAD and 95.70% on MS MARCO. The remaining gap to answering from the text has not been measured at convergence.

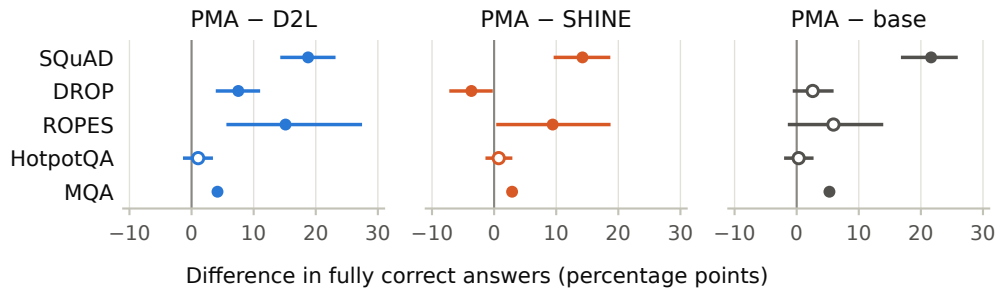

Figure 2: **Differences in QA correctness.** Dots show observed percentage-point differences; bars show 95% confidence intervals from resampling evaluation source groups. Positive values favor PMA; filled dots mark intervals that exclude zero. The same questions as Table 2 are used throughout.

5.2 Acquisition and retention on DocAR

Table 3: **Acquisition and retention on the DocAR test subset (%).** Adapted models use acquisition-only training on D2L's training data. Acquisition requires full probe satisfaction; retention measures preservation of base answers. All rows use the same probes; *D2L, released* is as in Table 2. Bold marks the best 600-step system.

	Acquisition		Retention	
System	Direct	Indirect	Near-miss	Unrelated
Frozen base	3.94	12.90	100.00	100.00
Document in context	64.36	46.15	28.29	40.47
D2L, released	66.61	52.96	18.82	35.01
D2L	6.57	15.28	17.02	34.84
SHINE	8.76	16.66	**17.44**	**35.04**
PMA	**25.04**	**26.81**	13.13	29.71
Scored probes	1,142	1,675	6,869	1,952

PMA achieves 25.04% direct and 26.81% indirect acquisition, compared with 6.57/15.28% for D2L and 8.76/16.66% for SHINE (Table 3). Under this acquisition-only objective, PMA retains fewer near-miss and unrelated answers than both comparators. Even the in-context reference often changes base behavior. The later supervision experiment tests whether a preservation objective reduces this.

Does the model use the encoded document? We generate an update from an unrelated document and ask the same questions about the original document. PMA's direct acquisition then falls from about 25% to 3%, and indirect acquisition from about 27% to 12%, close to the base's 4% and 13%. Figure 3 shows the advantage of encoding the correct document. This advantage, together with gains over the base on SQuAD and MS MARCO, supports useful document learning; D2L shows no clear advantage at this budget (Appendix B). A wrong-document update can also harm existing answers: on HotpotQA, PMA scores 6.89 points higher with the correct document than with an unrelated one, yet barely exceeds the base.

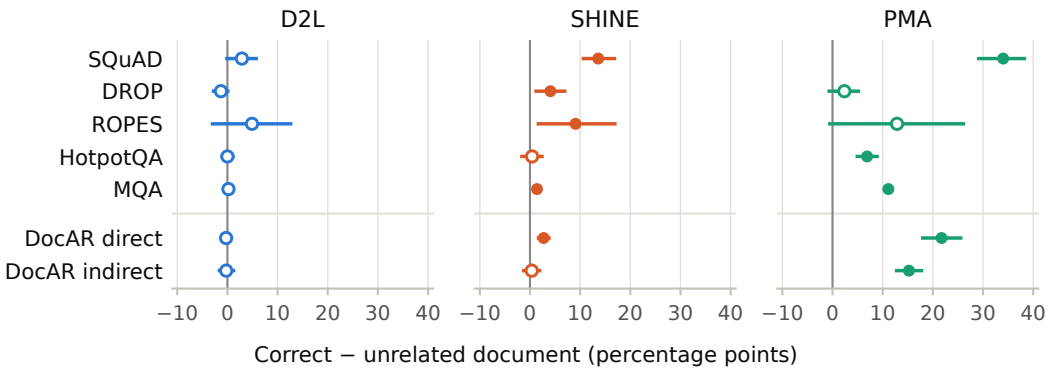

Figure 3: **Correct versus unrelated document updates.** Each dot compares the same questions answered with an update from their source document or an unrelated document. Positive values favor the correct document; bars show 95% confidence intervals, and filled dots mark those excluding zero.

5.3 What does the learned readout contribute?

PMA without the learned readout starts from the same initial parameters and training stream as full PMA, but trains only the memory encoder. It has 95.1M trainable parameters: 54.4% fewer than full PMA and 63.1% fewer than SHINE. On the DocAR test subset, its direct/indirect acquisition is 21.03/25.15%, versus SHINE's 8.76/16.67% (Table 4). SHINE remains stronger on DROP, ROPES and retention.

Table 4: **Readout ablation and architecture variants (%).** All use seed 42 and 600 steps. M denotes trainables in millions; Near/Unrel. are retention. PMA and SHINE repeat Tables 2 and 3; capacity variants appear in Appendix D.5.

		QA correctness			Acquisition		Retention	
Configuration	M	SQuAD	DROP	ROPES	Direct	Indirect	Near	Unrel.
PMA	208.30	43.51	13.12	50.00	25.04	26.81	13.13	29.71
SHINE	257.31	29.28	16.77	40.57	8.76	16.66	17.44	35.04
PMA w/o learned readout	95.05	35.00	4.53	23.99	21.03	25.15	13.14	32.84
Trained value embeddings	207.66	37.71	14.44	51.35	22.96	25.39	13.88	31.80

The ablation also depends on the document: on paired control questions, its direct acquisition falls from 21.02% to 4.90% with an unrelated document, and indirect acquisition from 25.13% to 12.00%. Thus the learned memory encoder can produce useful update factors directly. Learning the readout improves all three QA scores and direct acquisition; its indirect benefit is less certain. Removing it improves unrelated retention relative to full PMA. These results concern the complete memory encoder and do not isolate pairing alone. Capacity, sharing and value-input variants are detailed in Appendix D.5. Full PMA remains the primary model across tasks.

5.4 Preservation supervision changes the tradeoff

Table 5: **Effect of preservation supervision (%).** Adapted models use the same DocAR training set and 600-step budget, and are evaluated on the DocAR test subset. The teacher is the frozen model given the correct document in context. All rows use the same probes.

		Acquisition		Retention	
System	Training objective	Direct	Indirect	Near-miss	Unrelated
Frozen base	—	3.94	12.90	100.00	100.00
Teacher (in context)	—	64.36	46.15	28.37	40.02
D2L	Acquisition only	23.47	19.16	19.86	33.52
D2L	+ preservation	17.34	16.48	47.54	48.36
PMA	Acquisition only	41.33	27.28	18.98	32.50
PMA	+ preservation	31.70	18.63	44.83	46.01
Scored probes		1,142	1,675	6,859	1,954

Adding preservation supervision improves retention at an acquisition cost in both methods (Table 5). For PMA, near-miss retention rises from 18.98% to 44.83%, while direct acquisition falls from 41.33% to 31.70%. All four within-method changes for each system have intervals excluding zero (Appendix D.6). PMA has higher acquisition point estimates under both objectives, with retention 0.9–2.7 points below D2L's.

6 Discussion and conclusion

We introduced PMA, which builds separate read and write memories from a document and maps their activations into a reusable low-rank update, so new documents need no per-document gradient steps and later questions no document in context. Under a common 600-step budget, PMA improves DocAR acquisition and outperforms the configured D2L and SHINE on SQuAD and multi-turn MS MARCO with fewer trainable parameters; document controls tie these gains to the supplied document, and a second judge confirms the SQuAD and DocAR leads (Appendix D.3). With its readout fixed, the memory encoder still yields useful acquisition. The contribution of pairing remains to be isolated, gains vary across tasks, and updates disrupt unrelated answers, which preservation training reduces at an acquisition cost. The study uses one backbone, one seed and a short budget, and evaluation informed development (Appendix C); robustness across seeds and budgets and a validated measure of the retention tradeoff are the next steps toward reliable document adaptation.

AI USE STATEMENT

Generative AI tools assisted with software implementation, data analysis, and manuscript drafting and editing. As part of the method, models generated document/probe material and teacher answers; GPT-6 Luna supplied the primary answer grades and Claude Opus 5.5 a second set, as detailed in Appendices C.2 and D.3. Mathematical descriptions were checked against the implemented computation, and reported tables and plots were generated from saved result records. These checks do not replace independent scientific or human validation. The authors retain responsibility for the methods, claims and final manuscript.

REPRODUCIBILITY STATEMENT

Appendix A specifies the adaptation and initialization, Appendix B the training conditions and data derivation, and Appendix C the scoring and uncertainty definitions. The supplementary material contains code to train PMA, D2L, SHINE and PMA without the learned readout under the common recipe, export the generated updates, generate answers and judge SQuAD answers, together with the SQuAD evaluation panel, the DocAR test subset used in the paper and CPU tests.

ETHICS STATEMENT

Document adaptation can introduce erroneous information or change behavior outside its intended scope. Preservation in this study means retaining specified base behavior and is not a guarantee of safety or factual correctness. Model-generated probes may reproduce biases or ambiguities of their generators. DocAR contains both synthetic text and edited public passages. Any redistribution must preserve the original datasets' license and attribution requirements. Independent human validation of the generated probes and answer grades has not been completed.

REFERENCES

Artificial Analysis. Artificial Analysis Intelligence Index. https://artificialanalysis.ai/models, 2026. Version 4.3.2, accessed 26 September 2026.

Rujikorn Charakorn, Edoardo Cetin, Yujin Tang, and Robert Tjarko Lange. Text-to-LoRA: Instant transformer adaption. In *Proceedings of the 42nd International Conference on Machine Learning*, 2025. URL https://arxiv.org/abs/2506.06105.

Rujikorn Charakorn, Edoardo Cetin, Shinnosuke Uesaka, and Robert Tjarko Lange. Doc-to-LoRA: Learning to instantly internalize contexts. In *Proceedings of the 43rd International Conference on Machine Learning*, 2026. URL https://openreview.net/forum?id=iW1oBBO72S.

Tong Chen, Hao Fang, Patrick Xia, Xiaodong Liu, Benjamin Van Durme, Luke Zettlemoyer, Jianfeng Gao, and Hao Cheng. Generative adapter: Contextualizing language models in parameters with a single forward pass. In *International Conference on Learning Representations*, 2025. URL https://openreview.net/forum?id=bc3sUsS6ck.

Dheeru Dua, Yizhong Wang, Pradeep Dasigi, Gabriel Stanovsky, Sameer Singh, and Matt Gardner. DROP: A reading comprehension benchmark requiring discrete reasoning over paragraphs. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pp. 2368–2378. Association for Computational Linguistics, 2019. doi: 10.18653/v1/N19-1246. URL https://aclanthology.org/N19-1246/.

Tao Ge, Jing Hu, Lei Wang, Xun Wang, Si-Qing Chen, and Furu Wei. In-context autoencoder for context compression in a large language model. In *International Conference on Learning Representations*, 2024. URL https://arxiv.org/abs/2307.06945.

Mor Geva, Roei Schuster, Jonathan Berant, and Omer Levy. Transformer feed-forward layers are key-value memories. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pp. 5484–5495. Association for Computational Linguistics, 2021. doi: 10.18653/v1/2021.emnlp-main.446. URL https://aclanthology.org/2021.emnlp-main.446/.

Shibo Hao, Sainbayar Sukhbaatar, DiJia Su, Xian Li, Zhiting Hu, Jason Weston, and Yuandong Tian. Training large language models to reason in a continuous latent space, 2024. URL https://arxiv.org/abs/2412.06769.

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://arxiv.org/abs/2106.09685.

Josip Jukić, Martin Tutek, and Jan Šnajder. Context parametrization with compositional adapters, 2025. URL https://arxiv.org/abs/2509.22158v3.

Kevin Lin, Oyvind Tafjord, Peter Clark, and Matt Gardner. Reasoning over paragraph effects in situations. In *Proceedings of the 2nd Workshop on Machine Reading for Question Answering*, pp. 58–62. Association for Computational Linguistics, 2019. doi: 10.18653/v1/D19-5808. URL https://aclanthology.org/D19-5808/.

Yewei Liu, Xiyuan Wang, Yansheng Mao, Yoav Gelberg, Haggai Maron, and Muhan Zhang. SHINE: A scalable in-context hypernetwork for mapping context to LoRA in a single pass. In *Proceedings of the 43rd International Conference on Machine Learning*, volume 306 of *Proceedings of Machine Learning Research*, 2026. URL https://arxiv.org/pdf/2602.06358v3.

Kevin Meng, David Bau, Alex Andonian, and Yonatan Belinkov. Locating and editing factual associations in GPT. In *Advances in Neural Information Processing Systems*, volume 35, 2022. URL https://arxiv.org/abs/2202.05262.

Eric Mitchell, Charles Lin, Antoine Bosselut, Chelsea Finn, and Christopher D. Manning. Fast model editing at scale. In *International Conference on Learning Representations*, 2022. URL https://arxiv.org/abs/2110.11309.

Tri Nguyen, Mir Rosenberg, Xia Song, Jianfeng Gao, Saurabh Tiwary, Rangan Majumder, and Li Deng. MS MARCO: A human generated MAchine reading COmprehension dataset. In *Proceedings of the Workshop on Cognitive Computation: Integrating Neural and Symbolic Approaches, co-located with NIPS 2016*, 2016. URL https://arxiv.org/abs/1611.09268.

Jason Phang. Investigating the effectiveness of HyperTuning via gisting, 2024. URL https://arxiv.org/abs/2402.16817.

Jason Phang, Yi Mao, Pengcheng He, and Weizhu Chen. HyperTuning: Toward adapting large language models without back-propagation. In *Proceedings of the 40th International Conference on Machine Learning*, volume 202 of *Proceedings of Machine Learning Research*, pp. 27854–27875, 2023. URL https://proceedings.mlr.press/v202/phang23a.html.

Pranav Rajpurkar, Jian Zhang, Konstantin Lopyrev, and Percy Liang. SQuAD: 100,000+ questions for machine comprehension of text. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pp. 2383–2392. Association for Computational Linguistics, 2016. doi: 10.18653/v1/D16-1264. URL https://aclanthology.org/D16-1264/.

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, 2021. URL https://proceedings.mlr.press/v139/schlag21a.html.

Zhilin Yang, Peng Qi, Saizheng Zhang, Yoshua Bengio, William W. Cohen, Ruslan Salakhutdinov, and Christopher D. Manning. HotpotQA: A dataset for diverse, explainable multi-hop question answering. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pp. 2369–2380. Association for Computational Linguistics, 2018. doi: 10.18653/v1/D18-1259. URL https://aclanthology.org/D18-1259/.

A METHOD DETAILS

A.1 MEMORY CONSTRUCTION AND READOUT

The backbone has 36 decoder layers, residual width 2,560 and feed-forward width 9,728. PMA uses eight memory pairs. Separate rank-64 adapters act on the attention query, key, value and output projections at key and value positions, with dropout 0.05. Document positions, backbone weights and base token embeddings remain frozen. Key embeddings are learned; the captured states and readout maps are specific to each layer.

The conditioning map in Equation 2 is

$$z = \mathrm{Norm}_\theta(h), \qquad H(h) = s_H \left[z + U_H V_H z + b_H \right]. \tag{5}$$

The complete readout maps in Equation 3 are

$$P_\ell^K = g_\ell^K \left(I + U_\ell^K V_\ell^K \right), \qquad P_\ell^V = g_\ell^V s_\ell \left(I + U_\ell^V V_\ell^V \right). \tag{6}$$

Each correction has rank 128: for width d, $V \in \mathbb{R}^{128\times d}$ and $U \in \mathbb{R}^{d\times 128}$. The U matrices and b_H start at zero. The gains g_ℓ^K, g_ℓ^V and s_H are fixed after calibration; s_ℓ is learned, initialized to one, and shared across a layer's memory pairs. It scales only the write factor. The fixed adapter multiplier is $\gamma = 2r^{3/2}$, approximately 45.25 at $r = 8$, applied once to the factor product.

Initialization. Before training, we set the gains once using generic text inputs, without answer labels or evaluation examples. A fixed text probe sets s_H to the ratio of embedding-table root-mean-square (RMS) magnitude to normalized final-state RMS. This sets the starting scale of the value inputs.

For the readout gains, we cross eight generic documents with eight question prompts. A temporary, untrained paired-factor readout of the encoder's residual memory states supplies a target correction RMS t_ℓ. Matching this reference keeps the direct readout's initial correction magnitude comparable to the reference parameterization. Let d_ℓ be PMA's initial update RMS and μ_ℓ^K, μ_ℓ^V its factor mean absolute magnitudes. The gain multipliers are

$$c_\ell = \frac{t_\ell}{d_\ell}, \qquad m_\ell^K = \sqrt{c_\ell \frac{\mu_\ell^V}{\mu_\ell^K}}, \qquad m_\ell^V = \sqrt{c_\ell \frac{\mu_\ell^K}{\mu_\ell^V}}. \tag{7}$$

This matches the target RMS and balances the factor magnitudes. The gains remain fixed during training, and the reference readout is discarded. This is an initialization choice; its effect on final answer accuracy has not been isolated experimentally.

A.2 TRAINING OBJECTIVES

For document c and a relevant question x, let y^T be the frozen context teacher's answer and $\mathcal{T}_{16}(t)$ its top-16 tokens at position t. The acquisition loss is

$$\ell_{\mathrm{acq}}(c, x) = -\frac{1}{|y^T|} \sum_t \sum_{w \in \mathcal{T}_{16}(t)} p_\theta(w \mid c, x, y^T_{<t}) \log p_{\phi,c}(w \mid x, y^T_{<t}). \tag{8}$$

Teacher probabilities retain their original mass; they are not renormalized over the top-16 set. The student's softmax spans the full vocabulary. For a preservation question, with frozen-base answer y^0,

$$\ell_{\mathrm{pres}}(c, x) = \frac{1}{|y^0|} \sum_t \mathrm{KL}\left[p_\theta(\cdot \mid x, y^0_{<t}) \,\middle\|\, p_{\phi,c}(\cdot \mid x, y^0_{<t}) \right]. \tag{9}$$

The reference has neither the document nor any adapter. Both losses exclude prompt and padding positions and include the eligible natural end token; preservation uses the full vocabulary.

Let R_j and Q_j indicate acquisition and preservation samples among M document–question pairs. The reductions are

$$\mathcal{L}_{\mathrm{acq}} = \frac{1}{M} \sum_{j=1}^{M} R_j \ell_{\mathrm{acq},j}, \qquad \mathcal{L}_{\mathrm{pres}} = \frac{1}{M} \sum_{j=1}^{M} Q_j \ell_{\mathrm{pres},j}. \tag{10}$$

M counts both roles within each worker's accumulation window; worker gradients are averaged. With acquisition-only training, preservation samples contribute zero task loss but remain in the denominator and factor regularization. Both objectives therefore use the same samples and acquisition-loss scaling. Repeated documents are weighted through their question samples.

The factor regularizer averages absolute entries over layers, rank and feature dimensions, sums the read/write terms, and averages over sample presentations. D2L's shared factors are included. Its coefficient is 0.1; the preservation coefficient is zero or one as declared for each run.

B Training and data

B.1 Configurations

All generators are trained from fresh adaptation parameters with the common D2L-based pipeline and the settings in Table 6. Evaluation uses seed 42, the final 600-step checkpoint and adaptation strength one. Initialization, data and order are matched within each preservation-supervision comparison. The primary and architecture comparisons use D2L's training data.

Table 6: **Common training settings.** The preservation term is active only in its declared supervision condition.

Setting	Value
Optimizer	AdamW; learning rate 4×10^{-5}
Schedule	100 warm-up steps; cosine decay to 10^{-7} over 600 steps
Adam parameters	$(\beta_1, \beta_2) = (0.9, 0.999)$; $\epsilon = 10^{-8}$
Weight decay / gradient clipping	0.01 / norm 1
Nominal batch	88 packs: 8 workers $\times$ 11 accumulation microsteps
Packed context / question-and-answer limits	2,048 / 1,024 tokens
Teacher probability support	Top 16; original mass retained
Preservation / factor penalty	0 or 1 / 0.1
Student embedding noise	NEFTune amplitude 5
Precision	BF16 backbone; FP32 trainable parameters and generated factors

Baseline configurations. All three methods are trained from fresh parameters with the common recipe and distillation objective, so the comparison is between architectures at equal training rather than between released checkpoints, which follow longer, method-specific schedules (D2L: 80,000 single-chunk and 20,000 chunked steps; SHINE: reconstruction/completion pretraining and QA instruction tuning). The recipe uses D2L's default optimizer settings and single-chunk training.

D2L retains its frozen feature encoder, Perceiver, factor heads and shared document-independent rank-eight adapter, so its applied update has rank up to 16; its generated write-factor scale is initialized to zero, as released. At this budget the shared adapter carries most of D2L's update, and its correct-document advantage is small (Figure 3): the generated scale grows slowly (mean 0.0045 after 600 steps, versus 0.057 in the released checkpoint); Appendix D.4 reports D2L with this scale initialized at one and trained for 1,300 steps. Trained on DocAR, D2L's direct acquisition rises from 3.94% for the base to 23.47% (Table 5).

SHINE retains its memory embeddings, M2P Transformer (four blocks, width 2,560, 20 heads, feed-forward width 5,120, alternating layer/token attention) and initialization. It generates the same rank-eight feed-forward updates as the other methods (published SHINE: all seven projections). Its Meta-LoRA has rank 64 on the attention projections with dropout 0.05, the configuration of each of PMA's two role adapters, which keeps SHINE's trainable parameters between those of PMA and D2L (Table 1); the published configuration uses rank 128 on all seven projections. SHINE applies one adapter set to document and memory positions; PMA uses separate sets for keys and values. Training adds the common factor-level L1 penalty (coefficient 0.1), whereas SHINE's released configurations set their own output penalty to zero.

Following the rounding-up rule in SHINE's paper, we use $\lceil 8(9{,}728 + 2{,}560)/2{,}560 \rceil = 39$ memory positions. Each layer's output supplies 98,304 factor entries; the remaining 1,536 entries are discarded

before constructing that layer's factors. Total trainable parameter counts appear in Table 1; they include all trained components and exclude the frozen backbone.

B.2 Training corpora

D2L's training data. The subset contains 71,902 document records and 717,660 eligible questions. Sampling probabilities are 0.96 for FineWeb-derived questions and 0.01 each for PwC, SQuAD, ROPES and DROP. D2L's released token IDs, answer spans and teacher probabilities are preserved. Documents average about 283 tokens and reach 2,036 tokens. A separate 200-context development set is reserved from source shards; the recorded check finds no exact context-token match with training. Within-training duplicates are retained. Each question is a separate sample, so documents can be encoded repeatedly. The primary runs consume 52,800 packs, 340,324 question–document presentations and 30,205,185 acquisition-answer tokens.

DocAR training set. Each bundle contains a document, its intended change, and questions that should acquire that change or retain base behavior. Content includes factual assertions and denials, multi-fact and edited SQuAD passages, concept definitions, rules, explanations, style/naming instructions and distractors. Documents, probe sets and critiques were generated using Qwen3-235B-A22B-2507, DeepSeek-V4-Flash and Gemini-3.7/3.8-Flash, followed by structural checks and filtering.

Training eligibility excludes unusable or over-limit records, truncated targets, conflicting duplicates and specified cross-partition matches. It also requires complete auxiliary target fields in the prepared data, although the reported objectives use only teacher and base distributions. The training set contains 17,431 document bundles, of which 538 contain only preservation questions. Across all bundles, there are 97,648 acquisition samples and 268,094 preservation samples; each sample is one document–question pair. Preservation samples occur both in those 538 bundles and in bundles with acquisition questions. Documents average 55.1 tokens and reach 431 tokens; answer targets are capped at 256 tokens. Acquisition uses cached context-teacher probabilities, while preservation uses base-answer prefixes.

The two objectives use identical files and training order. Each run consumes 52,736 packs and 368,272 samples, including 8,707,710 acquisition and 32,050,997 preservation answer tokens. These are actual counts across the epoch boundary. The 200-document development monitor is drawn from the DocAR validation split and is separate from the DocAR test subset. D2L's training data and the DocAR training set differ in document lengths, prompts, target answers and the amount of acquisition supervision.

C Evaluation protocol

C.1 Populations and generation

Table 7: **Evaluation populations and decoding limits.** MQA contains 1,000 conversations with 15 questions each. All generation is greedy.

Population	Questions/probes	Encoded document	New-token cap
SQuAD	497	Passage	256
DROP	492	Passage	256
ROPES	500	Background + situation	256
HotpotQA	1,000	Distractor-setting passages	128
MS MARCO MQA	15,000	Concatenated source passages	500
DocAR test	13,543	One document per bundle	256

The SQuAD, DROP and ROPES panels are the 500-question validation subsets selected by D2L's released evaluator (Charakorn et al., 2026), excluding passages longer than 512 tokens (3 SQuAD and 8 DROP questions); exact counts are in Table 7. D2L reports ROUGE-L relative to the in-context model, so our scores are not comparable to its published numbers. Selection does not depend on base-model correctness. HotpotQA uses 1,000 distractor-validation examples. MQA uses a seeded

sample of 1,000 contexts from SHINE's 9,639-context test file (Liu et al., 2026), retaining all turns. No extension document is truncated or selected using supporting-fact labels.

The DocAR test subset uses 200 bundles from the 1,010-bundle reserved test corpus, selected proportionally by document kind with a fixed within-kind order. It contains 1,143 direct, 1,675 indirect, 8,387 near-miss and 2,025 unrelated probes; 313 probes outside these four roles are not used in the reported comparisons. The tasks cover knowledge and behavioral adaptation, so acquisition means satisfying the declared probe.

Within each population, all methods share questions, prompts, extraction and decoding limits. Adapted answering excludes document text and memory tokens. References bypass all generated and shared adapters. MQA retains each condition's own generated history, never gold answers, and reuses one adapter throughout the conversation. Later-turn effects therefore include history and question-difficulty differences.

C.2 Answer scoring

GPT-6 Luna judges one candidate per request, without method names, original scores or another method's answer. Requests use medium reasoning effort, a 4,096-token output cap and a fixed seed. The same rubric is used across methods and conditions within each task family.

For QA, the judge receives the document, question and reference answers. The document defines factual support and the question defines required detail; reference answers are fallible guides. A *correct* answer supplies all required information without material false claims. *Partial* means a correct subset with missing required information and no contradiction. Wrong essential claims, unsupported answers or incompatible alternatives are *incorrect*. Unclear questions or unresolved source evidence are *ungradable*. Paraphrases and harmless elaboration are accepted. Flagged sources are adjudicated without candidate answers before affected responses are regraded.

DocAR judging additionally receives the behavior contract and the base answer when required. Preservation requires retaining substantive base claims and stance, including mistakes, while honoring explicit constraints. Exact wording is required only by an explicit format/completion condition. Contracts requiring peers receive the same method's peer answers; probability-only contracts without an observable semantic target remain ungradable. For unrelated-document updates, acquisition is judged against the original question's correct source document and answer target.

Retention scoring. Retention is measured on probes whose requirements the base answer already satisfies, using the same questions for all methods. Partial answers do not count as full passes.

Judge validation. Eighteen synthetic QA checks and eleven DocAR checks passed before full scoring. Repeating 90 pilot requests produced 86 identical verdicts. Repeating 500 QA and 200 DocAR requests on the scoring endpoint reproduced 96.0% and 90.0% of verdicts exactly, and 97.2% and 92.0% of the correct/not-correct decisions. Appendix D.3 compares a second judge and the DocAR rules; human-label validation remains open. Judged correctness was chosen as the primary measure after inspecting lexical score disagreements. F1 and exact match are retained in Appendix D.1.

C.3 Question counts and uncertainty

Table 8 gives the exact question counts used to compute the reported scores. Each comparison uses the same questions for every method. Correct, partial and incorrect answers all enter the denominator; questions without a clear source-based judgment or a completed judgment for every compared method are omitted. Retention also requires the base answer to satisfy the probe, as described above. Variant rows of the architecture comparison use the updated source interpretation, which omits three SQuAD and six ROPES questions with incomplete regrading; their paired differences from PMA use this support for all rows, whereas the D2L, SHINE and PMA rows of the architecture tables repeat the primary tables. The *D2L, released* reference row uses the same questions, omitting those its answers left unresolved (5 ROPES, 12 MQA, 1 direct, 3 near-miss and 21 unrelated). That checkpoint is evaluated as distributed, with its 4-bit-quantized context encoder; our retrained D2L uses an unquantized encoder.

Table 8: **Questions used in each comparison.** Main results refer to Tables 2–3, architecture variants to Table 16, and preservation supervision to Table 5. A dash indicates a task not evaluated in that comparison.

Dataset / outcome	Main results	Architecture	Preservation supervision
SQuAD	485	480	—
DROP	465	464	—
ROPES	456	446	—
HotpotQA	942	—	—
MQA	14,651	—	—
DocAR: Direct	1,142	1,141	1,142
DocAR: Indirect	1,675	1,674	1,675
DocAR: Near-miss	6,869	6,859	6,859
DocAR: Unrelated	1,952	1,909	1,954

Rates count full passes divided by scored questions/probes. DocAR roles and QA datasets are reported separately. Confidence intervals use 2,000 paired percentile-bootstrap draws with analysis seed 42. Groups follow source dependencies: SQuAD articles, DROP passages, ROPES backgrounds, DocAR families, and conversation/document groups for the extension. Document controls join each source group with its distractor group before resampling. Their original QA controls have 24/136/22 groups, and DocAR direct/indirect controls have 101/103. The small ROPES group count limits precision. Comparisons between systems use each table's questions; document comparisons use the questions answered with both documents. Intervals condition on one checkpoint and judge, without accounting for training-seed variation or multiple comparisons.

Evaluation scope. An 8-gram overlap screen (threshold 0.30) flags passages in the SQuAD, DROP and ROPES validation sets, including one exact DROP passage, as well as two HotpotQA and six MQA sources. Reported semantic results retain the frozen populations; source and backbone-pretraining exposure cannot be excluded. DocAR uses family/pool/passage partitions, but evaluation outputs informed development.

Factor round-trip and zero-update checks support the implementation. Recorded native/export and cross-runtime greedy decoding differences remain fidelity limitations for some checkpoints. The comparisons concern one backbone and budget. Updates reset between documents, and serving efficiency or cumulative learning is not evaluated.

Reading the measures. Adapted systems often answer in full sentences despite the short-answer instruction. The judge accepts such elaboration, while exact match, and on some tasks F1, penalize it (Table 9); lexical and judged scores are therefore best read together. An update from an unrelated document can also move answers below the unadapted model, as on SQuAD, so the document-use contrasts in Figure 3 combine document use with this interference. Retention is measured against the base model's own answers, which even the in-context reference often changes (Table 3); retention rates are therefore most informative when compared across methods. DocAR documents are short and model-generated, targeting specific acquisition and retention behaviors rather than long-document internalization.

D Supporting results

D.1 Standard benchmark scores

On ROPES, PMA's F1 is below the base, despite a higher judged point estimate whose interval includes zero. SQuAD shows a different pattern, which we examine below on the same questions for every metric.

Table 9: **Token F1 / exact match (%)** on the same questions as Table 2. MQA includes all turns.

System	SQuAD	DROP	ROPES	HotpotQA	MQA
Frozen base	18.14 / 8.87	9.83 / 4.95	38.66 / 29.61	22.28 / 12.10	23.73 / 4.73
Document in context	76.19 / 58.97	43.40 / 29.03	67.67 / 58.77	64.07 / 45.86	54.63 / 10.76
D2L, released	30.47 / 1.44	8.36 / 0.22	10.41 / 1.33	18.87 / 6.90	35.30 / 2.14
D2L	13.46 / 1.24	5.80 / 0.65	17.38 / 3.73	17.58 / 7.43	29.56 / 1.75
SHINE	20.40 / 7.84	13.72 / 7.31	24.18 / 13.38	22.26 / 11.25	28.01 / 4.76
PMA	24.01 / 5.36	9.40 / 1.94	21.40 / 5.70	19.40 / 8.49	30.06 / 3.97

Table 10: **SQuAD metrics on the same questions.** All scores are percentages on the 485 questions shared by the five rows in Table 2. Only fully correct judge verdicts count as correct.

System	Token F1	Exact match	Judged correct
Frozen base	18.14	8.87	21.86
Document in context	76.19	58.97	94.02
D2L	13.46	1.24	24.74
SHINE	20.40	7.84	29.28
PMA	24.01	5.36	43.51

D.2 Interpreting scoring disagreements

PMA exceeds the base on both F1 (24.01 vs. 18.14) and judged correctness (43.51 vs. 21.86), but has lower exact match (5.36 vs. 8.87). Thus, on SQuAD, F1 and judged correctness agree on the direction of the comparison; exact match differs. Of PMA's 211 answers marked correct by the judge, 186 are not exact matches, compared with 64 of 106 for the base. These counts describe the saved verdicts, not independently verified correctness.

Exact match requires the answer to match a reference after normalization; token F1 measures word overlap, taking the best score over the references. Table 11 shows how a correct sentence can lose exact match, an equivalent number can receive zero F1, and an incorrect number can retain high F1. These effects can favor either system. Lexical scores also reflect adherence to the requested short-answer format.

These selected examples illustrate limits of lexical scoring; they do not establish the reliability of the aggregate judged ranking; Appendix D.3 reports agreement with a second judge and with the DocAR rules.

Table 11: **Examples of lexical scoring limitations.** Selected from the same 485 SQuAD questions after inspecting saved answers and scores. Answers are unabridged; source passages are excerpted and duplicate references are shown once. F1 and EM are percentages. These illustrations do not estimate error rates.

System	Full answer	F1	EM	Saved verdict
A. Correct answer in a full sentence				
Question: What was the name of the upgraded Saturn I called?				
References: "Saturn IB"; "S-IB"				
Source excerpt: "The Saturn IB was an upgraded version of the Saturn I."				
Base	"Saturn IB"	100.0	100	Correct
PMA	"The upgraded Saturn I was called the Saturn IB."	44.4	0	Correct
Both answers name the same rocket; the extra words reduce lexical scores.				
B. Equivalent numbers, different tokens				
Question: How many museums comprise Harvard Art Museums?				
References: "three museums."; "three"				
Source excerpt: "The Harvard Art Museums comprises three museums."				
Base	"3"	0.0	0	Correct
PMA	"Harvard Art Museums comprises 11 museums."	25.0	0	Incorrect
The correct digit receives zero F1; the wrong PMA answer receives partial overlap credit.				
C. High overlap with a wrong number				
Question: The United States is divided into how many jurisdictions?				
References: "five"; "The United States is divided into five jurisdictions"				
Source excerpt: "The United States is divided into five jurisdictions: Northeastern, Southeastern, North Central, South Central and Western."				
In context	"Five"	100.0	100	Correct
PMA	"The United States is divided into four jurisdictions."	85.7	0	Incorrect
The passage describes church jurisdictions. PMA changes the key number but retains most reference words.				

D.3 Judge agreement checks

We re-graded saved answers with a second judge from another model family, Claude Opus 5.5, whose maximum-effort configuration leads the Artificial Analysis Intelligence Index (Artificial Analysis, 2026); we ran it at low reasoning effort with the same rubrics and request payloads: every SQuAD answer of Table 2 and a random half of the DocAR acquisition probes (606 direct, 592 indirect) for the five main rows. On SQuAD, the two judges agree on 96.3% of correct/not-correct decisions (Cohen's $\kappa = 0.92$; Table 12); PMA's rate is identical, and PMA's differences from D2L and SHINE remain significant. Of PMA's 186 judged-correct SQuAD answers that are not exact matches, the second judge accepts 171. On the DocAR sample, both judges rank PMA above SHINE and D2L on direct and indirect acquisition ($\kappa = 0.84$ and 0.61). The second judge is more lenient for every system, especially for longer answers, and gives PMA slightly larger leads (direct PMA $-$ SHINE $+18.7$ versus $+16.2$ on the same probes).

The rule-based grading contracts attached to each DocAR probe agree with GPT-6 Luna on 89.6% of direct-acquisition answers ($\kappa = 0.66$). The judge is stricter (15.6% versus 22.5% of answers pass) and gives a smaller PMA $-$ SHINE difference on direct acquisition than the rules ($+16.3$ versus $+20.2$ points). Judged and rule-based retention check different criteria, full preservation of the base answer's claims versus the probe's contract, and agree poorly ($\kappa \approx 0.1$).

D.4 D2L's gate and a longer training check

D2L multiplies its generated write factor by a learned scale initialized at zero (Appendix B). To test whether this initialization or the short budget explains D2L's behavior, we trained D2L with the scale initialized at one (two sample orders) and D2L and PMA for 1,300 steps, otherwise under the primary recipe, and graded their saved answers with the same judge, rubrics and settings (Table 13). D2L's 1,300 steps took 112.6 minutes on eight H200 GPUs, longer than PMA's 600 steps (107.9 minutes).

Table 12: **Second judge on SQuAD (%).** Answers of Table 2 (485 questions, 48 source groups) graded by GPT-6 Luna and by Claude Opus 5.5 with the same rubric; differences with 95% intervals from 2,000 paired resamplings of source groups.

	GPT-6 Luna	Claude Opus 5.5
Frozen base	21.86	23.92
Document in context	94.02	94.85
D2L	24.74	25.98
SHINE	29.28	30.72
PMA	43.51	43.51
PMA − D2L	+18.76 [14.29, 23.19]	+17.53 [12.79, 22.09]
PMA − SHINE	+14.23 [9.58, 18.71]	+12.78 [7.47, 17.83]

Starting the scale at one moves D2L's loss reduction onto the generated update: removing that update raises its distillation loss by 0.37 (0.08 in the primary run), and removing the shared adapter by 0.06 (0.47). Yet D2L's direct acquisition stays within 1.1 points of the unrelated-document control in all three runs and within 1.0 point of the primary D2L run (intervals include zero). On SQuAD, D2L's correct − unrelated gap rises to about 9 points because unrelated-document updates lower its accuracy (16.3–16.9% versus 21.9%), while its correct-document accuracy changes by $+0.8$ to $+1.9$ points (not significant). PMA at 600 steps exceeds each of these runs by 16.9–17.9 points on SQuAD and 17.5–17.7 on direct acquisition, and retains 5.3–5.7 points fewer unrelated answers. PMA at 1,300 steps scores 44.5% on SQuAD ($+1.0$ [-2.2, 4.5] over 600 steps); its DocAR answers were not retained for judging, and under the probes' rule-based contracts its direct acquisition rises from 33.2% to 45.8%. Each run uses one seed; D2L's learning rate and batch are unchanged, and SHINE was not trained beyond 600 steps. Figure 4 shows the training curves.

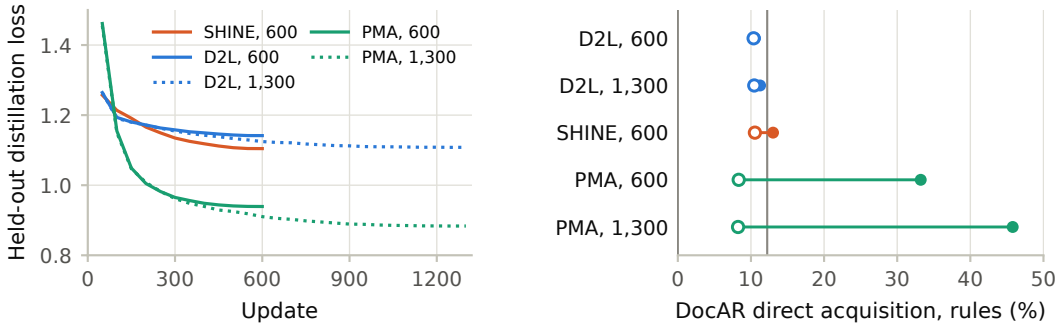

Figure 4: **Training curves.** Left: held-out distillation loss on 64 development samples of D2L's data every 50 updates (at update 0: PMA 2.27, SHINE 2.40, D2L 2.45). Right: DocAR direct acquisition under the probes' rule-based contracts with the correct (filled) or an unrelated (open) document; the gray line is the frozen base. Seed 42, D2L's training data.

Table 13: **D2L's gate and 1,300 steps (%, judged).** Seed 42, D2L's training data. The 600-step and base rows repeat Tables 2 and 3; new rows are graded on the same questions, less a few unresolved answers. Gaps are correct − unrelated document with 95% intervals (joined source groups). PMA's 1,300-step DocAR answers were not retained.

System	SQuAD	Direct	Indirect	SQuAD gap	Direct gap
Frozen base	21.86	3.94	12.90	—	—
D2L, 600 steps	24.74	6.57	15.28	+2.9 [−0.4, 6.1]	−0.3 [−1.3, 0.6]
SHINE, 600 steps	29.28	8.76	16.66	+13.6 [10.3, 17.2]	+2.7 [1.4, 4.1]
PMA, 600 steps	43.51	25.04	26.81	+34.0 [28.8, 38.6]	+21.7 [17.6, 25.9]
D2L, gate at one	26.60	7.36	16.21	+9.7 [5.5, 13.8]	+0.9 [−0.5, 2.2]
D2L, gate at one, 2nd order	25.57	7.54	17.28	+9.3 [4.5, 13.6]	+1.1 [−0.3, 2.7]
D2L, 1,300 steps	25.77	7.54	16.69	+8.8 [4.8, 13.6]	+0.9 [−0.3, 2.1]
PMA, 1,300 steps	44.54	—	—	+31.8 [25.9, 36.9]	—

D.5 Architectural interventions

PMA without the learned readout. All readout parameters are frozen after the same initialization and calibration as full PMA. The correction branches remain zero and write scales remain one, giving $P_\ell^K = g_\ell^K I$ and $P_\ell^V = g_\ell^V I$. The update remains active and document-specific. Key embeddings, role adapters and key-to-value conditioning remain trainable: 95,050,240 parameters in total. The count excludes frozen readout tensors. Initial parameters, training data and order match full PMA; optimization and clipping act on the remaining trainable parameters. The comparison therefore tests readout learning with calibration held fixed.

Table 14: **PMA w/o learned readout: paired differences** in percentage points, with 95% intervals, on the architecture comparison's questions.

Outcome	Ablation − full PMA	Ablation − SHINE
SQuAD	−8.96 [−13.16, −4.79]	+5.42 [0.00, 10.65]
DROP	−8.62 [−11.65, −5.51]	−12.07 [−15.65, −8.70]
ROPES	−26.68 [−38.63, −12.03]	−17.49 [−28.18, −7.71]
Direct	−4.03 [−7.15, −1.27]	+12.27 [8.63, 16.24]
Indirect	−1.67 [−3.81, 0.40]	+8.48 [6.02, 11.12]
Near-miss	+0.00 [−0.77, 0.79]	−4.30 [−5.37, −3.23]
Unrelated	+3.14 [1.37, 4.86]	−2.30 [−4.23, −0.43]

Learning the readout improves all three public QA scores and direct acquisition; its indirect benefit is less certain. Removing it leaves near-miss retention similar and improves unrelated retention by 3.14 points. Compared with SHINE, the ablation has higher direct/indirect acquisition but lower DROP, ROPES and retention scores; its higher SQuAD point estimate has an interval including zero.

Table 15: **Document use without the learned readout.** Correct versus unrelated document rates (%), their difference and 95% interval, scored pairs n and groups G.

Outcome	Correct doc.	Unrelated doc.	Difference [95% interval]	n	G
SQuAD	34.71	11.16	+23.55 [19.01, 28.00]	484	24
DROP	4.51	6.87	−2.36 [−4.87, 0.00]	466	136
ROPES	23.32	19.96	+3.36 [−10.90, 17.97]	446	22
Direct	21.02	4.90	+16.11 [12.36, 20.17]	1,142	101
Indirect	25.13	12.00	+13.13 [10.84, 15.37]	1,675	103

Correct-document scores also exceed the base on SQuAD and direct/indirect acquisition on the DocAR test subset. Together with the document controls, this supports useful learning through extracted memory activations. The identity path, fixed scaling and full memory encoder remain, so the experiment does not isolate pairing alone.

Capacity and value-input variants. Attention and MLP ranks describe the encoder's LoRA adapters, with separate adapters for key and value positions in each layer. Full PMA uses attention rank 64, no encoder MLP adapters, and separate rank-128 readout corrections per role and layer. Attention 96 increases only attention rank. MLP 16 retains attention rank 64 and adds rank-16 adapters on the MLP gate, up and down projections.

The shared-readout variant uses attention rank 96 and MLP rank 32. One pair of read-correction matrices is shared across all 36 layers, and a separate pair of write-correction matrices is shared across layers. Each correction has rank 1,024, with 1,024 learned coefficients per layer and role to weight its shared directions. The read and write matrices and layer coefficients are trained jointly. All variants still generate rank-eight document updates. This experiment changes sharing and encoder capacity together.

Trained value embeddings replace key-conditioned value inputs with learned embeddings reused across documents. Values still attend to the document, keys and previous values. This diagnostic uses the same training data as full PMA with a different sample order. All use seed 42 and 600 steps.

Table 16: **Full architecture panel (%).** D2L, SHINE and PMA repeat Tables 2 and 3; variant rows are scored on the architecture comparison's support (Appendix C.3). Abbreviations follow Table 4.

		QA correctness			Acquisition		Retention	
Configuration	M	SQuAD	DROP	ROPES	Direct	Indirect	Near	Unrel.
D2L	421.00	24.74	5.59	34.87	6.57	15.28	17.02	34.84
SHINE	257.31	29.28	16.77	40.57	8.76	16.66	17.44	35.04
PMA	208.30	43.51	13.12	50.00	25.04	26.81	13.13	29.71
Attention 96	255.48	43.12	14.22	50.00	26.82	30.35	13.15	29.54
Attention 64 + MLP 16	250.76	42.92	17.89	51.35	27.08	31.00	13.65	30.28
Shared readout + wider encoder	252.41	43.75	15.30	52.91	27.78	30.29	15.29	31.69
Trained value embeddings	207.66	37.71	14.44	51.35	22.96	25.39	13.88	31.80
PMA w/o learned readout	95.05	35.00	4.53	23.99	21.03	25.15	13.14	32.84

Table 17: **Variant minus original PMA:** percentage-point differences and 95% paired intervals, on the architecture comparison's questions.

Outcome	Attention 96	Attention 64 + MLP 16	Shared readout	Trained values
SQuAD	−0.83 [−3.54, 2.12]	−1.04 [−3.86, 2.19]	−0.21 [−3.77, 3.46]	−6.25 [−10.91, −1.63]
DROP	+1.08 [−1.17, 3.49]	+4.74 [1.97, 7.88]	+2.16 [−0.45, 4.98]	+1.29 [−1.77, 4.51]
ROPES	−0.67 [−3.53, 3.95]	+0.67 [−1.84, 4.65]	+2.24 [−3.98, 9.83]	+0.67 [−7.17, 6.21]
Direct	+1.75 [−0.57, 4.00]	+2.02 [0.00, 4.19]	+2.72 [−0.36, 5.84]	−2.10 [−4.49, 0.09]
Indirect	+3.52 [1.69, 5.37]	+4.18 [2.27, 5.97]	+3.46 [1.54, 5.46]	−1.43 [−3.31, 0.31]
Near-miss	+0.01 [−0.58, 0.62]	+0.51 [−0.13, 1.16]	+2.16 [1.28, 3.07]	+0.74 [−0.01, 1.51]
Unrelated	−0.16 [−1.61, 1.35]	+0.58 [−0.81, 1.92]	+1.99 [0.38, 3.67]	+2.10 [0.42, 3.69]

These interventions improve selected outcomes, without establishing a uniform benefit of greater capacity. The MLP-16 variant's DROP point estimate exceeds SHINE's, but a significant advantage has not been established. The trained-value-embedding diagnostic's SQuAD decline provides limited evidence about the explicit conditioning path, given its retained attention access.

D.6 Preservation supervision

The two objectives share samples, acquisition targets, initialization and order. Their rates and paired effects use the same six-row support as Table 5: 1,142 direct, 1,675 indirect, 6,859 near-miss and 1,954 unrelated probes. Preservation samples remain in the loss denominator and factor regularization under both objectives.

The roughly tenfold KL reduction accompanies improved behavioral retention and lower acquisition in both methods. The KL measurements and decoded answer grades remain separate measures.

Table 18: **Saved preservation KL (nats/token)** from the base to adapted model along base-answer prefixes. Acquisition and retention rates are in Table 5.

Method	Training objective	Near-miss KL	Unrelated KL
D2L	Acquisition only	0.2152	0.1133
	+ preservation	0.0194	0.0090
PMA	Acquisition only	0.2707	0.1296
	+ preservation	0.0254	0.0120

Table 19: **Adding preservation supervision:** percentage-point changes and 95% paired intervals, on the same probes as Table 5.

Outcome	D2L	PMA
Direct	−6.13 [−8.86, −3.48]	−9.63 [−12.72, −6.80]
Indirect	−2.69 [−5.02, −0.52]	−8.66 [−11.34, −6.00]
Near-miss	+27.69 [25.79, 29.61]	+25.85 [23.87, 27.93]
Unrelated	+14.84 [12.66, 16.95]	+13.51 [10.61, 16.66]