ISOLATING THE TRAINING OBJECTIVE IN TEST-TIME KNOWLEDGE INCORPORATION

Anonymous authors
Paper under double-blind review

ABSTRACT

Test-time training (TTT) can write a document into the adapter parameters of a language model with a few gradient updates, and recent methods improve this process with components that require training of their own, such as a meta-learned update, a teacher or a generator trained by reinforcement learning. All such methods share a training objective, and because it can be changed without additional training, what it achieves alone is the natural reference for these components. This reference has not been measured. In the standard setting, with LoRA applied to the query and value projections of Qwen2.5-7B, no objective significantly exceeds next-token training, yet with LoRA applied to all linear projections the best objective exceeds it by 18.5 points, so the placement can hide the effect of the objective. We therefore decompose the objective into four components and compare fourteen objectives that change one component at a time, on two backbones, two placements and two datasets. Shaping the loss rarely improves accuracy. Accuracy instead depends on how much distinct text the loss covers, and falls monotonically as passage tokens are removed from the loss. Guided by this finding, we propose MIX3, which supervises the full passage together with the model's own question–answer pairs and rewrites. In ten updates and without a trained generator, teacher or outer loop, it reaches 73.7% and 73.5% closed-book accuracy on Qwen2.5-7B and Llama-3.1-8B-Instruct, and on facts newer than the training data it improves retrieval while keeping judgment at the level of the unadapted model. MIX3 thus provides a reference for costlier components, and our results suggest that objectives should be compared under more than one placement.

1 INTRODUCTION

Language models acquire broad knowledge during pretraining, but their parameters stay fixed while new documents keep appearing. Test-time training (TTT) (Sun et al., 2020; Hardt & Sun, 2024; Bansal et al., 2026) can incorporate such a document by writing it into a small set of trainable parameters with a few gradient updates, so that later questions are answered without the document in context (Zweiger et al., 2025; Chen et al., 2023; 2026; Kuratov et al., 2026; Eyuboglu et al., 2025). Recent methods improve this process with components that require substantial training of their own, such as a meta-learned update (Chen et al., 2023; 2026; Kuratov et al., 2026), a distilled teacher (Eyuboglu et al., 2025) or a generator of adaptation data trained by reinforcement learning (Zweiger et al., 2025). Whether these components justify their cost remains open.

Every method that incorporates a document through gradient updates trains the adapter with a training objective, which specifies the sequences on which the adapter is trained and how their tokens enter the loss. The objective itself, however, is rarely varied in isolation. It is studied together with the components above or in other regimes (Zweiger et al., 2025; Allen-Zhu & Li, 2024; Ovadia et al., 2024; Akyürek et al., 2025), and studies of where LoRA is applied keep it fixed (Hu et al., 2022; Wang et al., 2026). This makes it difficult to assess whether improvements stem from these components or from the objective they are paired with, although the objective is the only component that can be changed without additional training. We therefore ask how much the objective alone contributes to test-time knowledge incorporation, and on what its contribution depends.

Answering this question is less straightforward than it appears. In the setting in which Zweiger et al. (2025) introduced this form of incorporation, Qwen2.5-7B with LoRA applied to the query

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.

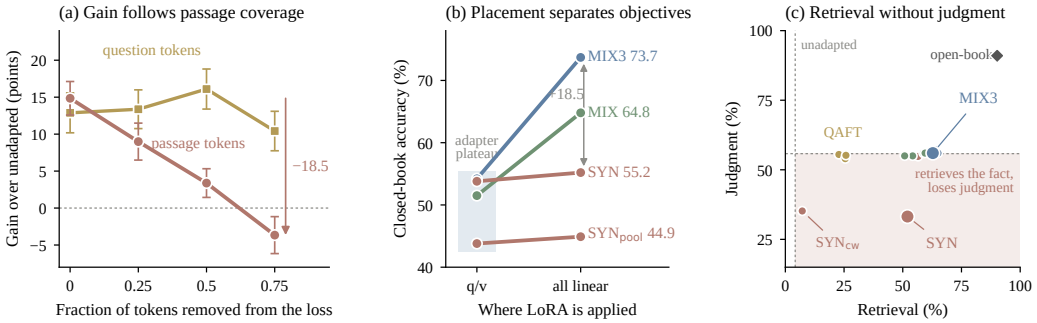

Figure 1: (a) Gain in closed-book accuracy over the unadapted model when a random fraction ρ of the passage tokens (from SYN) or of the question tokens (from question–answer pairs) is removed from the loss, on Qwen2.5-7B with LoRA on the query and value projections; bars show ± 1 standard error over 100 passages (Appendix D.1). (b) Closed-book accuracy on SQuAD with LoRA applied to the query and value projections (q/v) or to all linear projections (Qwen2.5-7B); the shaded band marks the range of the four objectives under q/v. (c) Retrieval against judgment on WikiDYK (Llama-3.1-8B-Instruct); dashed lines mark the unadapted model, and the shaded region contains objectives that retrieve the fact more often than the unadapted model but judge claims about it less accurately. Colors follow Table 1.

and value projections, none of the fourteen objectives significantly exceeds next-token training on the passage and its generated implications, although these objectives include those of prior work and variants that change one of their components. This bound is not a limit of the model: with the passage in its context, the same model answers 35 points more of the questions. However, as shown in Figure 1b, with LoRA applied to all linear projections, the best objective exceeds next-token training by 18.5 points, so the bound depends on the placement. Therefore, studying the objective requires varying the placement as well, and changing one component at a time.

Accordingly, as shown in Table 1, we decompose the objective into four components (§3) and build fourteen objectives, each variant changing one component of a configuration. And according to the results shown in Table 2, only one of the ten such contrasts on SQuAD, and two of the ten on WikiDYK, improve accuracy significantly, so shaping the loss rarely helps. As shown in Figure 1a, what matters is how much text the loss covers. Removing passage tokens from the loss lowers the gain monotonically, removing question tokens does not, and adding distinct self-generated text raises it. Guided by this, we propose MIX3, which supervises the full passage together with the model's own question–answer pairs and rewrites. In ten updates and without a trained generator, teacher or outer loop, it reaches 73.7% and 73.5% closed-book accuracy on Qwen2.5-7B and Llama-3.1-8B-Instruct (see Table 2), and as shown in Figure 1c, on facts newer than the training data it improves retrieval while keeping judgment at the level of the unadapted model.

Our contributions are as follows:

- We formalize the objective of test-time knowledge incorporation as four components, namely the training sequences, the loss mask, the shaping of the loss and the update schedule, and compare fourteen objectives that change one component at a time on two backbones, two LoRA placements and two datasets, with paired tests over the same documents (§3, §4).
- We show that the objective acts through how much distinct text its loss covers rather than through the shaping of the loss, and derive from this finding MIX3, which reaches 73.7% and 73.5% closed-book accuracy on the two backbones without a trained generator, teacher or outer loop while preserving judgment on new facts (§5.3–§5.7).
- We show that whether objectives differ depends on where LoRA is applied and on the backbone, since no objective significantly exceeds next-token training on Qwen2.5-7B under the default placement and the effects of individual components change sign across settings. Comparisons of objectives under a single placement can therefore mislead (§5.2, §5.5).

2 Related Work

Test-time knowledge incorporation. Test-time training updates model weights on data that arrives at test time (Sun et al., 2020; Hardt & Sun, 2024; Bansal et al., 2026). For knowledge incorporation, the document is written into the weights so that it can be removed from the context (Chen et al., 2023; Zweiger et al., 2025). Recent methods add components that require training of their own, such as a meta-learned update (Chen et al., 2023; 2026; Kuratov et al., 2026), a distilled teacher (Askell et al., 2021; Eyuboglu et al., 2025) or a generator trained by reinforcement learning (Zweiger et al., 2025), and evaluate each together with its own objective.

Training text and loss. Question–answer pairs and rewrites make knowledge extractable in pretraining (Allen-Zhu & Li, 2024), and paraphrases or synthetic text improve the recall of new facts in continued training (Ovadia et al., 2024; Yang et al., 2025). Which tokens receive the loss has been studied in instruction tuning (Shi et al., 2024) and in few-shot reasoning (Akyürek et al., 2025).

Where LoRA is applied. Hu et al. (2022) and Biderman et al. (2024) compare the modules to which LoRA is applied, and Wang et al. (2026) learn which layers to update for each context, all with the objective fixed. Appendix A discusses these works in more detail.

3 Test-Time Knowledge Incorporation

We study the no-context knowledge-incorporation setting of Zweiger et al. (2025). Each document $\mathbf{D}$ comes with evaluation question–answer pairs $\mathcal{Q}(\mathbf{D}) = ((\mathbf{Q}_i, \mathbf{A}_i))_{i=1}^{n_\mathbf{D}}$, where $\mathbf{Q}_i$ is answerable from $\mathbf{D}$ and $\mathbf{A}_i$ is its reference answer. Let $\mathcal{V}$ be the vocabulary, $\Delta(\mathcal{V})$ the set of distributions over it, and $\mathbb{P}_{\boldsymbol{\theta},\boldsymbol{\phi}}$ a causal language model with frozen backbone parameters $\boldsymbol{\theta}$ and LoRA parameters $\boldsymbol{\phi}$ (Hu et al., 2022). We denote by C the adaptation configuration, comprising the placement and rank of the LoRA adapter, the optimizer, and the learning rate. The placement, rank, optimizer and learning rate are held fixed whenever objectives are compared, with a single exception described in §5.1. The adapter is initialized at $\boldsymbol{\phi}_0$ such that $\mathbb{P}_{\boldsymbol{\theta},\boldsymbol{\phi}_0} = \mathbb{P}_{\boldsymbol{\theta}}$.

Material and objectives. The adaptation material for $\mathbf{D}$ consists of the document and three texts generated by the unadapted model from fixed prompts: implications, question–answer pairs, and rewrites. The generated components are sampled once and reused across objectives; prompts, decoding and parsers are given in Appendix C.2. An objective Ω turns this material into a set of training sequences $\mathcal{R}_\Omega(\mathbf{D})$. Each sequence $\mathbf{X}$ comes with a loss mask $S_\mathbf{X} \subseteq \{2, \dots, |\mathbf{X}|\}$, the nonempty set of token positions that contribute to the loss.

The objective is a weighted cross-entropy over these training sequences, minimized by gradient updates indexed by $k = 1, \dots, K_\Omega(\mathbf{D})$. For the k-th update, the objective selects a minibatch $\mathcal{B}_k \subseteq \mathcal{R}_\Omega(\mathbf{D})$ and assigns a sequence weight $w_{k,\mathbf{X}} \geq 0$ to each $\mathbf{X} \in \mathcal{B}_k$, and a token weight $c_{k,\mathbf{X},t} \geq 0$ and a target distribution $\boldsymbol{\tau}_{k,\mathbf{X},t} \in \Delta(\mathcal{V})$ to each token x_t of $\mathbf{X}$ with $t \in S_\mathbf{X}$. The weights are normalized so that

$$\sum_{\mathbf{X} \in \mathcal{B}_k} w_{k,\mathbf{X}} = 1, \qquad \sum_{t \in S_\mathbf{X}} c_{k,\mathbf{X},t} = 1 \quad \text{for every } \mathbf{X} \in \mathcal{B}_k.$$

With $H(\boldsymbol{\tau}, \mathbf{p}) = -\sum_{v \in \mathcal{V}} \boldsymbol{\tau}(v) \log \mathbf{p}(v)$ the cross-entropy, the minibatch loss is

$$L_k^\Omega(\boldsymbol{\phi}) = \sum_{\mathbf{X} \in \mathcal{B}_k} w_{k,\mathbf{X}} \sum_{t \in S_\mathbf{X}} c_{k,\mathbf{X},t}\, H(\boldsymbol{\tau}_{k,\mathbf{X},t}, \mathbb{P}_{\boldsymbol{\theta},\boldsymbol{\phi}}(\cdot \mid \mathbf{X}_{<t}))\,. \tag{1}$$

An objective is thus specified by four components, namely the training sequences $\mathcal{R}_\Omega(\mathbf{D})$, the loss masks $S_\mathbf{X}$, the shaping of the loss through $w_{k,\mathbf{X}}$, $c_{k,\mathbf{X},t}$ and $\boldsymbol{\tau}_{k,\mathbf{X},t}$, and the update schedule through the minibatches $\mathcal{B}_k$. The one-hot target $\boldsymbol{\tau}_{k,\mathbf{X},t} = \boldsymbol{\delta}_{x_t}$ recovers next-token prediction, and a target produced by the frozen backbone with $\mathbf{D}$ in context gives context distillation. Per-sequence objectives use one sequence per minibatch and pooled objectives use all of them, so that for a fixed number of epochs the former take $|\mathcal{R}_\Omega(\mathbf{D})|$ times as many updates. Table 1 lists the fourteen objectives and what each changes, namely the training sequences and their loss masks, the targets and weights, or the grouping into minibatches, and Appendix B.1 gives the full definitions.

Adaptation and evaluation. At the k-th update, one step of the optimizer specified by C is applied to $\boldsymbol{\phi}_{k-1}$ using the gradient $\nabla_{\boldsymbol{\phi}} L_k^\Omega(\boldsymbol{\phi}_{k-1})$, and we write $\boldsymbol{\phi}^{\Omega,C}(\mathbf{D}) := \boldsymbol{\phi}_{K_\Omega(\mathbf{D})}$ for the adapter

Table 1: The fourteen objectives, grouped into bands by the kinds of material they draw on. Each band lists material configurations (§4.1) and, indented beneath them, the variants derived from them (§4.2), each changing one component as given in *Change* and *Axis*. *Training sequences* lists the sets each objective trains on. RQA belongs to the three-kind band because its question–answer pairs are prefixed with rewrites ($\mathcal{R}_{\mathrm{q}}^{\mathrm{r}}$). *Source* gives the work from which each objective or change is taken or adapted, and marks controls and the objective we propose. $\kappa(\mathbf{D})$ denotes a fixed passage prefix.

Objective	Training sequences	Change	Axis	Source
Passage material				
Train on Passage + Synthetic (SYN)	$\mathcal{R}_{\mathrm{p}}$	–	material	Zweiger et al., 2025
SYN$_{\mathrm{cw}}$	$\mathcal{R}_{\mathrm{p}}$	$c_{k,\mathbf{X},t} \propto \mathbb{P}_{\boldsymbol{\theta},\boldsymbol{\phi}_{k-1}}(x_t \mid \mathbf{X}_{<t})$	shaping	Wang et al., 2025
SYN$_{\mathrm{pool}}$	$\mathcal{R}_{\mathrm{p}}$	$\mathcal{B}_k = \mathcal{R}_{\mathrm{p}}$, ten updates	schedule	control
Question–answer pairs				
QA Fine-Tuning (QAFT)	$\mathcal{R}_{\mathrm{q}}$	–	material	Allen-Zhu & Li, 2024
QAFT$_{\mathrm{lw}}$	$\mathcal{R}_{\mathrm{q}}$	$w_{k,\mathbf{X}} \propto \bar{\ell}(\mathbf{X}; \boldsymbol{\phi}_{k-1})$	shaping	Katharopoulos & Fleuret, 2017
Context Distillation (CD)	$\mathcal{R}_{\mathrm{q}}$	$\boldsymbol{\tau}_{k,\mathbf{X},t} = \mathbb{P}_{\boldsymbol{\theta}}(\cdot \mid \kappa(\mathbf{D}), \mathbf{X}_{<t})$	shaping	Askell et al., 2021; Eyuboglu et al., 2025
Two kinds of material				
Mixed Training (MIX)	$\mathcal{R}_{\mathrm{p}}$, $\mathcal{R}_{\mathrm{q}}$	–	material	Allen-Zhu & Li, 2024
Knowledge Augmentation (KA)	$\mathcal{R}_{\mathrm{p}}$, $\mathcal{R}_{\mathrm{r}}$	–	material	Allen-Zhu & Li, 2024; Ovadia et al., 2024
Passage-Conditioned QA (PQA)	$\mathcal{R}_{\mathrm{p}}$, $\mathcal{R}_{\mathrm{q}}^{\mathbf{D}}$	–	material	adapted from Shi et al., 2024
PQA$_{\mathrm{ans}}$	$\mathcal{R}_{\mathrm{p}}$, $\mathcal{R}_{\mathrm{q}}^{\mathbf{D}}$	passage prefix excluded from $S_{\mathbf{X}}$	loss mask	Shi et al., 2024
PQA$_{\mathrm{pf}}$	$\mathcal{R}_{\mathrm{p}}$, $\mathcal{R}_{\mathrm{q}}^{\mathbf{D}}$	$\lambda_{k,\mathrm{p}} = 1$ for $k \le 6$	shaping	Bengio et al., 2009
Three kinds of material				
Mixed Training, three kinds (MIX3)	$\mathcal{R}_{\mathrm{p}}$, $\mathcal{R}_{\mathrm{q}}$, $\mathcal{R}_{\mathrm{r}}$	–	material	Ours
MIX3$_{\mathrm{unif}}$	$\mathcal{R}_{\mathrm{p}}$, $\mathcal{R}_{\mathrm{q}}$, $\mathcal{R}_{\mathrm{r}}$	$\lambda_{k,\mathrm{m}} \propto \lvert\mathcal{R}_{\mathrm{m}}\rvert$	shaping	control
Rewrite-Conditioned QA (RQA)	$\mathcal{R}_{\mathrm{p}}$, $\mathcal{R}_{\mathrm{q}}^{\mathrm{r}}$	–	material	adapted from Shi et al., 2024

obtained after the last update. Adaptation is thus defined by the endpoint of a finite optimization trajectory rather than by an exact minimizer.

Given only $\mathbf{Q}_i$, without the document or its material in context, the adapted model produces an answer $\widehat{\mathbf{A}}_i$. Following Zweiger et al. (2025), a GPT grader J compares each answer with the reference and returns 1 if it is judged correct and 0 otherwise, and accuracy is the fraction judged correct:

$$\mathrm{Acc}(\mathbf{D}; \Omega, C) = \frac{1}{n_{\mathbf{D}}} \sum_{i=1}^{n_{\mathbf{D}}} J\left(\mathbf{Q}_i, \mathbf{A}_i, \widehat{\mathbf{A}}_i\right). \tag{2}$$

Scope. Two constraints define the setting. The evaluation pairs $\mathcal{Q}(\mathbf{D})$ never enter adaptation, and no external model supplies material or targets. No outer loop meta-trains the adapter initialization, the material generator or the update rule across documents, and the only soft targets we use come from the backbone itself, with the adapter disabled and $\mathbf{D}$ in context, as in context distillation. The GPT grader is used only after adaptation and provides no training signal.

4 Analyzing Objectives

Section 1 raises three questions, which this study is designed to answer: whether the effect of the objective depends on where LoRA is applied and on the backbone, which of its components account for this effect, and how much accuracy the objective alone achieves. Following the setup of Zweiger et al. (2025), we therefore construct objectives that differ in one component at a time and compare them under two placements and on two backbones, keeping all other choices fixed within each comparison. Table 1 lists the fourteen objectives, which §4.1 and §4.2 motivate.

Each objective below is an instance of Eq. (1). We group the material into three kinds, namely the passage material, consisting of the passage and its generated implications, the generated question–answer pairs, and the generated rewrites. We index the kinds by $\mathrm{m} \in \{\mathrm{p}, \mathrm{q}, \mathrm{r}\}$ and write $\mathcal{R}_{\mathrm{m}}$ for the training sequences built from kind m. Most objectives use next-token prediction averaged over the loss mask of each sequence,

$$\bar{\ell}(\mathbf{X}; \boldsymbol{\phi}) = -\frac{1}{|S_{\mathbf{X}}|} \sum_{t \in S_{\mathbf{X}}} \log \mathbb{P}_{\boldsymbol{\theta},\boldsymbol{\phi}}(x_t \mid \mathbf{X}_{<t}). \tag{3}$$

When all training sequences form a single minibatch weighted by kind, Eq. (1) becomes

$$L_k^{\Omega}(\boldsymbol{\phi}) = \sum_{\mathrm{m}} \lambda_{k,\mathrm{m}} \frac{1}{|\mathcal{R}_{\mathrm{m}}|} \sum_{\mathbf{X} \in \mathcal{R}_{\mathrm{m}}} \bar{\ell}(\mathbf{X}; \boldsymbol{\phi}), \qquad \sum_{\mathrm{m}} \lambda_{k,\mathrm{m}} = 1, \tag{4}$$

where $\lambda_{k,\mathrm{m}} \geq 0$ is the weight of kind m at the k-th update.

4.1 Which text should be supervised?

The first component we vary is the training text. How a fact is phrased in the training text determines what a model can answer about it. In pretraining and continued training, a model trained on *A is B* cannot answer the reverse question (Berglund et al., 2024), and a fact is recalled reliably only after it is seen in several phrasings (Allen-Zhu & Li, 2024; Ovadia et al., 2024; Yang et al., 2025). Whether this holds when a single document is written into a low-rank adapter in a few updates is what the configurations below examine. *Train on Passage + Synthetic* (Zweiger et al., 2025) (**SYN**) supervises the passage material $\mathcal{R}_{\mathrm{p}}$ with one update per sequence. The remaining configurations instantiate Eq. (4) with ten updates and draw on the kinds of generated text used in pretraining. Question–answer fine-tuning (**QAFT**; Allen-Zhu & Li, 2024) uses only the pairs $\mathcal{R}_{\mathrm{q}}$, each supervised on its answer; mixed training (**MIX**; Allen-Zhu & Li, 2024) and knowledge augmentation (**KA**; Allen-Zhu & Li, 2024; Ovadia et al., 2024) add $\mathcal{R}_{\mathrm{q}}$ and the fully supervised rewrites $\mathcal{R}_{\mathrm{r}}$ to $\mathcal{R}_{\mathrm{p}}$, respectively; and **MIX3** combines all three kinds with equal weights. $\mathrm{SYN}_{\mathrm{pool}}$ (§4.2), MIX and MIX3 thus add one kind at a time under a fixed number of updates.

Since the phrasing of a fact matters, we also vary how a pair is presented while keeping its kinds of material fixed. Instruction modeling (Shi et al., 2024) applies the loss to the prompt as well as the response and is most effective when prompts are long relative to responses, as for a passage followed by a question–answer pair. Passage-conditioned QA (**PQA**) adopts it by prefixing each pair with the passage ($\mathcal{R}_{\mathrm{q}}^{\mathbf{D}}$) and supervising the full sequence. Because this prefix repeats the same passage before every pair, rewrite-conditioned QA (**RQA**) uses the generated rewrites as prefixes instead, assigning them to the pairs in turn ($\mathcal{R}_{\mathrm{q}}^{\mathrm{r}}$). Both replace $\mathcal{R}_{\mathrm{q}}$ in MIX, so PQA and RQA share their kinds of material with MIX and MIX3, respectively, and differ from them only in how the sequences are built and weighted. PQA against MIX thus tests the effect of the prefix, and RQA against PQA the effect of replacing the repeated passage by rewrites.

4.2 What does the objective add beyond the text?

With the training sequences fixed, the remaining components of Eq. (1) are the loss mask $S_{\mathbf{X}}$, the shaping of the loss, given by the weights $w_{k,\mathbf{X}}$, $c_{k,\mathbf{X},t}$ and the targets $\boldsymbol{\tau}_{k,\mathbf{X},t}$, and the update schedule, given by the minibatches $\mathcal{B}_k$. These components have been studied in other settings, such as instruction tuning (Shi et al., 2024), unlearning (Wang et al., 2025), few-shot reasoning (Akyürek et al., 2025), the compression of long corpora (Eyuboglu et al., 2025) and the training of neural networks in general (Bengio et al., 2009; Katharopoulos & Fleuret, 2017), but their effect on test-time incorporation, in which a low-rank adapter is trained on a single document for a small number of updates, has not been established. We therefore derive each variant from one configuration by changing a single component, as given in the *Change* and *Source* columns of Table 1. Since a variant shares all other choices with the configuration it is derived from, any difference in closed-book accuracy between the two can be attributed to the component that the variant changes.

5 Experiments

5.1 Experimental Setup

Data. We evaluate on the 200 SQuAD (Rajpurkar et al., 2016) passages and 974 questions with which Zweiger et al. (2025) introduced this setting. SQuAD passages are drawn from Wikipedia and may already be familiar to the backbones, we therefore add WikiDYK (Zhang et al., 2026), from which we take 200 facts that are more recent than the training data of either backbone. Each WikiDYK document consists of a fact and the opening paragraph of its Wikipedia entry, and its questions test two abilities. Open questions ask the model to retrieve the fact, and true-or-false statements ask it to judge a claim about the fact. Half of these statements contradict the fact.

Models and material. We use two backbones at different stages of training, the base model Qwen2.5-7B (Yang et al., 2024) used by Zweiger et al. (2025) and the instruction-tuned Llama-3.1-8B-Instruct (Dubey et al., 2024), so that our conclusions are not tied to a single model. Each

Table 2: Closed-book accuracy (%) of the fourteen objectives, grouped as in Table 1, with LoRA applied to the query and value projections, except in the WikiDYK columns for Qwen2.5-7B, where it is applied to all linear projections (see Appendix C.1). WikiDYK reports retrieval. Δ is the paired difference of each variant from its configuration, and * marks a difference exceeding two standard errors. The best objective in each column is in bold. †Learning rate 10^{-4}.

	SQuAD				WikiDYK			
	Qwen2.5-7B		Llama-3.1-8B-Instruct		Qwen2.5-7B		Llama-3.1-8B-Instruct	
Objective	Accuracy	Δ	Accuracy	Δ	Accuracy	Δ	Accuracy	Δ
Passage material								
SYN	53.8		57.8		47.2†		52.0	
SYN_{cw}	32.2	−21.5*	25.5	−32.3*	2.2	−45.0*	7.2	−44.8*
SYN_{pool}	43.8	−9.9*	65.0	+7.2*	40.8	−6.5*	56.5	+4.5
Question–answer pairs								
QAFT	43.2		51.4		18.5		22.8	
QAFT_{lw}	44.6	+1.3	53.4	+2.0	17.8	−0.8	25.5	+2.8*
CD	43.8	+0.5	58.2	+6.7*	18.5	+0.0	25.8	+3.0
Two kinds of material								
MIX	51.5		68.5		51.8		54.2	
KA	49.0		73.2		50.5		63.2	
PQA	47.3		72.7		60.2		58.8	
PQA_{ans}	44.7	−2.6	62.2	−10.5*	44.8	−15.5*	50.8	−8.0*
PQA_{pf}	46.2	−1.1	64.5	−8.2*	48.5	−11.8*	59.5	+0.8
Three kinds of material								
MIX3	**54.2**		72.8		58.5		62.8	
$\text{MIX3}_{\text{unif}}$	53.5	−0.8	73.3	+0.5	**65.0**	+6.5*	63.8	+1.0
RQA	53.6		**75.2**		62.2		**65.0**	
Unadapted	31.0–32.3		39.1–40.0		2.5		4.2	
Open-book	89.4		95.5		92.5		90.2	

backbone generates its own material from the three prompts of Zweiger et al. (2025), with one sample per prompt and document, so that no objective relies on text written by a stronger model.

Adaptation. Following the protocol described by Zweiger et al. (2025), a new LoRA adapter of rank 32 (Hu et al., 2022) is initialized for each document and trained with AdamW (Loshchilov & Hutter, 2019) for ten epochs. To examine the dependence on placement raised in §4, the LoRA is applied either to the query and value projections, as in the original LoRA study (Hu et al., 2022), or to all linear projections. All objectives are trained with a learning rate of 10^{-3}. The only exception is SYN with LoRA applied to all linear projections and in the multi-passage setting, where it collapses at this rate. There we use 10^{-4}, so that SYN is compared at a working rate rather than in a collapsed state (Appendix C.3). We also consider the multi-passage setting of Zweiger et al. (2025), in which a single adapter is trained on the material of all 200 passages.

Evaluation. Following Zweiger et al. (2025), we evaluate closed-book accuracy. The adapted model answers each question without the document in its context, and GPT-4.1 judges whether the answer matches the reference. Accuracy is averaged over documents. Because documents differ considerably in difficulty, objectives are compared on the same documents by the mean of their paired differences and its standard error, and a difference exceeding two standard errors is regarded as significant (marked *). To remove the noise in the unadapted accuracy across evaluations, each gain is measured against the unadapted model of the same evaluation. As an upper reference, we also evaluate the unadapted model with the document in its context, which we call open-book.

5.2 Main Results

As shown in Table 2, the best objective in every column supervises all three kinds of material. These objectives differ only in how the same material is weighted or arranged (see Table 1), and among them only $\text{MIX3}_{\text{unif}}$ on WikiDYK with Qwen2.5-7B differs significantly from MIX3 ($+6.5 \pm 2.2$). Accuracy thus follows the material more closely than the shaping of the loss (see §5.3 and §5.4).

This bound is not a limit of the model, which answers 89.4% of the questions with the passage in its context, and MIX3 answers 223 of the 567 questions that the unadapted model answers only in that condition (counts over questions). A second seed changes SYN, MIX and MIX3 by less than half a point, and on WikiDYK the pooled objectives even collapse under this placement (see Appendix C.1). As shown in Table 3 and Figure 1b, applying LoRA to all linear projections separates the objectives. MIX3 then exceeds SYN by 18.47 ± 2.19 points, whereas SYN and SYN_{pool} change by about one point. We refer to the bound under the default placement as the *adapter plateau*.

A possible cause is the capacity of the adapter, since Biderman et al. (2024) find that LoRA on the attention layers alone underperforms LoRA on all layers, although our experiments do not separate the placement from the number of trainable parameters. The plateau also depends on the backbone. On Llama-3.1-8B-Instruct, the objectives already separate under the default placement, where MIX3 exceeds SYN by 15.03 ± 2.09 points, and under all linear projections SYN matches MIX3 only with one update per sequence and a learning rate of 10^{-4}. Across the four settings on SQuAD, no objective significantly exceeds MIX3, which takes ten updates and no trained generator, teacher or outer loop, and the best objective remains 16 to 35 points below open-book accuracy.

Table 3: Closed-book accuracy (%) on SQuAD with LoRA applied to the query and value projections (q/v) or to all linear projections (all-lin.). SYN takes $10\,|\mathcal{R}_{\text{p}}|$ updates and the others ten. †SYN uses a learning rate of 10^{-4}.

	Qwen2.5-7B		Llama-3.1-8B-Instruct	
Objective	q/v	all-lin.	q/v	all-lin.
SYN	53.8	55.2†	57.8	73.8†
SYN_{pool}	43.8	44.9	65.0	62.8
MIX	51.5	64.8	68.5	69.0
MIX3	54.2	73.7	72.8	73.5
Open-book	89.4		95.5	

5.3 Supervising more distinct text, not more kinds of material

With the number of updates fixed, each kind of material added raises accuracy. SYN_{pool}, MIX and MIX3 supervise one, two and three kinds in ten updates, and under all linear projections each step is significant on both backbones (MIX$-\text{SYN}_{\text{pool}}$ $+19.92 \pm 2.31$ and $+6.24 \pm 2.06$; MIX3$-$MIX $+8.92 \pm 1.75$ and $+4.43 \pm 1.68$; see Table 3). Under the query and value projections the steps keep their sign, but one per backbone is not significant. Pairs alone are not sufficient (QAFT$-$MIX -8.21 ± 1.70 and -17.08 ± 2.01), and which kind to add depends on the backbone model.

Table 4: Token-matched and compute-matched controls on Qwen2.5-7B, with tokens processed per document and epoch, number of updates, and closed-book accuracy (%) on SQuAD. †Learning rate 10^{-4}.

Objective	Tokens	Updates	q/v	all-lin.
SYN	381	$10\,|\mathcal{R}_{\text{p}}|$	53.8	55.2
MIX	694	10	51.5	64.8
MIX3	1,345	10	54.2	73.7
token-matched	690	10	49.0	58.2
five epochs	1,345	5	43.3	68.5

Each added kind also adds supervised text, from 694 tokens per document and epoch for MIX to 1,345 for MIX3 (see Table 4). A token-matched version of MIX3 on Qwen2.5-7B, which subsamples the pairs and rewrites to the token count of MIX, does not exceed MIX (-2.42 ± 1.62 and -6.57 ± 1.94 under the two placements), whereas MIX3 trained for five epochs on its full material, which matches the total tokens of the token-matched version, reaches 68.5% against 58.2% under all linear projections. The gain of MIX3 thus follows the amount of distinct text it supervises. This agrees with pretraining (Allen-Zhu & Li, 2024) and fine-tuning (Ovadia et al., 2024), but here it appears only under a placement that does not bound the objectives.

5.4 Beyond the text: loss mask and shaping

The loss mask matters through how much of the passage it covers. PQA_{ans}, which excludes the passage prefix from the loss, loses 10.49 ± 1.69 points on Llama-3.1-8B-Instruct and 8.00 ± 2.13 on Qwen2.5-7B under all linear projections, consistent with instruction modeling (Shi et al., 2024). As shown in Figure 1a, removing a random fraction ρ of the passage tokens from the loss of SYN lowers the gain monotonically from 14.9 points at $\rho = 0$ to -3.7 at $\rho = 0.75$, whereas removing question tokens from the pairs changes it by no more than about two standard errors. *More details are provided in Appendix D.1.*

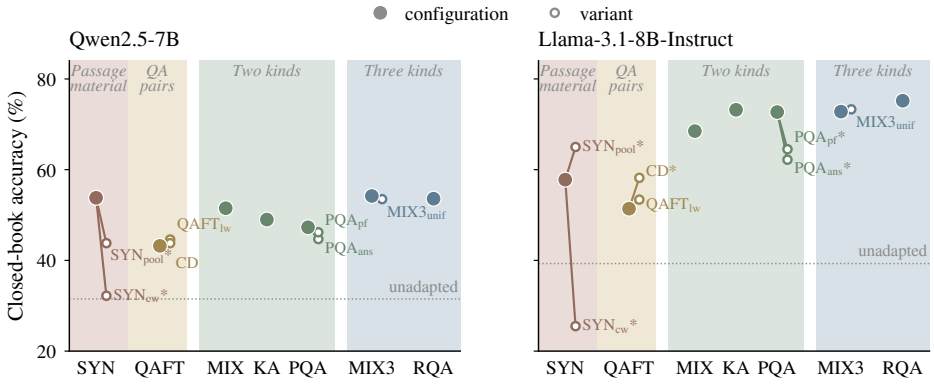

Figure 2: The SQuAD accuracies of Table 2. Filled markers denote configurations, and open markers joined to them denote their variants. A star marks a significant difference, and the dotted line marks the unadapted model.

Shaping the loss rarely helps (see Figure 2). Of the ten shaping contrasts on SQuAD, only CD on Llama-3.1-8B-Instruct improves accuracy significantly ($+6.73 \pm 1.81$), and on WikiDYK only QAFT_{lw} on Llama-3.1-8B-Instruct ($+2.8 \pm 1.3$) and $\text{MIX3}_{\text{unif}}$ on Qwen2.5-7B ($+6.5 \pm 2.2$). The curriculum of PQA_{pf} lowers accuracy on Llama-3.1-8B-Instruct (-8.15 ± 1.79) and on Qwen2.5-7B under all linear projections (-8.83 ± 1.97). SYN_{cw} lowers it in every column by 21.5 to 45.0 points, presumably because its weights, which protect tokens of low probability under gradient ascent in unlearning (Wang et al., 2025), suppress under gradient descent the tokens not yet learned.

Table 5: Multi-passage setting on SQuAD, with one adapter (LoRA on the query and value projections) trained on all 200 passages. Pooled objectives take 2,000 updates. †16,010 and 18,760 updates at a learning rate of 10^{-4}.

	Qwen2.5-7B		Llama-3.1-8B-Instruct	
Objective	Accuracy	Gain	Accuracy	Gain
SYN†	36.5	+5.0	43.7	+4.2
SYN_{pool}	25.5	−6.3	36.2	−3.7
MIX	36.8	+4.7	43.7	+4.1
KA	38.1	+6.4	44.2	+4.9
MIX3	43.3	+11.8	54.5	+14.6
$\text{MIX3}_{\text{unif}}$	43.5	+11.8	48.1	+8.8
RQA	40.7	+8.7	56.2	+16.5

5.5 Rearranging the same text

Rearranging the same text moves accuracy by as much as eleven points, in a direction that depends on the backbone and the placement. PQA prefixes each pair with the passage instead of training the pair as a separate sequence, and SYN_{pool} groups all sequences into one minibatch, which reduces the updates from $10\,|\mathcal{R}_{\text{p}}|$ to ten. Under the query and value projections, both lower accuracy on Qwen2.5-7B and raise it on Llama-3.1-8B-Instruct (PQA−MIX -4.18 ± 1.59 and $+4.15 \pm 1.94$; SYN_{pool}−SYN -9.93 ± 1.99 and $+7.16 \pm 2.11$). Under all linear projections, SYN_{pool} falls below SYN on both backbones (-10.37 ± 2.29 and -11.02 ± 1.87) and PQA−MIX on Qwen2.5-7B is no longer significant (-2.39 ± 2.02).

Where the rewrites are placed makes no significant difference (RQA−MIX3 -0.61 ± 1.58 and $+2.34 \pm 1.74$). The only contrast with the same sign on both backbones, RQA−PQA ($+6.35 \pm 1.60$ and $+2.51 \pm 1.61$), replaces the repeated passage prefix with distinct rewrites and thus changes the content, in line with §5.3. Since the arrangement alone moves accuracy by as much as eleven points, the arrangement and the number of updates should be controlled when objectives are compared.

5.6 A simple configuration

Sections 5.3 and 5.4 suggest a simple principle, which is to keep the passage fully supervised and to add distinct self-generated text. MIX3 is its most direct instance. It supervises the passage together with the model's own question–answer pairs and rewrites with equal weights, takes ten updates at a learning rate of 10^{-3}, and uses material that the backbone generates once from the prompts of

Table 6: Accuracy (%) on WikiDYK in judging true-or-false statements about the fact (Jud.), averaged over statements consistent with the fact and statements that contradict it, and on the contradicting statements alone (CF). Placements follow Table 2. Single runs, without paired standard errors. † denotes that the learning rate is 10^{-4} for a fair comparison.

	Qwen2.5-7B		Llama-3.1-8B-It			Qwen2.5-7B		Llama-3.1-8B-It	
Objective	Jud.	CF	Jud.	CF	Objective	Jud.	CF	Jud.	CF
SYN	53.5†	31.0†	33.2	24.5	PQA	42.5	23.0	56.0	22.0
SYN_{cw}	5.8	6.0	35.2	29.5	PQA_{ans}	43.0	22.5	55.0	22.5
SYN_{pool}	38.2	36.5	54.8	29.0	PQA_{pf}	49.2	30.5	56.0	27.5
QAFT	51.5	22.5	55.5	18.0	MIX3	51.5	17.5	56.0	27.0
QAFT_{lw}	51.5	25.5	54.0	16.5	$\text{MIX3}_{\text{unif}}$	49.8	19.0	57.2	24.5
CD	48.0	22.0	55.2	19.0	RQA	46.2	27.0	56.0	30.0
MIX	49.8	20.5	55.0	21.5	Unadapted	47.5	32.5	55.8	35.5
KA	37.8	31.0	56.2	27.5	Open-book	88.5	97.5	91.0	98.0

Zweiger et al. (2025), whose generator is trained by reinforcement learning in an outer loop. No objective significantly exceeds MIX3 on SQuAD (§5.2).

The advantage carries over to the multi-passage setting, in which a single adapter is trained on the material of all 200 passages (see Table 5). On both backbones, the three objectives that draw on three kinds of material gain 8.7 to 16.5 points over the unadapted model, against at most 6.4 for MIX, KA and SYN, whereas SYN_{pool} falls below the unadapted model. The best objective differs between the backbones, $\text{MIX3}_{\text{unif}}$ and MIX3 on Qwen2.5-7B and RQA on Llama-3.1-8B-Instruct, but all three follow the same principle.

5.7 Retrieving a fact versus judging claims about it

On WikiDYK, retrieval follows the pattern of SQuAD (see Table 2), but judgment does not follow retrieval (see Table 6 and Figure 1c). With the document in its context, the model judges about 90% of the statements correctly, yet after incorporation most objectives remain near the unadapted model and several fall well below it although they retrieve the fact. On Llama-3.1-8B-Instruct, SYN retrieves 52.0% of the facts but judges 33.2% of the statements, 22.6 points below the unadapted model, whereas on Qwen2.5-7B SYN keeps its judgment and PQA and KA lose it, so which objectives lose judgment depends on the backbone.

Statements that contradict the fact are the hardest, and on Llama-3.1-8B-Instruct every objective judges them below the unadapted model. Among the objectives that supervise the passage, only MIX, PQA_{pf}, MIX3 and $\text{MIX3}_{\text{unif}}$ fall no more than one point below the unadapted model on both backbones, and MIX3 does so while retrieving the fact in 58.5% and 62.8% of the open questions. Closed-book accuracy therefore does not establish that the model can use an incorporated fact to judge claims about it, and the objective should be chosen with both abilities in view.

6 Conclusion

We isolated the training objective of test-time knowledge incorporation, the one component that every method shares and that can be changed without additional training, by decomposing it into four components and comparing fourteen objectives that change one component at a time. We find that shaping of the loss, where most design effort goes, rarely improves accuracy, and in the standard setting no objective significantly exceeds next-token training, not because the objectives are equivalent but because the default placement of LoRA masks their differences. What matters is how much distinct text the loss covers. Following this principle, MIX3 simply adds the model's own question–answer pairs and rewrites to the fully supervised passage, and with ten updates and no trained generator, teacher or outer loop it can exceed next-token training by up to 18.5 points, reaching 73.7% and 73.5% on the two backbones while preserving judgment on new facts. Components that require training of their own should therefore be evaluated against this objective, and objectives should be compared under more than one placement.

AI USE STATEMENT

In this work, we used generative AI tools to assist with language polishing and improving the clarity and readability of the manuscript. We have reviewed and revised all AI-assisted text and take responsibility for the final content of this work.

ETHICS STATEMENT

This work uses two public datasets, SQuAD and WikiDYK, both derived from Wikipedia, and involves no human subjects or personal data. The adaptation material is generated by the backbones themselves and may contain inaccurate statements, which the adapted models can absorb. Our results show that an adapted model may retrieve a fact while judging claims about it incorrectly, and we caution against treating closed-book accuracy alone as evidence that incorporated knowledge is reliable. GPT-4.1 is used only to grade answers after adaptation. We foresee no direct harmful application of the findings.

REPRODUCIBILITY STATEMENT

All objectives are defined in §4 and specified component by component in Appendix B.2, with the formal setting in Appendix B.1. The data, including the construction of the WikiDYK documents and questions, are described in §5.1 and Appendix C.1. The prompts, decoding parameters and parsers used to generate the material are given verbatim in Appendix C.2, and the training, evaluation and learning-rate settings in Appendix C.3. Appendix C.4 lists the settings in which each objective was run, and Appendices C.5 to D.4 give the details of the multi-passage, token-matched, overlap and protocol experiments. Every comparison is a paired difference over the same documents with its standard error. Source code and the generated material will be released.

REFERENCES

Ekin Akyürek, Mehul Damani, Adam Zweiger, et al. The surprising effectiveness of test-time training for few-shot learning. In *Proc. of ICML*, 2025.

Zeyuan Allen-Zhu and Yuanzhi Li. Physics of language models: Part 3.1, knowledge storage and extraction. In *Proc. of ICML*, 2024.

Amanda Askell, Yuntao Bai, Anna Chen, et al. A general language assistant as a laboratory for alignment, 2021. arXiv preprint arXiv:2112.00861.

Rachit Bansal, Aston Zhang, Rishabh Tiwari, et al. Let's (not) just put things in context: Test-time training for long-context LLMs. In *Proc. of ICLR*, 2026.

Yoshua Bengio, Jérôme Louradour, Ronan Collobert, et al. Curriculum learning. In *Proc. of ICML*, pp. 41–48, 2009.

Lukas Berglund, Meg Tong, Maximilian Kaufmann, et al. The reversal curse: LLMs trained on "a is b" fail to learn "b is a". In *Proc. of ICLR*, 2024.

Dan Biderman, Jacob P. Portes, Jose Javier Gonzalez Ortiz, et al. Lora learns less and forgets less. *Trans. Mach. Learn. Res.*, 2024.

Zeming Chen, Gail Weiss, Eric Mitchell, et al. RECKONING: reasoning through dynamic knowledge encoding. In *Proc. of NeurIPS*, 2023.

Zeming Chen, Angelika Romanou, Gail Weiss, et al. PERK: Long-context reasoning as parameter-efficient test-time learning. In *Proc. of ICLR*, 2026.

Abhimanyu Dubey, Abhinav Jauhri, Abhinav Pandey, et al. The llama 3 herd of models. 2024.

Sabri Eyuboglu, Ryan Saul Ehrlich, Simran Arora, et al. Cartridges: Lightweight and general-purpose long context representations via self-study. In *ES-FoMo III: 3rd Workshop on Efficient Systems for Foundation Models*, 2025.

Moritz Hardt and Yu Sun. Test-time training on nearest neighbors for large language models. In *Proc. of ICLR*, 2024.

Edward J. Hu, Yelong Shen, Phillip Wallis, et al. Lora: Low-rank adaptation of large language models. In *Proc. of ICLR*, 2022.

Angelos Katharopoulos and François Fleuret. Biased importance sampling for deep neural network training, 2017. arXiv preprint arXiv:1706.00043.

Yuri Kuratov, Matvey Kairov, Aydar Bulatov, et al. Gradmem: Learning to write context into memory with test-time gradient descent. In *Third Workshop on Test-Time Updates (Main Track)*, 2026.

Ilya Loshchilov and Frank Hutter. Decoupled weight decay regularization. In *Proc. of ICLR*, 2019.

Oded Ovadia, Menachem Brief, Moshik Mishaeli, et al. Fine-tuning or retrieval? comparing knowledge injection in LLMs. In *Proc. of EMNLP*, pp. 237–250, 2024.

Pranav Rajpurkar, Jian Zhang, Konstantin Lopyrev, et al. SQuAD: 100,000+ questions for machine comprehension of text. In *Proc. of EMNLP*, pp. 2383–2392, 2016.

Zhengxiang Shi, Adam X. Yang, Bin Wu, et al. Instruction tuning with loss over instructions. In *Proc. of NeurIPS*, 2024.

Yu Sun, Xiaolong Wang, Zhuang Liu, et al. Test-time training with self-supervision for generalization under distribution shifts. In *Proc. of ICML*, pp. 9229–9248, 2020.

Qizhou Wang, Jin Peng Zhou, Zhanke Zhou, et al. Rethinking LLM unlearning objectives: A gradient perspective and go beyond. In *Proc. of ICLR*, 2025.

Zekun Wang, Anant Gupta, Zi-Han Dong, et al. Self-consolidating language models: Continual knowledge incorporation from context, 2026. arXiv preprint arXiv:2605.07076.

An Yang, Baosong Yang, Beichen Zhang, et al. Qwen2.5 technical report, 2024. arXiv preprint arXiv:2412.15115.

Zitong Yang, Neil Band, Shuangping Li, et al. Synthetic continued pretraining. In *Proc. of ICLR*, 2025.

Yuwei Zhang, Wenhao Yu, Shangbin Feng, et al. Bidirectional LMs are better knowledge memorizers? a benchmark for real-world knowledge injection. In *Proc. of ACL*, pp. 16964–16980, 2026.

Adam Zweiger, Jyothish Pari, Han Guo, et al. Self-adapting language models. In *Proc. of NeurIPS*, 2025.

A Extended Related Work

Test-time knowledge incorporation. Test-time training adapts model weights on data available at test time (Sun et al., 2020), and has been extended to language models by optimizing an autoregressive loss on retrieved neighbors or on the context itself (Hardt & Sun, 2024; Bansal et al., 2026). Knowledge incorporation removes the document from the context after adaptation, so that the adapted parameters must carry its content. Several methods meta-learn how to write a document into the weights (Chen et al., 2023; 2026) or into a set of memory tokens (Kuratov et al., 2026). Eyuboglu et al. (2025) train a key-value cache on synthetic conversations about corpora of tens to hundreds of thousands of tokens, and find that distilling a model that reads the corpus outperforms next-token training on the same synthetic data. Zweiger et al. (2025) train on the passage and generated implications with next-token prediction, and train the generator of these implications with reinforcement learning in an outer loop. In each case, the objective is evaluated together with a component that requires its own training, whereas we fix every such component and vary only the objective.

Training text and loss for knowledge. In pretraining on synthetic biographies, Allen-Zhu & Li (2024) show that knowledge becomes extractable when question–answer pairs are mixed into the training data or each biography is rewritten several times. Ovadia et al. (2024) find that accuracy on new facts rises with the number of paraphrases in unsupervised continued training on Wikipedia articles, and Yang et al. (2025) synthesize text relating the entities of a small corpus to make continued pretraining on it effective. These studies train on document collections, whereas we adapt a low-rank adapter to a single document in a few updates. Which tokens receive the loss has been studied separately. Shi et al. (2024) show that supervising the prompt as well as the response helps most when prompts are long relative to responses, and Akyürek et al. (2025) compare losses on demonstrations and on outputs in test-time training for few-shot reasoning. Our loss-mask variants and dose-response curves (§5.4, Appendix D.1) examine this question for knowledge incorporation.

Where LoRA is applied. Hu et al. (2022) compare combinations of attention matrices under a fixed budget of trainable parameters, find that adapting the query and value projections works best among them, and leave the MLP layers to future work. Biderman et al. (2024) find that LoRA learns less than full fine-tuning and that, in instruction fine-tuning on code, applying it only to the attention modules underperforms applying it to the MLP modules as well. Wang et al. (2026) learn which layers to update for each context. These studies fix the training objective, whereas studies of the objective fix the placement. Our comparison under two placements shows that the placement can mask differences between objectives and reverse the effect of individual components (§5.2, §5.5).

Using incorporated knowledge. Knowledge that a model can retrieve in one form is not necessarily available in another. A model trained on a fact stated in one direction fails to answer it in the reverse direction (Berglund et al., 2024). The evaluations of Zweiger et al. (2025) and Ovadia et al. (2024) measure closed-book accuracy on questions about the incorporated documents. We add true-or-false statements about facts more recent than the training data, including statements that contradict them, and find that retrieval and judgment can diverge (§5.7).

B Definitions of the Objectives

B.1 Formal details for §3

This appendix states in full the definitions summarized in §3. It introduces no new objects and makes the construction and typing of those used in the main text explicit.

Adaptation configuration. The configuration C specifies the placement, rank, scaling and dropout of the LoRA adapter, together with the optimizer, learning-rate schedule, weight decay, gradient clipping, numerical precision and random seed. Different placements induce different parameter spaces, so we write Φ_C for the adapter parameter space under C and $\phi_0^C \in \Phi_C$ for the initialization, chosen so that

$$\mathbb{P}_{\boldsymbol{\theta},\boldsymbol{\phi}_0^C}(\cdot \mid \mathbf{X}) = \mathbb{P}_{\boldsymbol{\theta}}(\cdot \mid \mathbf{X}) \qquad \text{for every token sequence } \mathbf{X}.$$

The main text omits the superscript C on $\boldsymbol{\phi}_0$, since C is held fixed whenever objectives are compared.

Material. For each kind of material $\mathrm{m} \in \{\mathrm{p}, \mathrm{q}, \mathrm{r}\}$, a fixed prompt template π_{m} maps the document to a complete model input, a sampling procedure Gen with the decoding parameters of Appendix C.2 produces one continuation from the unadapted model, and a fixed parser g_{m} splits that continuation into an indexed sequence of text items,

$$\mathcal{G}_{\mathrm{m}}(\mathbf{D}) := g_{\mathrm{m}}(\mathrm{Gen}(\mathbb{P}_{\boldsymbol{\theta}}; \pi_{\mathrm{m}}(\mathbf{D}))) .$$

The three kinds of material are

$$\mathcal{M}_{\mathrm{p}}(\mathbf{D}) := (\mathbf{D}) \,\|\, \mathcal{G}_{\mathrm{p}}(\mathbf{D}), \qquad \mathcal{M}_{\mathrm{q}}(\mathbf{D}) := \mathcal{G}_{\mathrm{q}}(\mathbf{D}), \qquad \mathcal{M}_{\mathrm{r}}(\mathbf{D}) := \mathcal{G}_{\mathrm{r}}(\mathbf{D}),$$

where $\|$ denotes concatenation of indexed sequences, so that $\mathcal{M}_{\mathrm{p}}$ holds the document and its implications, $\mathcal{M}_{\mathrm{q}}$ the question–answer pairs and $\mathcal{M}_{\mathrm{r}}$ the rewrites. Only $\mathcal{M}_{\mathrm{p}}$ contains the document itself. Generation is performed once per document with the unadapted model, and every objective is trained on the resulting material.

Objectives. An objective Ω maps the material, the document and the frozen backbone to a set of training sequences with their loss masks and to a sequence of minibatches with their weights and targets,

$$\Omega : \big(\mathcal{M}(\mathbf{D}),\, \mathbf{D},\, \mathbb{P}_{\boldsymbol{\theta}}\big) \longmapsto \Big(\mathcal{R}_{\Omega}(\mathbf{D}),\; \big(\mathcal{B}_k,\, w_k,\, c_k,\, \boldsymbol{\tau}_k\big)_{k=1}^{K_{\Omega}(\mathbf{D})}\Big),$$

where $\mathcal{M}(\mathbf{D})$ denotes the three kinds together, each $\mathbf{X} \in \mathcal{R}_{\Omega}(\mathbf{D})$ carries a loss mask $S_{\mathbf{X}} \subseteq \{2, \ldots, |\mathbf{X}|\}$, and $\mathcal{B}_k \subseteq \mathcal{R}_{\Omega}(\mathbf{D})$. The document and the frozen backbone appear in the domain because context-distillation targets are computed from $\mathbb{P}_{\boldsymbol{\theta}}$ conditioned on $\mathbf{D}$. The first position of every sequence is excluded from $S_{\mathbf{X}}$ because it has no preceding context.

The sequence and token weights may be combined into a single weight per supervised token, $\alpha_{k,\mathbf{X},t} := w_{k,\mathbf{X}}\, c_{k,\mathbf{X},t}$, which satisfies $\alpha_{k,\mathbf{X},t} \geq 0$ and $\sum_{\mathbf{X} \in \mathcal{B}_k} \sum_{t \in S_{\mathbf{X}}} \alpha_{k,\mathbf{X},t} = 1$. The factorized form is used in the main text because several objectives are defined by how they weight whole sequences.

Per-sequence objectives take each minibatch to be a single sequence and pooled objectives take it to be all of $\mathcal{R}_{\Omega}(\mathbf{D})$. Under a fixed number of epochs E, the former take $K_{\Omega}(\mathbf{D}) = E\,|\mathcal{R}_{\Omega}(\mathbf{D})|$ updates and the latter $K_{\Omega}(\mathbf{D}) = E$. Among the fourteen objectives, only SYN and $\mathrm{SYN}_{\mathrm{cw}}$ are per-sequence.

Weights may change with k or with the current adapter. In $\mathrm{PQA}_{\mathrm{pf}}$ the sequence weights depend on k. In $\mathrm{SYN}_{\mathrm{cw}}$ the token weights, and in $\mathrm{QAFT}_{\mathrm{lw}}$ the sequence weights, are computed from the model at $\boldsymbol{\phi}_{k-1}$. These quantities carry a stop-gradient, so that Eq. (1) is differentiated with w_k, c_k and $\boldsymbol{\tau}_k$ held fixed.

Finite adaptation. Let $\mathbf{s}_k$ denote the optimizer state and $U_{C,k}$ one step of the optimizer specified by C. Starting from $(\boldsymbol{\phi}_0^C, \mathbf{s}_0)$, adaptation follows

$$\mathbf{g}_k = \nabla_{\boldsymbol{\phi}} L_k^{\Omega}(\boldsymbol{\phi})\big|_{\boldsymbol{\phi}=\boldsymbol{\phi}_{k-1}},$$
$$(\boldsymbol{\phi}_k, \mathbf{s}_k) = U_{C,k}(\boldsymbol{\phi}_{k-1}, \mathbf{s}_{k-1}, \mathbf{g}_k), \qquad k = 1, \ldots, K_{\Omega}(\mathbf{D}),$$

and the adapter returned for $\mathbf{D}$ is $\boldsymbol{\phi}^{\Omega,C}(\mathbf{D}) = \boldsymbol{\phi}_{K_{\Omega}(\mathbf{D})}$.

Evaluation. Let $h(\mathbf{Q}_i)$ be the fixed evaluation prompt, which contains neither the document nor its material, and Dec greedy decoding with at most 64 new tokens. The answer in Eq. (2) is

$$\widehat{\mathbf{A}}_i := \mathrm{Dec}\big(\mathbb{P}_{\boldsymbol{\theta},\boldsymbol{\phi}^{\Omega,C}(\mathbf{D})}; h(\mathbf{Q}_i)\big),$$

and the grader J maps a question, a reference answer and a predicted answer to $\{0, 1\}$. For an evaluation corpus $\mathcal{D}$, the single-passage score is the document-macro average

$$\mathrm{Score}_{\mathrm{single}}(\Omega, C) = \frac{1}{|\mathcal{D}|} \sum_{\mathbf{D} \in \mathcal{D}} \mathrm{Acc}(\mathbf{D}; \Omega, C).$$

In the multi-passage setting, an ordered collection $\mathcal{P} = (\mathbf{D}_1, \ldots, \mathbf{D}_m)$ is adapted jointly. The objective is applied to the combined material $\|_{\ell=1}^{m} \mathcal{M}(\mathbf{D}_\ell)$, yielding one shared adapter $\boldsymbol{\phi}^{\Omega,C}(\mathcal{P})$, and the score is the question-micro average

$$\mathrm{Score}_{\mathrm{multi}}(\mathcal{P}; \Omega, C) = \frac{1}{|\mathcal{I}(\mathcal{P})|} \sum_{(\ell,i) \in \mathcal{I}(\mathcal{P})} J\Big(\mathbf{Q}_{\ell,i}, \mathbf{A}_{\ell,i}, \widehat{\mathbf{A}}_{\ell,i}\Big),$$

where $\widehat{\mathbf{A}}_{\ell,i}$ is produced by the shared adapter and $\mathcal{I}(\mathcal{P})$ is the set of all document–question indices of $\mathcal{P}$.

Cross-entropy and KL divergence. Eq. (1) is written with cross-entropy. For soft targets our implementation minimizes $\mathrm{KL}\big(\boldsymbol{\tau}_{k,\mathbf{X},t} \,\big\|\, \mathbb{P}_{\boldsymbol{\theta},\boldsymbol{\phi}}(\cdot \mid \mathbf{X}_{<t})\big)$, which differs from the cross-entropy by the entropy of the target. Since targets carry a stop-gradient, that term is constant in $\boldsymbol{\phi}$ and the two induce the same update. Cross-entropy is used in the main text because it is linear in the target, so an objective that interpolates between a one-hot target and a teacher distribution is again of the form of Eq. (1), with $\boldsymbol{\tau}_{k,\mathbf{X},t}$ the corresponding mixture.

B.2 Component form of each objective

This appendix gives every objective of Table 1 as a choice of the four components of Eq. (1), together with the implementation details that affect the loss. Table 7 maps the names used in the paper to the identifiers in our code.

Training sequences. For a document $\mathbf{D}$, a *passage sequence* is one element of $\mathcal{M}_{\mathrm{p}}(\mathbf{D})$, namely one generated implication or the document itself, each prefixed by the title of the document. A *pair sequence* is `Question:` $\mathbf{Q}$`\nAnswer:` $\mathbf{A}$ built from a generated pair, without title. The template up to and including `Answer:` is never supervised, so the *answer positions* of a pair sequence are the positions of $\mathbf{A}$. A *rewrite sequence* is one generated rewrite, without title. A *passage-prefixed pair* ($\mathcal{R}_{\mathrm{q}}^{\mathrm{D}}$) is $\mathbf{D}$, two newlines, then the pair sequence. A *rewrite-prefixed pair* ($\mathcal{R}_{\mathrm{q}}^{\mathrm{r}}$) prefixes the i-th pair with the $(i \bmod n_{\mathrm{r}})$-th of the n_{r} rewrites, truncated to 512 tokens, and falls back to $\mathbf{D}$ for the documents in which no rewrite was parsed. All sequences are truncated to 2,048 tokens.

Conventions. The loss at position t predicts x_t from $\mathbf{X}_{<t}$, so the first position of a sequence is never supervised and a *full* loss mask means $S_{\mathbf{X}} = \{2, \ldots, |\mathbf{X}|\}$. Unless stated otherwise, the target is one-hot, the token weights are uniform over the loss mask, and the loss of a sequence is $\bar{\ell}$ of Eq. (3). Per-sequence objectives take one update per sequence and pooled objectives one update per epoch, so that with ten epochs the former take about eighty updates per document and the latter ten. When a kind of material is absent for a document, its weight in Eq. (4) is divided among the kinds present. Any weight computed from the current adapter is held constant in the gradient. During training the losses of a minibatch are accumulated one sequence at a time, which we verified to be equivalent to a single backward pass over the minibatch to within 1.5×10^{-8} in gradient norm on eight objectives.

Table 7: Names used in the paper and the corresponding identifiers in our code.

Objective	Identifier	Objective	Identifier
SYN	`ar`	PQA	`anch`
$\mathrm{SYN}_{\mathrm{cw}}$	`wga_ar`	$\mathrm{PQA}_{\mathrm{ans}}$	`anchq`
$\mathrm{SYN}_{\mathrm{pool}}$	`seg1`	$\mathrm{PQA}_{\mathrm{pf}}$	`smix`
QAFT	`distill`	MIX3	`dmix`
$\mathrm{QAFT}_{\mathrm{lw}}$	`lw_distill`	$\mathrm{MIX3}_{\mathrm{unif}}$	`dmixp`
CD	`kd_distill`	RQA	`danch`
MIX	`mix`	Open-book	`openbook`
KA	`para_ar`		

Passage material. SYN trains on the passage sequences with a full loss mask and one update per sequence. This is next-token prediction on the passage and its implications, the *Train on Passage + Synthetic Data* baseline of Zweiger et al. (2025). $\mathrm{SYN}_{\mathrm{cw}}$ is SYN with the token weights $c_{k,\mathbf{X},t} \propto \mathbb{P}_{\boldsymbol{\theta},\boldsymbol{\phi}_{k-1}}(x_t \mid \mathbf{X}_{<t})$, the weighting of Wang et al. (2025) at $\alpha = 1$, normalized within the sequence. $\mathrm{SYN}_{\mathrm{pool}}$ is SYN with all passage sequences in one minibatch, weighted uniformly, so that it takes ten updates on the same sequences as SYN.

Question–answer pairs. QAFT trains on the pair sequences with the loss mask restricted to the answer positions, uniform sequence weights and pooled minibatches. $\mathrm{QAFT}_{\mathrm{lw}}$ is QAFT with the sequence weights $w_{k,\mathbf{X}} \propto \bar{\ell}(\mathbf{X}; \boldsymbol{\phi}_{k-1})$, so that pairs with a higher loss receive more weight. CD is QAFT with the target at each answer position replaced by $\boldsymbol{\tau}_{k,\mathbf{X},t} = \mathbb{P}_{\boldsymbol{\theta}}(\cdot \mid \kappa(\mathbf{D}), \mathbf{X}_{<t})$, where $\kappa(\mathbf{D})$ is the prefix `Context:` $\mathbf{D}$ followed by two newlines. The teacher is the same model with the adapter disabled and no gradient, as in the context distillation of Askell et al. (2021).

Two kinds of material. MIX trains on the passage sequences with a full loss mask and on the pair sequences with the answer positions, each kind receiving half of the weight, shared uniformly among its sequences. KA trains on the passage and rewrite sequences with a full loss mask and uniform weights over all sequences. PQA trains on the passage sequences and the passage-prefixed pairs, both with a full loss mask, so that the passage prefix, the question and the answer are all supervised, and each kind receives half of the weight. PQA_{ans} is PQA with the loss mask of the prefixed pairs restricted to the question and the answer, which leaves the passage prefix unsupervised. PQA_{pf} is PQA with a schedule on the weights, under which only the passage sequences receive weight for the first six updates and each kind receives half for the remaining four.

Three kinds of material. MIX3 trains on the passage, pair and rewrite sequences, with a full loss mask for the passage and rewrite sequences and the answer positions for the pairs, and each kind receives one third of the weight, shared uniformly among its sequences. $\text{MIX3}_{\text{unif}}$ is MIX3 with uniform weights over all sequences, so that each kind receives a weight proportional to its number of sequences. RQA is PQA with the passage-prefixed pairs replaced by rewrite-prefixed pairs. Under the weighting of MIX and MIX3, SYN_{pool}, MIX and MIX3 differ only in the kinds of material they train on.

Reference objectives outside the fourteen. Two further objectives were run and are reported only here. The first is SYN trained through the released trainer of Zweiger et al. (2025), in which sequences are padded to 2,048 tokens with the end-of-sequence token and the padded positions are included in the loss mask. On Qwen2.5-7B this trains the adapter to emit end-of-sequence and produces empty answers on 13.4% of the questions, and removing the padded positions from the loss mask raises accuracy from 42.0% to 53.7% ($+11.65 \pm 2.33$). On Llama-3.1-8B-Instruct the same change is within noise ($+0.70 \pm 2.14$). With the padded positions removed, this trainer and our implementation differ by -0.09 ± 1.94 and $+2.65 \pm 2.12$ points on the two backbones. The second is SYN with the token weights raised to ten at the positions of a reference answer found verbatim in the sequence. Such a match occurred for 8.3% of the answers, so the intended mechanism was largely inactive, and the paired difference from SYN is within two standard errors on both backbones ($+1.9 \pm 1.8$ and -1.7 ± 1.9).

Interpolating objectives used in the dose-response curves. Three objectives interpolate between the objectives above and are used in Appendix D.1. *Passage masking* removes a random fraction ρ of the supervised positions of each passage sequence from the loss mask of SYN, so that $\rho = 0$ recovers SYN. *Question masking* starts from pair sequences whose questions and answers are both supervised and removes a fraction ρ of the question positions, so that $\rho = 1$ recovers QAFT. *Distillation mixing* replaces the loss of QAFT by $(1 - \lambda)$ times the one-hot loss plus λ times the loss of CD, so that $\lambda = 1$ recovers CD.

C Experimental Details

C.1 WikiDYK

Construction. WikiDYK (Zhang et al., 2026) collects facts from the *Did you know* section of Wikipedia together with questions about them. We take 200 facts dated January to April 2025, later than the training data of both backbones, from its test split. Each document consists of the fact followed by the opening paragraph of the Wikipedia entry of its bold entity, truncated to 150 words (median 134). The dataset provides, for each fact, a *reliability* question that asks for the fact directly, a *paraphrase* question that asks for it in other words, a *factual* statement consistent with the fact and a *counterfactual* statement that contradicts it. Retrieval accuracy is the mean over the two open questions and judgment accuracy the mean over the two statements, so that half of the statements contradict the fact. The 800 questions contain no duplicate texts. The evaluation prompt of Zweiger et al. (2025) names the document in a topic line, and using the entity as the document title leaks the answer of the reliability question, so the title is a fixed placeholder. On Llama-3.1-8B-Instruct, replacing one placeholder by another changes the accuracy of MIX and MIX3 by at most 1.8 points. Material is generated and adapters are trained exactly as on SQuAD, and the unadapted model retrieves 2.5% and 4.2% of the facts on the two backbones.

Placement on Qwen2.5-7B. Under the query and value projections, the objectives that pool material fail on WikiDYK with Qwen2.5-7B. SYN retrieves 50.8% of the facts, whereas MIX retrieves 16.0%

and MIX3 17.2%, and their answers take the form of the fact with the wrong entity. On SQuAD, whose passages the backbone has likely seen in pretraining, the same objectives reach 51.5% and 54.2% under this placement (see Table 2). With LoRA applied to all linear projections, MIX and MIX3 retrieve 51.8% and 58.5% of the facts, so we report this placement for Qwen2.5-7B in Tables 2 and 6. Figure 3 plots retrieval against judgment for all fourteen objectives on both backbones.

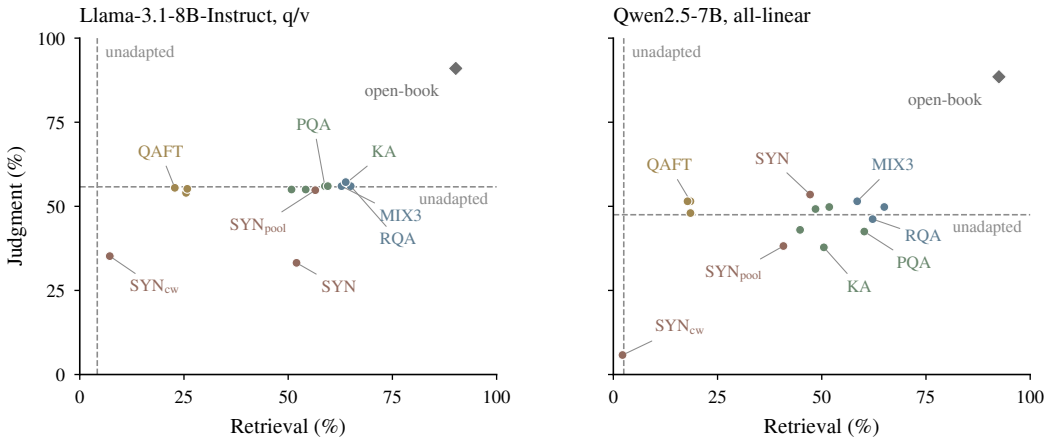

Figure 3: Retrieval against judgment on WikiDYK for the fourteen objectives, on Llama-3.1-8B-Instruct with LoRA applied to the query and value projections and on Qwen2.5-7B with LoRA applied to all linear projections. Dashed lines mark the unadapted model. Colors follow Table 1.

C.2 Material generation

The three kinds of material are generated with the three prompts released by Zweiger et al. (2025), reproduced below, which ask for implications of the passage, for question–answer pairs about it and for rewrites of it. Each prompt receives only the title and the passage, and the evaluation questions never enter a prompt. Each backbone generates its own material with one sample per prompt and document at temperature 1.0 and top-p 0.95. Qwen2.5-7B is prompted by raw continuation and Llama-3.1-8B-Instruct through its chat template with an assistant prefill. The implications are split on the separator `---` and on newlines, and the document is appended as the last passage sequence. The pairs are parsed into questions and answers from the generated list, and the rewrites are parsed from the generated numbered list, keeping items longer than 20 characters. On Qwen2.5-7B, a document yields about eight passage sequences and 6.7 pairs on average, and the reference answer of an evaluation question appears verbatim in the passage of every document, in the implications of 52.6%, in the pairs of 73.4% and in the rewrites of 67.9%. On Llama-3.1-8B-Instruct, all three kinds were parsed for every document.

Implications

```
Let's read the following passage and produce a list of implications
derived directly or indirectly from the content.
Passage:
{title}
{context}
Implications:
```

Question–answer pairs

```
Let's read the following passage and rewrite it in a question-answer
format.
Passage:
{title}
```

```
{context}
Question 1:
```

Rewrites

```
Let's read the following passage and rewrite it in a few different
ways, each one separated by a newline.
Passage:
{title}
{context}
Rewritten passages:
```

C.3 Training, evaluation and learning rate

Training and evaluation. Every adapter is a LoRA adapter of rank 32 with $\alpha = 64$ and no dropout, trained with AdamW and a linear learning-rate schedule for ten epochs with a batch size of one sequence, so that per-sequence objectives take one update per sequence and pooled objectives one update per epoch. Answers are decoded greedily with at most 64 new tokens and graded by GPT-4.1 with the grading prompt of Zweiger et al. (2025). The unadapted model is evaluated once per run under the same prompt, and its accuracy varies by at most 1.3 points across runs on SQuAD (31.0–32.3% on Qwen2.5-7B and 39.1–40.0% on Llama-3.1-8B-Instruct).

Learning rate. All objectives are trained at 10^{-3}. SYN collapses at this rate with LoRA applied to all linear projections and in the multi-passage setting, where it takes $10\,|\mathcal{R}_{\mathrm{p}}|$ rather than ten updates, and is therefore trained at 10^{-4} in these settings. Table 8 reports every objective that was run at both rates. The pooled objectives prefer 10^{-3}, and MIX3 falls from 73.7% to 49.5% at 10^{-4}, so the exception favors SYN rather than the objectives compared with it. Under the query and value projections, SYN prefers 10^{-3} on Qwen2.5-7B and the two rates differ by about one point on Llama-3.1-8B-Instruct, so no exception is needed there.

Table 8: Closed-book accuracy (%) on SQuAD at the two learning rates, for every objective and setting run at both. The rate used in the main text is in bold.

Objective	Setting	10^{-3}	10^{-4}
SYN	Qwen2.5-7B, q/v	**53.8**	37.8
SYN	Llama-3.1-8B-Instruct, q/v	**57.8**	56.7
SYN	Qwen2.5-7B, all-lin.	33.1	**55.2**
SYN	Llama-3.1-8B-Instruct, all-lin.	26.1	**73.8**
SYN	Qwen2.5-7B, multi-passage	0.1	**36.5**
SYN	Llama-3.1-8B-Instruct, multi-passage	0.0	**43.7**
$\mathrm{SYN}_{\mathrm{pool}}$	Qwen2.5-7B, all-lin.	**44.9**	41.1
MIX	Qwen2.5-7B, all-lin.	**64.8**	46.1
MIX3	Qwen2.5-7B, all-lin.	**73.7**	49.5

C.4 Coverage of the experiments

All fourteen objectives are evaluated on SQuAD with LoRA applied to the query and value projections on both backbones, and on WikiDYK under the placements of Table 2. With LoRA applied to all linear projections on SQuAD, SYN, $\mathrm{SYN}_{\mathrm{pool}}$, MIX, MIX3, PQA, $\mathrm{PQA}_{\mathrm{ans}}$ and $\mathrm{PQA}_{\mathrm{pf}}$ are evaluated on Qwen2.5-7B, and SYN, $\mathrm{SYN}_{\mathrm{pool}}$, MIX and MIX3 on Llama-3.1-8B-Instruct. The multi-passage setting covers SYN, $\mathrm{SYN}_{\mathrm{pool}}$, MIX, KA, MIX3, $\mathrm{MIX3}_{\mathrm{unif}}$ and RQA on both backbones, and the token-matched controls are run on Qwen2.5-7B only. On SQuAD with Qwen2.5-7B under the query and value projections, SYN, MIX and MIX3 were also trained with a second random seed, which changes their accuracy by $+0.42 \pm 1.73$, $+0.18 \pm 1.14$ and $+0.28 \pm 1.11$ points (53.8% to 54.2%, 51.5% to 51.6% and 54.2% to 54.5%), and under the second seed MIX3 exceeds MIX by 2.87 ± 1.52 points.

C.5 MULTI-PASSAGE SETTING

In the multi-passage setting, the material of all 200 SQuAD passages is combined and a single adapter, with LoRA applied to the query and value projections, is trained on it for ten epochs and evaluated on all 974 questions. The pooled objectives split the combined material into 200 minibatches per epoch, each holding the kinds of material in the same proportion as the whole, so that they take the 2,000 updates they would take over 200 separate passages. SYN takes one update per sequence, that is 16,010 updates on Qwen2.5-7B and 18,760 on Llama-3.1-8B-Instruct, and diverges at 10^{-3} on both backbones, with accuracy falling to 0.1% and 0.0%. At 10^{-4} it reaches 36.5% and 43.7% (see Table 5). Two further configurations were run. The multi-passage configuration of Zweiger et al. (2025), with three epochs, a batch size of four, two steps of gradient accumulation, unsplit sequences and the same rank-32 adapter, changes accuracy from 31.3% to 34.0% on Qwen2.5-7B and from 39.3% to 38.6% on Llama-3.1-8B-Instruct. With LoRA applied to all linear projections on Qwen2.5-7B at 10^{-3}, every objective diverges over 2,000 updates, whereas MIX3 and MIX reach 44.8% and 42.3% when the combined material forms a single minibatch and ten updates are taken.

D ADDITIONAL RESULTS

D.1 DOSE-RESPONSE CURVES

These curves were run before the protocol of §5.1 was fixed, on Qwen2.5-7B with LoRA applied to the query and value projections, with the implications split on the separator only and 100 passages per point. Each point is therefore comparable to the others in this appendix but not directly to Table 2. The three interpolating objectives are defined in Appendix B.2. Table 9 reports the gain over the unadapted model. Removing passage tokens lowers the gain by 5.6 to 7.0 points per step, a span of 18.5 points or about nine standard errors, and at $\rho = 0.75$ the adapted model falls below the unadapted one. At $\rho = 1$ no token is supervised and the adapter is not updated. Removing question tokens or mixing soft and hard targets changes the gain by no more than about two standard errors across the whole range. The granularity into which the passage material is split and the number of rewrites in KA likewise change accuracy by less than two standard errors (not shown).

Table 9: Gain in closed-book accuracy (points) over the unadapted model on SQuAD with Qwen2.5-7B, 100 passages per point, ± 1 standard error. The first two rows remove a fraction ρ of the passage tokens (from SYN) or of the question tokens (from pairs with supervised questions) from the loss mask, and the third mixes a fraction λ of soft targets into QAFT. The $\rho = 1$ point of the second row and the $\lambda = 0$ point of the third are the same run.

ρ or λ	0	0.25	0.5	0.75	1
Passage tokens removed	14.9 ± 2.3	9.0 ± 2.5	3.4 ± 1.9	-3.7 ± 2.5	no supervision
Question tokens removed	12.9 ± 2.7	13.4 ± 2.6	16.1 ± 2.7	10.4 ± 2.7	11.6 ± 2.9
Soft targets mixed in	11.6 ± 2.9	12.1 ± 2.5	12.4 ± 2.8	11.7 ± 2.5	10.8 ± 2.6

D.2 TOKEN-MATCHED AND COMPUTE-MATCHED CONTROLS

The token counts of Table 4 are the number of tokens processed per document and epoch under the tokenizer of Qwen2.5-7B, averaged over the 200 documents. The token-matched version of MIX3 keeps every passage sequence and subsamples the pairs and rewrites of each document, retaining 35% of the pairs and 39% of the rewrites on average, so that the pairs and rewrites together contain about as many tokens as the pairs of MIX. The evaluation questions and the number of updates are unchanged. The five-epoch run of MIX3 uses the full material and takes five updates, so that its tokens times updates approximately matches that of the token-matched version over ten. Relative to MIX3, the token-matched version loses 5.20 ± 1.77 points under the query and value projections and 15.49 ± 1.96 under all linear projections, and the five-epoch run loses 10.9 and 5.2 points. Relative to MIX, the five-epoch run is -8.17 ± 1.59 points under the query and value projections and $+3.68 \pm 1.85$ under all linear projections. On Llama-3.1-8B-Instruct under the query and value

projections, the five-epoch run of MIX3 reaches 66.4% against 68.5% for MIX, and the token-matched version was not run.

D.3 Overlap between generated and evaluation questions

Because the pairs are generated from the passage, some generated questions resemble evaluation questions. We measure the word-set Jaccard similarity between each evaluation question and every generated question of its document. On Qwen2.5-7B, 0.6% of the 974 evaluation questions have an identical generated question and 4.2% (41 questions) have one with a similarity of at least 0.6, with a median maximum similarity of 0.24. On Llama-3.1-8B-Instruct, 6.0% (57 questions) have a near-duplicate. Table 10 reports question-level accuracy on the questions with and without a near-duplicate. Every objective, including SYN, is more accurate on the questions with a near-duplicate, which suggests that these questions ask for the most salient facts of their passages. Excluding them changes the difference between MIX3 and SYN by at most 0.3 points in every setting, and the gain of MIX3 over MIX is not concentrated on the questions with a near-duplicate.

Table 10: Question-level closed-book accuracy (%) on SQuAD, on the evaluation questions with and without a self-generated near-duplicate (word-set Jaccard similarity of at least 0.6). SYN under all linear projections uses a learning rate of 10^{-4}.

	Qwen2.5-7B, q/v		Qwen2.5-7B, all-lin.		Llama-3.1-8B-Instruct, q/v	
Objective	With	Without	With	Without	With	Without
SYN	63.4	53.3	56.1	55.0	63.2	57.2
MIX	68.3	50.4	80.5	63.6	75.4	67.8
MIX3	70.7	53.7	82.9	73.2	80.7	72.0
MIX3−SYN	+7.3	+0.4	+26.8	+18.2	+17.5	+14.8
MIX3−MIX	+2.4	+3.3	+2.4	+9.6	+5.3	+4.2

D.4 Sensitivity of SYN to the training protocol

Because SYN takes one update per sequence, its number of updates and the length of its sequences change with the protocol, whereas the pooled objectives take ten updates whatever the protocol. Table 11 compares SYN and MIX3 under three changes on SQuAD with LoRA applied to the query and value projections. Splitting the implications on newlines as well as on the separator shortens the sequences and raises the number of updates from about 20 to about 70 per document. On Qwen2.5-7B, the additional updates raise the accuracy of SYN by 4.01 ± 1.84 points and the shorter sequences by a further 3.37 ± 2.14. On Llama-3.1-8B-Instruct, the same two changes lower it by 3.91 ± 1.81 and 7.87 ± 2.21 points, and the adapted model repeats the passage instead of answering, with a median answer length of 49 words. Including the padded positions in the loss mask, as the released trainer does, lowers SYN by 11.65 ± 2.33 points on Qwen2.5-7B and leaves it unchanged on Llama-3.1-8B-Instruct (Appendix B.2). MIX3 changes by less than one point under the split on both backbones. The main text uses the split protocol for every objective.

Table 11: Paired differences in closed-book accuracy (points, ± 1 standard error) on SQuAD with LoRA applied to the query and value projections, for one change of the training protocol at a time. Updates are per document, and * marks a difference exceeding two standard errors.

Objective	Change	Qwen2.5-7B	Llama-3.1-8B-Instruct
SYN	padded positions removed from the loss mask	$+11.65 \pm 2.33^*$	$+0.70 \pm 2.14$
SYN	updates raised from 20 to 70, sequences unsplit	$+4.01 \pm 1.84^*$	$-3.91 \pm 1.81^*$
SYN	sequences split on newlines, 70 updates	$+3.37 \pm 2.14$	$-7.87 \pm 2.21^*$
MIX3	sequences split on newlines, ten updates	-0.20 ± 1.48	$+0.49 \pm 1.60$