ALIGNEDIT: ALIGNMENT-PRESERVING KNOWLEDGE EDITING FOR POST-TRAINED LANGUAGE MODELS

Anonymous authors
Paper under double-blind review

ABSTRACT

Knowledge editing updates specific knowledge while preserving unrelated behavior. This is challenging for post-trained language models, where edits must work with model-specific **chat templates** and **open-ended generation** without disrupting acquired capabilities. Yet most methods and evaluations target language models used directly after pretraining and assess fixed target continuations. We introduce an **evaluation protocol** combining model-specific input formatting with response-level correctness. It shows that representative methods struggle to balance Reliability, Generality, and Locality during sequential editing, even with chat-formatted editing inputs. We then propose **ALIGNEDIT**, which independently constructs knowledge-specific teachers from the original model through localized fine-tuning and distills them into a shared student using dense supervision along student-generated trajectories. We evaluate ALIGNEDIT on Qwen3-4B and SmolLM3-3B using CounterFact and ZsRE, together with five general-capability benchmarks. Across both editing benchmarks, ALIGNEDIT achieves a better balance among Reliability, Generality, and Locality than the evaluated baselines, with limited degradation in average general-capability performance. These results underscore the importance of accounting for post-training behavior in practical knowledge editing.

1 INTRODUCTION

Large language models (LLMs) encode factual knowledge and relationships in their parameters and use this knowledge to answer questions and perform downstream tasks (Sun et al., 2024; Li et al., 2024b). However, their knowledge may be incorrect, outdated, biased, or inconsistent with user needs. Updating an entire model through retraining or broad fine-tuning is costly and may unintentionally alter unrelated behavior. Knowledge editing instead aims to modify specific knowledge while preserving the model's remaining capabilities (Zhang et al., 2024b).

Recent knowledge editing methods have achieved impressive performance (Wang et al., 2024f), but most were developed and evaluated on language models used directly after pretraining, before instruction tuning or alignment-oriented post-training. Even studies of instruction-tuned models often retain conventional editing objectives and evaluation protocols, with limited consideration of two defining characteristics of post-trained models: **model-specific chat templates** and **open-ended response generation**. Consequently, it remains unclear whether editing success measured against fixed target continuations translates into correct generated responses when models are prompted through chat templates. This distinction is especially important in practical deployments, where targeted updates must be incorporated without disrupting instruction-following behavior or other capabilities acquired during post-training.

To address this gap, we first establish an **evaluation protocol** tailored to post-trained models and use it to systematically assess existing knowledge editing methods. The protocol applies model-specific chat templates and assesses the correctness of generated responses. Our evaluation reveals that these methods struggle to balance edit success and preservation of unrelated knowledge under sequential editing, even when their editing objectives use chat-formatted inputs. We then propose **ALIGNEDIT**, a **post-training-aware knowledge editing method** designed to update target knowledge while preserving the model's existing capabilities. ALIGNEDIT formulates knowledge editing as generation-level reward maximization. Building on this formulation, it trains a shared student

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.

using knowledge-specific teachers constructed through localized fine-tuning. Each teacher is independently adapted from the original model to the target knowledge and provides dense token-level supervision on student-generated responses (Shenfeld et al., 2026). This design avoids accumulating updates in the teachers, allowing the student to acquire the target knowledge while limiting degradation of its existing capabilities.

We conduct our main evaluation of ALIGNEDIT on two post-trained language models, Qwen3-4B (Yang et al., 2025a) and SmolLM3-3B (Bakouch et al., 2025), using the knowledge editing benchmarks CounterFact (Meng et al., 2022) and ZsRE (Levy et al., 2017). We also assess capability retention using five additional benchmarks covering instruction following, mathematical and general reasoning, code generation, and tool calling. Across both knowledge editing benchmarks, ALIGNEDIT achieves higher overall editing scores than representative baselines, with limited degradation in average general-capability performance.

Our contributions are as follows:

- We introduce an evaluation protocol for knowledge editing on post-trained models that combines model-specific input formatting with response-level assessment, and show that matching input formatting alone does not resolve the limitations of existing methods under sequential editing.
- We propose ALIGNEDIT, a post-training-aware knowledge editing method in which independently constructed knowledge-specific teachers supervise student-generated responses, enabling target knowledge to be edited while preserving the model's existing capabilities.
- We demonstrate that ALIGNEDIT achieves a better balance among Reliability, Generality, and Locality than representative baselines across knowledge editing datasets, with limited degradation in average performance across five general-capability benchmarks.

2 PRELIMINARIES AND RELATED WORK

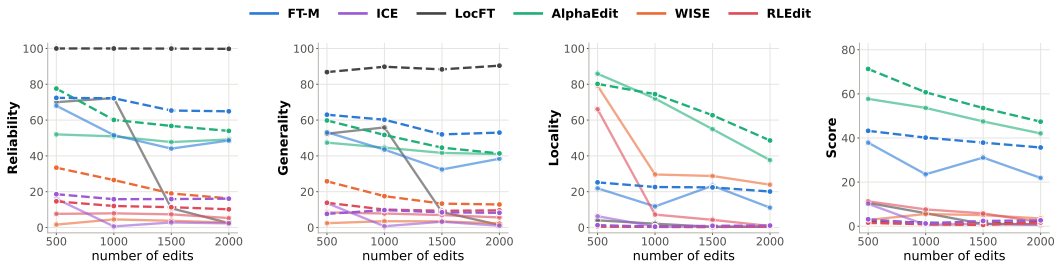
Knowledge Editing. Knowledge editing aims to update knowledge encoded in LLMs while preserving their behavior on unrelated inputs (Wang et al., 2024f). Formally, let $k = (x, y^\star) \in \mathcal{K}$ denote a knowledge edit, where x is an input prompt (e.g., "What is the capital of France?") and $y^\star$ is the desired answer (e.g., "London"). The pair $(x, y^\star)$ specifies the target association to be introduced. Let θ_0 and θ denote the parameters of the original and edited models, respectively. A broad class of knowledge-editing objectives can be expressed in the following unified form:

$$\max_{\theta} \ \mathbb{E}_{(x,y^\star)\sim\mathcal{K}}[\log p_\theta(y^\star \mid x)] + \mathrm{Reg}(\theta;\theta_0), \tag{1}$$

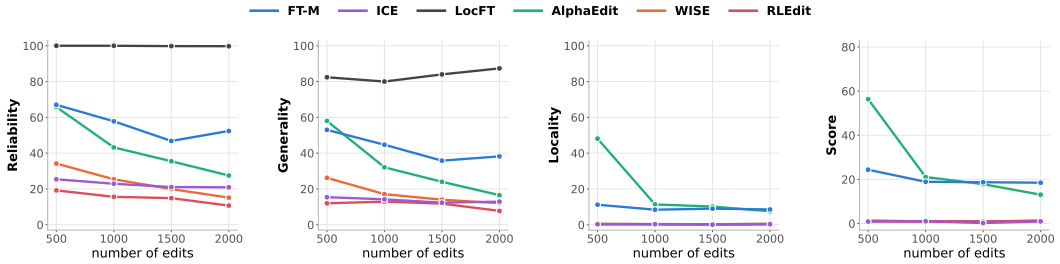
where $\mathrm{Reg}(\theta;\theta_0)$ denotes a method-specific regularization term that discourages unintended deviations from the original model's parameters or behavior. Existing methods realize the edited behavior through **direct gradient-based updates** (Gangadhar & Stratos, 2024; Qi et al., 2025; Yang et al., 2025c), **external memory or adapters** (Mitchell et al., 2022b; Hartvigsen et al., 2023; Yu et al., 2024; Wang et al., 2024a;d), **hypernetworks** that predict model updates (Cao et al., 2021; Mitchell et al., 2022a; Zhang et al., 2024a; Li et al., 2025), or **locate-and-edit** procedures that identify model components associated with the target knowledge and directly modify their parameters to encode the desired association (Meng et al., 2022; 2023; Li et al., 2024a; Fang et al., 2025).

In practice, knowledge edits often arrive incrementally rather than all at once. To incorporate them, an editor updates the model from θ_{i-1} to θ_i using an individual edit k_i or a batch of edits $\mathcal{B}_i \subseteq \mathcal{K}$, and each updated model serves as the starting point for the next update. This process is commonly referred to as **sequential** or **lifelong** editing (Hartvigsen et al., 2023; Hu et al., 2024; Ma et al., 2025; Hu et al., 2025). Performance is typically evaluated after the full edit sequence, reflecting deployment settings in which models must retain prior edits and preserve unrelated knowledge as new updates accumulate.

Evaluation of Editing Methods. Editing methods are typically evaluated using four primary criteria: (1) **Reliability**, whether the target knowledge is successfully edited, (2) **Generality**, whether the edit transfers to semantically equivalent prompts, (3) **Locality**, whether the model's behavior on unrelated inputs is preserved, and (4) **Score**, the harmonic mean of the three preceding metrics.



(a) Original editing objective.



(b) chat-adapted editing objective.

Figure 1: Sequential editing performance under the original and chat-adapted editing objectives. Dotted and solid curves show results under BASEEVAL and CHATEVAL, respectively.

The precise protocols used to compute these metrics vary across studies and datasets (Pohl et al., 2025). Early work measured the proportion of edits for which the desired answer was assigned a higher probability than the model's pre-edit answer. Another widely used protocol measures exact-match accuracy between the expected answer and the model's prediction. However, recent work argues that these protocols do not reliably indicate whether generated responses correctly express the target knowledge. These studies instead advocate generating open-ended responses and evaluating their correctness using an LLM judge for Reliability and Generality (Pohl et al., 2025; Yang et al., 2025b). Building on these insights, we define our evaluation protocol as follows. For each target edit $k = (x, y^\star) \in \mathcal{K}$, let $\mathcal{K}'$ contain semantically equivalent prompts paired with the same target, $k' = (x', y^\star)$, and let $\mathcal{K}^\neg$ contain unrelated prompts $x^\neg$. We define BASEEVAL as an evaluation protocol that generates responses without applying model-specific chat templates. Reliability and Generality are evaluated with an LLM judge, whereas Locality measures exact matches between the original and edited models' top-1 predictions on unrelated prompts.

$$\textsc{BaseEval}(\mathcal{D}) := \begin{cases} \mathbb{E}_{(u,v)\sim\mathcal{D}}\mathbb{E}_{\mathbf{a}\sim p_\theta(\cdot|u)}\left[\mathbf{1}\{\mathrm{LLM}(u,\mathbf{a},v) = \text{correct}\}\right], & \mathcal{D} = \mathcal{K} \text{ or } \mathcal{K}', \\ \mathbb{E}_{u\sim\mathcal{D}}\left[\mathbf{1}\left\{\arg\max_v p_{\theta_0}(v \mid u) = \arg\max_v p_\theta(v \mid u)\right\}\right], & \mathcal{D} = \mathcal{K}^\neg, \end{cases} \tag{2}$$

where $\mathbf{a} = (a_1, \ldots, a_T)$ denotes a response generated autoregressively by the edited model. The three editing metrics are then

$$\text{Reliability} = \textsc{BaseEval}(\mathcal{K}), \quad \text{Generality} = \textsc{BaseEval}(\mathcal{K}'), \quad \text{Locality} = \textsc{BaseEval}(\mathcal{K}^\neg). \tag{3}$$

3 KNOWLEDGE EDITING ON POST-TRAINED MODELS

In this section, we define an evaluation protocol for knowledge editing on post-trained models and investigate two questions: (1) *How well do existing knowledge-editing methods perform on post-trained models?* (2) *Can their effectiveness be improved by adapting the editing objective to the post-trained setting?*

Evaluation Protocol for Post-Trained Models. Post-trained models use model-specific input templates, commonly referred to as chat templates, during training and deployment. We therefore

apply the corresponding formatting function when generating responses and define the evaluation protocol for post-trained models as follows:

$$\textsc{ChatEval}(\mathcal{D}) := \begin{cases} \mathbb{E}_{(u,v)\sim\mathcal{D}}\mathbb{E}_{\mathbf{a}\sim p_\theta(\cdot\mid f^c(u))}\left[\mathbf{1}\{\mathrm{LLM}(u,\mathbf{a},v)=\text{correct}\}\right], & \mathcal{D}=\mathcal{K} \text{ or } \mathcal{K}', \\ \mathbb{E}_{u\sim\mathcal{D}}\Big[\mathbf{1}\Big\{\arg\max_v p_{\theta_0}(v\mid f^c(u)) = \arg\max_v p_\theta(v\mid f^c(u))\Big\}\Big], & \mathcal{D}=\mathcal{K}^\neg, \end{cases} \tag{4}$$

where $f^c(\cdot)$ denotes the model-specific chat-template formatting function. As in Equation 3, Reliability, Generality, and Locality are obtained by evaluating CHATEVAL on $\mathcal{K}$, $\mathcal{K}'$, and $\mathcal{K}^\neg$, respectively.

Evaluating Editing Methods on Post-Trained Models. Using CHATEVAL, we ask: *How well do existing knowledge-editing methods perform on post-trained models?*

We consider six representative methods spanning four categories. These comprise the gradient-based methods **FT-M** (Zhang et al., 2024b), **ICE** (Qi et al., 2025), and **LocFT** (Yang et al., 2025c), the external-memory-based method **WISE** (Wang et al., 2024d), the hypernetwork-based method **RLEdit** (Li et al., 2025), and the locate-and-edit method **AlphaEdit** (Fang et al., 2025). We evaluate these methods under sequential editing on the post-trained Qwen3-4B model (Yang et al., 2025a), using an edit batch size of 10 and a question-answering version of CounterFact (Meng et al., 2022).[1] Although CHATEVAL serves as our primary evaluation protocol, we additionally report BASEEVAL to quantify the effect of model-specific input formatting on measured editing performance.

Figure 1a reports performance as the number of sequential edits increases. BASEEVAL generally yields higher Reliability and Generality than CHATEVAL, whereas the Locality comparison is method-dependent: BASEEVAL performs better for FT-M and AlphaEdit, while CHATEVAL performs better for WISE and RLEdit, with smaller differences for ICE and LocFT. This discrepancy also affects the aggregate Score, with BASEEVAL most notably overestimating the performance of FT-M and AlphaEdit after 2,000 edits relative to CHATEVAL. These results show that strong performance under BASEEVAL does not necessarily transfer to the chat-formatted setting, highlighting the **limitations of existing editing methods on post-trained models.**

Adapting Editing Methods with Post-Trained Models. The discrepancy between BASEEVAL and CHATEVAL motivates a follow-up question: *Can existing methods be improved by adapting their editing objectives with* CHATEVAL*?* To test this possibility, we replace each prompt x in Equation 1 with its chat-formatted counterpart $f^c(x)$, yielding the following **chat-adapted objective**:

$$\max_\theta \ \mathbb{E}_{(x,y^\star)\sim\mathcal{K}}[\log p_\theta(y^\star \mid f^c(x))] + \mathrm{Reg}(\theta;\theta_0,f^c). \tag{5}$$

Figure 1b reports the results. Although the chat-adapted objective improves Reliability and Generality under CHATEVAL, Locality deteriorates sharply, causing the final Score to fall below 20 for every evaluated method. Thus, aligning the editing and evaluation formats merely shifts the trade-off between edit success and preservation rather than overcoming the limitations of existing methods for reliable sequential editing of post-trained models. This motivates an **editing method explicitly tailored to the post-trained setting**.

4 KNOWLEDGE EDITING AS REWARD MAXIMIZATION

We hypothesize that existing knowledge-editing methods perform poorly on post-trained models because the token-level likelihood objectives in Equations 1 and 5 impose an overly rigid constraint: they force the model to assign high probability to a specific target continuation. During sequential editing, these constraints accumulate, progressively shifting the model away from its post-training distribution and degrading its general capabilities (Luo et al., 2025). This shift may be particularly harmful to post-trained models, whose instruction-following behavior is shaped by carefully curated post-training data mixtures and model-specific formatting conventions (Xu et al., 2026; Kimi Team et al., 2026).

[1] We adapt CounterFact to a question-answering format suitable for post-trained models. Appendix A describes this conversion process.

Motivated by this hypothesis, we formulate editing success through a generation-level objective that rewards any response correctly expressing the target knowledge. Rather than relying solely on the likelihood objectives in Equations 1 and 5, we therefore seek to directly maximize the Score under CHATEVAL:

$$\max_{\theta} \ \mathrm{HM}(\textsc{ChatEval}(\mathcal{K}), \textsc{ChatEval}(\mathcal{K}'), \textsc{ChatEval}(\mathcal{K}^{\neg})) \tag{6}$$

Objective 6 can be viewed as a reward-maximization problem and, in principle, optimized using policy-gradient methods (Williams, 1992; Schulman et al., 2015; 2017; Ouyang et al., 2022; Shao et al., 2024). Direct optimization, however, faces two practical challenges. First, $\mathcal{K}'$ and $\mathcal{K}^{\neg}$ are held-out evaluation sets that are generally unavailable during editing. Only the target-edit set $\mathcal{K}$ is accessible. Second, the LLM-judge signal used for Reliability and Generality is a sparse binary reward: any response judged incorrect receives zero reward, providing no positive learning signal precisely when the model has not yet acquired the edited association $(x, y^\star)$. To address these challenges, we restrict optimization to $k \sim \mathcal{K}$ and replace the sparse judge reward with a dense surrogate reward $r(k, \mathbf{a})$. We then adopt the following trust-region policy-optimization objective as our starting point (Schulman et al., 2015; Jaques et al., 2017):

$$\max_{\pi_\theta} \mathbb{E}_{k\sim\mathcal{K}}\Big[\mathbb{E}_{\mathbf{a}\sim\pi_\theta(\cdot|f^c(x))}[r(k,\mathbf{a})] - \beta D_{\mathrm{KL}}(\pi_\theta(\cdot \mid f^c(x)) \,\|\, \pi_{\mathrm{ref}}(\cdot \mid f^c(x)))\Big], \tag{7}$$

where $\pi_{\mathrm{ref}} := \pi_{\theta_0}$ is the fixed policy induced by the original model, and $\beta > 0$ controls the strength of the KL constraint. The optimal policy that maximizes this objective admits the following closed-form solution (Rafailov et al., 2023; Shenfeld et al., 2026):

$$\begin{aligned} \pi^\star(\mathbf{a} \mid f^c(x)) &= \frac{1}{Z(k)} \pi_{\mathrm{ref}}(\mathbf{a} \mid f^c(x)) \exp\left(\frac{r(k,\mathbf{a})}{\beta}\right), \\ Z(k) &= \sum_{\mathbf{a}'} \pi_{\mathrm{ref}}(\mathbf{a}' \mid f^c(x)) \exp\left(\frac{r(k,\mathbf{a}')}{\beta}\right). \end{aligned} \tag{8}$$

Rearranging Equation 8 gives the equivalent reward parameterization

$$r(k,\mathbf{a}) = \beta\left[\log \pi^\star(\mathbf{a} \mid f^c(x)) - \log \pi_{\mathrm{ref}}(\mathbf{a} \mid f^c(x))\right] + C(k), \tag{9}$$

where $C(k) = \beta \log Z(k)$ is independent of $\mathbf{a}$ and therefore does not affect the optimal policy.

In practice, neither the dense reward r nor its corresponding optimal policy $\pi^\star$ is available. We therefore introduce a fixed, **knowledge-specific teacher policy** π_k^T that approximates $\pi^\star$ for each knowledge k. Replacing $\pi^\star$ with π_k^T in the reward parameterization of Equation 9 and substituting the resulting teacher-induced reward into Equation 7 yields

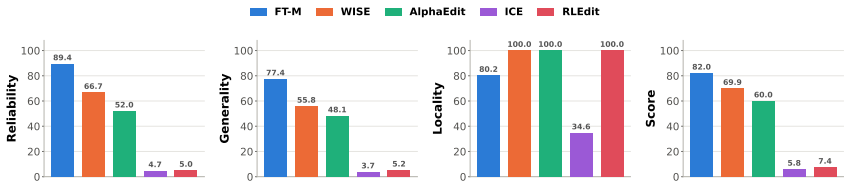
$$\begin{aligned} &\max_{\pi_\theta} \beta\, \mathbb{E}_{k\sim\mathcal{K}}\Big[\mathbb{E}_{\mathbf{a}\sim\pi_\theta(\cdot|f^c(x))}\big[\log \pi_k^T(\mathbf{a} \mid f^c(x)) - \log \pi_\theta(\mathbf{a} \mid f^c(x))\big]\Big] + \mathbb{E}_{k\sim\mathcal{K}}[C(k)] \\ &\iff \min_{\pi_\theta} \beta\, \mathbb{E}_{k\sim\mathcal{K}}\big[D_{\mathrm{KL}}\big(\pi_\theta(\cdot \mid f^c(x)) \,\|\, \pi_k^T(\cdot \mid f^c(x))\big)\big] + \mathrm{const}. \end{aligned} \tag{10}$$

Thus, under the teacher-induced reward, the objective in Equation 7 is equivalent to minimizing the reverse KL divergence from the student policy to the teacher policy. For autoregressive language models, this sequence-level divergence decomposes into token-level reverse-KL terms evaluated on prefixes generated by the student, yielding the on-policy distillation objective (Lu & Thinking Machines Lab, 2025; Li et al., 2026b).

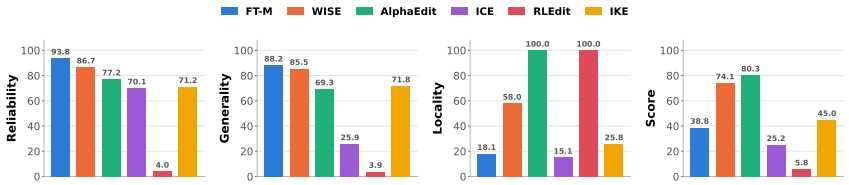
5 CONSTRUCTING ALIGNEDIT

We now describe how ALIGNEDIT implements the reward-maximization framework introduced in Section 4. For each edit k_i, we update the shared student π_θ using the corresponding knowledge-specific teacher $\pi_{k_i}^T$. At each student update, we sample a trajectory $\mathbf{a} \sim \pi_{\mathrm{sg}(\theta)}(\cdot \mid f^c(x_i))$ from the current student. Holding the sampled trajectory fixed, we define the following token-level reward-maximization loss:

$$\mathcal{L}_{\mathrm{reward}}(\theta) = \frac{1}{|\mathbf{a}|} \sum_{t=1}^{|\mathbf{a}|} D_{\mathrm{KL}}\big(\pi_\theta(\cdot \mid f^c(x_i), \mathbf{a}_{<t}) \,\|\, \pi_{k_i}^T(\cdot \mid f^c(x_i), \mathbf{a}_{<t})\big). \tag{11}$$



(a) Original editing objective.



(b) chat-adapted editing objective.

Figure 2: Non-sequential editing performance under original and chat-adapted editing objectives, evaluated with CHATEVAL.

Because the student learns by matching the teacher, teacher quality is critical. A suitable teacher should provide supervision aligned with the desired behavior represented by the optimal policy $\pi^\star$, namely, a favorable balance among Reliability, Generality, and Locality. We therefore evaluate candidate teachers on all three metrics and their resulting CHATEVAL Score.

5.1 SELECTING A KNOWLEDGE-SPECIFIC TEACHER

A key characteristic of our teacher construction is that teachers operate **non-sequentially**. For each edit k_i, we train a teacher **independently** by initializing it from the original model, adapting it only to k_i, transferring its behavior to the shared student, and discarding it before the next edit. Because updates do not accumulate across teachers, this procedure prevents interference from carrying over between teacher constructions and may mitigate the degradation observed under sequential editing in Section 3.

Evaluating Editing Methods in a Non-Sequential Setting. To identify an editing method suitable for constructing knowledge-specific teachers, we evaluate editing methods in a non-sequential setting. As in Section 3, we apply each method using both the original objective in Equation 1 and the chat-adapted objective in Equation 5, and evaluate the resulting models with CHATEVAL. We omit LocFT as its distinction from FT-M arises from distributing gradient updates across sequential edits. When each edit is applied independently, LocFT reduces to FT-M. In addition to the parameter-editing methods considered above, we include IKE, a non-parametric editing method based on in-context learning (Zheng et al., 2023). IKE is suitable for teacher construction because the resulting teacher provides only supervision and need not retain persistent parameter updates, whereas the student is updated during distillation.

Figure 2 presents the results in the non-sequential setting. Whereas most methods achieve a Score below 20 under sequential editing, most exceed 20 when each edit is performed independently, with at least one method achieving a Score above 80. Several methods also attain Locality close to 100. These results show that existing editing methods can construct strong models specialized to individual edits and can therefore provide effective supervision as knowledge-specific teachers.

Selecting FT-M for Teacher Construction. The strong non-sequential editing performance of FT-M makes it a promising candidate for constructing knowledge-specific teachers. Its suitability is further supported by four practical advantages. First, unlike methods that maintain auxiliary modules or external memory, such as WISE and GRACE, FT-M requires no additional component and is readily compatible with model-serving backends such as `vLLM` (Kwon et al., 2023) and `SGLang` (Zheng et al., 2024a). Second, unlike locate-and-edit methods such as AlphaEdit and MEMIT, FT-M does not require memory-intensive operations such as matrix inversion, making it easier to scale to large

numbers of edits. Third, since each teacher is constructed independently, its updates do not accumulate across edits, reducing the need for conservative update mechanisms or separately trained hypernetworks such as those used by RLEdit and MEND. Fourth, unlike IKE, an FT-M teacher does not receive an answer-bearing demonstration during generation, thereby avoiding potential side effects of answer-conditioned generation (Lee et al., 2026). More broadly, FT-M is simple, achieves strong editing performance (Zhang et al., 2024b; Gangadhar & Stratos, 2024; Yang et al., 2025c), and is supported by widely used fine-tuning libraries such as `LLaMA-Factory` (Zheng et al., 2024b), making it easy to scale across models. Because FT-M trained with the original editing objective achieves a better balance between incorporating target knowledge and preserving unrelated knowledge than its chat-adapted counterpart, we use this configuration to construct the teachers.

5.2 Final AlignEdit Loss

Although knowledge-specific teachers closely approximate the optimal policy, they may not provide ideal supervision for every example. We therefore add a target negative-log-likelihood term to obtain the final AlignEdit loss:

$$\begin{aligned} \mathcal{L}_{\text{target}}(\theta) &= -\log \pi_\theta(y_i^\star \mid x_i)\,, \\ \mathcal{L}_{\text{AlignEdit}}(\theta) &= \mathcal{L}_{\text{reward}}(\theta) + \lambda \mathcal{L}_{\text{target}}(\theta). \end{aligned} \tag{12}$$

where $\lambda \geq 0$ controls the strength of direct target supervision. Because the teacher provides token-level supervision over the entire generated response, including reasoning and instruction-following tokens beyond those expressing the target knowledge, we update all layers of the student model using $\mathcal{L}_{\text{AlignEdit}}$. Algorithm 1 summarizes AlignEdit for batch editing, and Table 10 reports the hyperparameter settings.

6 Experiments

In this section, we investigate two questions: (1) How does AlignEdit compare with existing knowledge-editing baselines in a sequential editing setting? (2) How well does AlignEdit preserve the general capabilities of post-trained models?

6.1 Experimental Settings

We compare AlignEdit with the editing methods examined in Section 3, reporting baselines trained with the original objective because the chat-adapted objective generally improves Reliability and Generality but substantially reduces Locality and overall Score. We use SmolLM3-3B and Qwen3-4B as backbone models and evaluate all methods on 2,000 edits sampled from each of CounterFact and ZsRE, using an edit batch size of 10.

To assess general-capability preservation, we evaluate the edited models across five dimensions central to post-training: (1) **mathematical reasoning** using AIME 2025 (Dekoninck et al., 2026), (2) **general reasoning** using GPQA-Diamond (Rein et al., 2023), (3) **instruction following** using IFEval (Zhou et al., 2023), (4) **code generation** using LiveCodeBench-v6 (Jain et al., 2025), and (5) **tool calling** using BFCL-v4 (Patil et al., 2025).

6.2 Sequential Knowledge Editing

Table 1 summarizes the sequential editing results. AlignEdit achieves the highest Reliability, Generality, and overall Score in every model–dataset setting. It also attains the highest Locality in three of the four settings and the second-highest Locality on CounterFact with SmolLM3-3B. Among the baselines, LocFT achieves relatively strong Reliability and Generality. For example, it reaches a Reliability of 66.70 on CounterFact with SmolLM3-3B, highlighting the effectiveness of its breadth-first update strategy. In contrast, most of the remaining baselines struggle either to incorporate the target knowledge or to preserve unrelated knowledge, with some leading to model collapse. Taken together, these findings show that accounting for the characteristics of post-trained models enables more robust sequential editing.

Table 1: Model-editing performance on ZsRE and CounterFact. The best result for each metric is **bold** and the second-best is underlined.

Dataset	Method	Qwen3-4B				SmolLM3-3B			
		Reliability	Generality	Locality	Score	Reliability	Generality	Locality	Score
CounterFact	Pre-edited	2.25	2.50	–	–	2.20	2.35	–	–
	FT-M	48.55	38.40	11.07	21.90	3.70	2.55	52.45	4.40
	ICE	2.30	0.95	0.27	0.60	3.10	2.65	34.75	4.12
	LocFT	2.45	1.40	0.45	0.90	66.70	55.40	36.15	49.42
	AlphaEdit	49.15	41.00	37.62	42.10	39.55	38.60	0.00	0.00
	WISE	15.15	12.05	0.38	1.08	18.30	12.95	0.38	1.09
	RLEdit	5.25	5.60	0.80	1.85	3.40	3.45	**97.45**	5.05
	AlignEdit	**77.80**	**59.65**	**54.50**	**62.54**	**73.10**	**65.80**	77.98	**71.94**
ZsRE	Pre-edited	4.45	3.65	–	–	4.95	3.90	–	–
	FT-M	5.50	5.20	32.87	7.42	10.30	7.55	80.91	12.40
	ICE	5.50	4.50	36.29	6.95	8.05	5.35	67.93	9.21
	LocFT	49.70	37.25	29.89	37.30	36.90	31.40	77.83	41.79
	AlphaEdit	1.60	1.70	23.98	2.39	10.65	10.75	0.54	1.47
	WISE	9.40	8.45	19.74	10.89	21.20	16.50	21.90	19.55
	RLEdit	7.05	5.30	59.66	8.64	11.95	11.05	31.12	14.54
	AlignEdit	**63.90**	**53.15**	**66.60**	**60.63**	**58.20**	**46.50**	**81.80**	**58.92**

Table 2: General-capability performance after editing on ZsRE and CounterFact. Avg. is the mean over five benchmarks. The best result for each benchmark is **bold** and the second-best is underlined.

Dataset	Method	Qwen3-4B						SmolLM3-3B					
		AIME25	GPQA	IFEval	LCB	BFCL	Avg.	AIME25	GPQA	IFEval	LCB	BFCL	Avg.
CounterFact	Pre-edited	43.33	52.02	81.15	30.18	82.81	57.90	31.11	39.90	65.80	14.98	54.55	41.27
	FT-M	3.33	7.58	25.14	2.42	52.79	18.25	**37.78**	36.87	54.47	14.32	54.11	39.59
	ICE	0.00	0.51	15.16	0.22	36.61	10.50	32.22	28.79	62.85	13.22	52.98	38.01
	LocFT	0.00	3.03	15.53	0.00	44.05	12.52	33.33	31.82	64.33	**14.76**	50.23	38.89
	AlphaEdit	4.44	17.17	39.93	8.37	62.68	26.52	0.00	0.00	14.97	0.00	30.87	9.17
	RLEdit	0.00	0.00	14.97	0.00	30.98	9.19	29.67	**40.93**	**65.25**	13.44	53.89	**40.64**
	AlignEdit	**42.22**	**48.99**	**75.97**	**32.82**	**80.44**	**56.09**	23.33	36.87	61.74	13.66	**64.24**	39.97
ZsRE	FT-M	1.11	26.77	34.75	5.07	59.05	25.35	27.78	**37.88**	**63.96**	13.22	52.98	39.16
	ICE	11.11	9.60	33.83	7.27	63.86	25.13	28.89	34.85	59.70	13.22	40.57	35.45
	LocFT	15.56	10.61	29.94	11.23	67.21	26.91	**31.11**	33.84	60.81	**15.42**	48.09	37.85
	AlphaEdit	0.00	4.55	13.86	0.00	30.71	9.82	0.00	0.00	14.97	0.00	30.87	9.17
	RLEdit	36.67	**51.52**	65.62	21.59	81.13	51.31	14.44	4.04	25.69	13.88	32.66	18.14
	AlignEdit	**40.00**	46.46	**76.89**	**32.16**	**81.27**	**55.36**	25.56	36.87	59.15	14.98	**59.35**	**39.18**

6.3 Impact on Post-Trained Model Capabilities

Table 2 reports general-capability retention after editing on ZsRE and CounterFact. We exclude WISE as it requires a separate adapter and inference procedure that are incompatible with accelerated LLM inference engines, making large-scale evaluation on these reasoning-intensive benchmarks computationally impractical.

Across the evaluated editing methods, AlignEdit preserves general capabilities most effectively, with the average performance decreasing by at most 2.6 percentage points. On SmolLM3-3B, several baselines retain relatively strong performance on AIME25, GPQA, and IFEval but exhibit larger degradation on BFCL. Since BFCL relies heavily on generating special tokens for tool calls, aggressive modification of localized layers may disrupt the model's ability to produce these tokens correctly. Overall, the consistently small degradation achieved by AlignEdit indicates that it best preserves the alignment acquired during post-training.

7 Ablations

In this section, we examine how λ and batch size affect editing performance. We also investigate whether external context describing the causal basis of the new knowledge improves AlignEdit by enriching the supervision provided by its teachers (Li et al., 2026a).

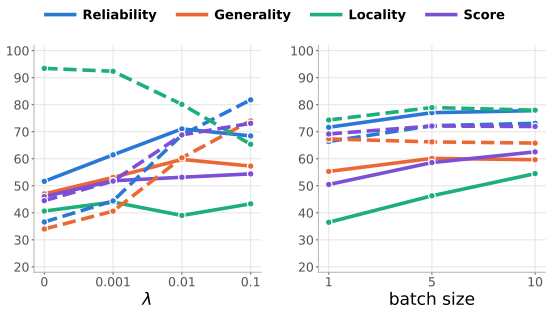


Figure 3: Effects of λ and batch size on editing performance. Solid and dotted curves denote Qwen3-4B and SmolLM3-3B, respectively.

Table 3: Two-hop knowledge-editing performance. Higher values are better for all metrics except SRR.

Model	Method	H-Acc↑	M-Acc↑	SRR↓
Qwen3-4B	Pre-edited	64.97	47.60	–
	SDFT	78.33	74.00	2.90
	CODE	80.80	72.60	3.70
	AlignEdit	82.90	75.40	18.00
	+context	**90.17**	**78.20**	**0.40**
SmolLM3-3B	Pre-edited	61.47	46.80	–
	SDFT	80.90	71.20	6.10
	CODE	87.50	**77.40**	1.88
	AlignEdit	79.87	74.20	20.00
	+context	**90.63**	75.20	**1.80**

7.1 Effects of Different Hyperparameters

λ ablation. The left panel of Figure 3 shows the effect of varying λ. As expected, increasing λ improves Reliability and Generality at the cost of reduced Locality, allowing λ to control the trade-off between edit efficacy and preservation.

Batch-size ablation. The right panel of Figure 3 shows the effect of varying the batch size. For Qwen3-4B, Locality improves as the batch size increases. One possible explanation is that larger batches provide more stable gradient estimates, allowing optimization to better balance edit efficacy with the preservation of unrelated knowledge. Moreover, the overall Score improves more from a batch size of 1 to 5 than from 5 to 10, suggesting that AlignEdit achieves most of its gains with moderate batch sizes and does not require large batches for strong performance.

7.2 Incorporating External Context

To incorporate context into AlignEdit, we combine the target-knowledge context c_i with the input x_i after constructing each FT-M teacher and apply the chat template, using $f^c(c_i, x_i)$ during distillation. We evaluate this variant on MQuAKE-T (Zhong et al., 2024) using the same contexts as CODE (Li et al., 2026a), and compare against CODE and SDFT (Shenfeld et al., 2026), context-based knowledge-editing and knowledge-injection methods, respectively. H-Acc measures intermediate-fact accuracy, M-Acc measures final-answer accuracy, and Self-Refutation Rate (SRR) measures rejection or contradiction of the supplied edit (Li et al., 2026a).

Table 3 presents the results. Across both models, incorporating context improves H-Acc and M-Acc while substantially reducing SRR, demonstrating the benefit of additional contextual information. Notably, even without context, AlignEdit outperforms CODE on both accuracy metrics for Qwen3-4B, showing that careful teacher construction can achieve strong performance without CODE's additional causal-bootstrapping procedure or preliminary SFT stage.

8 Conclusion

We revisited knowledge editing for post-trained language models, where edits must preserve unrelated knowledge and capabilities acquired during post-training. Our evaluation shows that existing methods struggle with sequential editing under chat-formatted inputs and open-ended generation. We introduced AlignEdit, which constructs knowledge-specific teachers through lightweight, localized fine-tuning and distills them into a shared student while limiting cross-edit interference. On Qwen3-4B and SmolLM3-3B, AlignEdit achieves a better balance among Reliability, Generality, and Locality than the evaluated baselines on CounterFact and ZsRE while better preserving general capabilities. These results position AlignEdit as a practical method for updating post-trained language models without substantially compromising their existing capabilities.

AI USE STATEMENT

In this work, we used generative AI tools to assist with language editing and polishing, refactor experimental code, and identify potentially relevant literature for manual review and follow-up investigation. We did not use generative AI tools to design or implement our method, autonomously select or insert citations, or draft the paper from scratch. All AI-assisted text and code suggestions, as well as literature leads, were reviewed and verified by the authors. We take responsibility for the final content of this work, including all text, claims, and artifacts produced with the assistance of generative AI.

ETHICS STATEMENT

We conduct our experiments with an emphasis on fairness, transparency, and methodological consistency. All models and datasets are evaluated using the same experimental procedures and metrics to support meaningful comparisons. We rely exclusively on publicly available datasets that contain no personally identifiable information (PII), adhere to responsible data-use practices, and report our findings accurately.

We nevertheless acknowledge that knowledge-editing techniques may be misused to introduce harmful information into LLMs or manipulate their behavior in undesirable ways (Chen et al., 2026). Conversely, prior work has explored knowledge editing as a means of improving model reliability and safety, including reducing hallucinations (Huang et al., 2025; Seo et al., 2025), strengthening safety alignment (Wang et al., 2024c), and mitigating bias (Chen et al., 2024). Our work seeks to support these beneficial applications by making sequential editing more robust while preserving the model's existing behavior.

REFERENCES

Elie Bakouch, Loubna Ben Allal, Anton Lozhkov, Nouamane Tazi, Lewis Tunstall, Carlos Miguel Patiño, Edward Beeching, Aymeric Roucher, Aksel Joonas Reedi, Quentin Gallouédec, Kashif Rasul, Nathan Habib, Clémentine Fourrier, Hynek Kydlicek, Guilherme Penedo, Hugo Larcher, Mathieu Morlon, Vaibhav Srivastav, Joshua Lochner, Xuan-Son Nguyen, Colin Raffel, Leandro von Werra, and Thomas Wolf. SmolLM3: smol, multilingual, long-context reasoner. https://huggingface.co/blog/smollm3, 2025.

Nicola De Cao, Wilker Aziz, and Ivan Titov. Editing factual knowledge in language models, 2021. URL https://arxiv.org/abs/2104.08164.

Canyu Chen, Baixiang Huang, Zekun Li, Zhaorun Chen, Shiyang Lai, Xiongxiao Xu, Jia-Chen Gu, Jindong Gu, Huaxiu Yao, Chaowei Xiao, Xifeng Yan, William Yang Wang, Philip Torr, Dawn Song, and Kai Shu. Can editing llms inject harm?, 2026. URL https://arxiv.org/abs/2407.20224.

Ruizhe Chen, Yichen Li, Zikai Xiao, and Zuozhu Liu. Large language model bias mitigation from the perspective of knowledge editing, 2024. URL https://arxiv.org/abs/2405.09341.

Jasper Dekoninck, Nikola Jovanović, Tim Gehrunger, Kári Rögnvaldsson, Ivo Petrov, Chenhao Sun, and Martin Vechev. Beyond benchmarks: Matharena as an evaluation platform for mathematics with llms. 2026. URL https://arxiv.org/abs/2605.00674.

Jingcheng Deng, Zihao Wei, Liang Pang, Hanxing Ding, Huawei Shen, and Xueqi Cheng. Everything is editable: Extend knowledge editing to unstructured data in large language models. *arXiv preprint arXiv:2405.15349*, 2024.

Nadir Durrani, Basel Mousi, and Fahim Dalvi. Editing across languages: A survey of multilingual knowledge editing. In *Proceedings of the 2025 Conference on Empirical Methods in Natural Language Processing*, pp. 15917–15929, 2025.

Junfeng Fang, Houcheng Jiang, Kun Wang, Yunshan Ma, Shi Jie, Xiang Wang, Xiangnan He, and Tat seng Chua. Alphaedit: Null-space constrained knowledge editing for language models, 2025. URL https://arxiv.org/abs/2410.02355.

Govind Krishnan Gangadhar and Karl Stratos. Model editing by standard fine-tuning. In Lun-Wei Ku, Andre Martins, and Vivek Srikumar (eds.), *Findings of the Association for Computational Linguistics: ACL 2024*, pp. 5907–5913, Bangkok, Thailand, August 2024. Association for Computational Linguistics. doi: 10.18653/v1/2024.findings-acl.352. URL https://aclanthology.org/2024.findings-acl.352/.

Tom Hartvigsen, Swami Sankaranarayanan, Hamid Palangi, Yoon Kim, and Marzyeh Ghassemi. Aging with grace: Lifelong model editing with discrete key-value adaptors. *Advances in Neural Information Processing Systems*, 36:47934–47959, 2023.

Chenhui Hu, Pengfei Cao, Yubo Chen, Kang Liu, and Jun Zhao. WilKE: Wise-layer knowledge editor for lifelong knowledge editing. In Lun-Wei Ku, Andre Martins, and Vivek Srikumar (eds.), *Findings of the Association for Computational Linguistics: ACL 2024*, pp. 3476–3503, Bangkok, Thailand, August 2024. Association for Computational Linguistics. doi: 10.18653/v1/2024.findings-acl.207. URL https://aclanthology.org/2024.findings-acl.207/.

Chenhui Hu, Pengfei Cao, Yubo Chen, Kang Liu, and Jun Zhao. Knowledge in superposition: unveiling the failures of lifelong knowledge editing for large language models. In *Proceedings of the Thirty-Ninth AAAI Conference on Artificial Intelligence and Thirty-Seventh Conference on Innovative Applications of Artificial Intelligence and Fifteenth Symposium on Educational Advances in Artificial Intelligence*, AAAI'25/IAAI'25/EAAI'25. AAAI Press, 2025. ISBN 978-1-57735-897-8. doi: 10.1609/aaai.v39i22.34583. URL https://doi.org/10.1609/aaai.v39i22.34583.

Baixiang Huang, Canyu Chen, Xiongxiao Xu, Ali Payani, and Kai Shu. Can knowledge editing really correct hallucinations?, 2025. URL https://arxiv.org/abs/2410.16251.

Naman Jain, Alex Gu, Wen-Ding Li, Fanjia Yan, Tianjun Zhang, Sida Wang, Armando Solar-Lezama, Koushik Sen, and Ion Stoica. Livecodebench: Holistic and contamination free evaluation of large language models for code. In *International Conference on Learning Representations*, volume 2025, pp. 58791–58831, 2025.

Natasha Jaques, Shixiang Gu, Dzmitry Bahdanau, José Miguel Hernández-Lobato, Richard E. Turner, and Douglas Eck. Sequence tutor: Conservative fine-tuning of sequence generation models with KL-control. In Doina Precup and Yee Whye Teh (eds.), *Proceedings of the 34th International Conference on Machine Learning*, volume 70 of *Proceedings of Machine Learning Research*, pp. 1645–1654. PMLR, 06–11 Aug 2017. URL https://proceedings.mlr.press/v70/jaques17a.html.

Houcheng Jiang, Junfeng Fang, Ningyu Zhang, Guojun Ma, Mingyang Wan, Xiang Wang, Xiangnan He, and Tat-seng Chua. Anyedit: Edit any knowledge encoded in language models. *arXiv preprint arXiv:2502.05628*, 2025.

Kimi Team, Tongtong Bai, Yifan Bai, Yiping Bao, M. C., Jianfeng Cai, Xinyuan Cai, Peizhou Cao, Yuxuan Cao, Ziwei Chai, Y. Charles, H. S. Che, Guanduo Chen, Guangyu Chen, Guanzheng Chen, Huarong Chen, Jia Chen, Jianlong Chen, Jun Chen, Kexin Chen, Peng Chen, Ruijue Chen, Wentao Chen, Xin Chen, Yang Chen, Yanru Chen, Yifei Chen, Yingjiang Chen, Yuankun Chen, Yujie Chen, Yutian Chen, Zhirong Chen, Dazhi Cheng, Yean Cheng, Jialei Cui, Jingbing Cui, Anqi Dai, Jiaqi Deng, Hao Ding, Rui Ding, Shaofeng Ding, Mengfan Dong, Mengnan Dong, Yuhao Dong, Yuxin Dong, Angang Du, Chenzhuang Du, Dikang Du, Jusen Du, Yulun Du, Yu Fan, Jing Feng, Qiulin Feng, Yichen Feng, Kelin Fu, Qiang Fu, Fuxuan Gao, Hongcheng Gao, Jingyue Gao, Tong Gao, Weijia Gao, Shangyi Geng, Jie Gong, Linhu Gong, Shengao Gong, Xiaochen Gong, Qizheng Gu, Yicheng Gu, Shuhao Guan, Haiqing Guo, Shiqi Guo, Xiang Guo, Zhengyan Guo, Beixi Hao, Wenxin Hao, Xiaoru Hao, Dailan He, Haotian He, Lehan He, Qi He, Weiran He, Xinran He, Xinyi He, Yibo He, Yunjia He, Chao Hong, Tiange Hong, Hao Hu, Jiaxi Hu, Ruikun Hu, Weiming Hu, Yangyang Hu, Zhenxing Hu, Liang Hua, Jinbin Huang, Ke Huang, Ruiyuan Huang, Siying Huang, Weixiao Huang, Yan Huang, Zhengjie Huang, Zhiqi Huang, Yulong Hui, Chaobo Jia, Yutong Jiang, Zhejun Jiang, Zuoyou Jiang, Wenyi Jin, Xinyi Jin, Yu Jing, Huanjun Kong, Guokun Lai, Aidi Li, Cheng Li, Chengyuan Li, Cong Li, Fang Li, Guanyu Li, Haoyang Li, Jia Li, Junxiong Li, Lei Li, Letian Li, Lincan Li, Weihong Li, Wentao Li, Xintong Li, Yang Li, Yishen Li, Yiwei Li, Yuxiao Li, Zhaowei Li, Zhaoxi Li, Zheming Li, Zhengxiao Li, Zhiyuan

Li, Jiawei Lin, Xiaohan Lin, Yibo Lin, Zichao Lin, Ziyan Lin, Bill Liu, Boxiao Liu, Chuan Liu, Liang Liu, Shaowei Liu, Shudong Liu, Shuran Liu, Tianwei Liu, Weizhou Liu, Yangyang Liu, Yanming Liu, Yibo Liu, Yipeng Liu, Zhengying Liu, Zhiheng Liu, Enzhe Lu, Haoyu Lu, Linqiang Lu, Tingzhan Lu, Zhiyuan Lu, Aotian Luo, G. Luo, Junyu Luo, Yifan Luo, B. Lyu, Wenzhou Lyu, Shaoguang Mao, Yuan Mei, Xin Men, Minqing Ni, Yixuan Niu, Siyuan Pan, Shujun Peng, Zhangyang Qi, Ruoyu Qin, ZeChao Qin, Zeyu Qin, Haiquan Qiu, Jianxin Qiu, Jiezhong Qiu, Bowen Qu, Yuhao Qu, Zeyu Shang, Youbo Shao, Han Shen, Jincheng Shi, Juanfeng Shi, Lidong Shi, Shengyuan Shi, Wingchun Siu, Pengwei Song, Xiaoxi Song, Jianlin Su, Yunfeng Su, Zhaochen Su, Lin Sui, Jingsong Sun, Junyao Sun, Shaoning Sun, Shuzhe Sun, Tongyu Sun, Yujun Sun, Yunpeng Tai, Chuning Tang, Heyi Tang, Sirui Tang, Zecheng Tang, Chaoran Tian, Rongpeng Tian, Yu Tian, Wei Tu, Chensi Wang, Chuang Wang, Chunjie Wang, Dinglu Wang, Feng Wang, Hailong Wang, Haiming Wang, Hao Wang, Hao Wang, Huaqing Wang, Hui Wang, Jiayi Wang, Jinglong Wang, Jinhong Wang, Jiuzheng Wang, Linian Wang, Shaobo Wang, Shenzhi Wang, Shuyi Wang, Si Wang, Siyuan Wang, Tianfu Wang, Wenjue Wang, Xingran Wang, Xinmei Wang, Xinyuan Wang, Xusheng Wang, Yalin Wang, Yangkun Wang, Yao Wang, Yaoyu Wang, Yejie Wang, Yiqin Wang, Yucheng Wang, Yuzhi Wang, Zhaoji Wang, Zhaowei Wang, Zhengtao Wang, Zhenhao Wang, Zhongsheng Wang, Zifan Wang, Chu Wei, Ming Wei, Shouxin Wei, Zichen Wen, Fan Wu, Haoning Wu, Rucong Wu, Wenhao Wu, Xiaoxue Wu, Yingcong Wu, Yongqi Wu, Yuxin Wu, Zijian Wu, Xinglang Xian, Chenxuan Xiang, Yuye Xiang, Bocheng Xiao, Chenjun Xiao, Xin Xiao, Jin Xie, Xiaotong Xie, Yifeng Xie, Zhe Xie, Bowei Xing, Yiming Xiong, Baosheng Xu, Boyu Xu, Jiale Xu, Jianfan Xu, Jing Xu, Jinjing Xu, L. H. Xu, Qingtao Xu, Shuyao Xu, Suting Xu, Tiantian Xu, Tianxiang Xu, Weixin Xu, Xinran Xu, Yangchuan Xu, Ye Xu, Yueni Xu, Ziyao Xu, Haonan Xue, Junjie Yan, Yaoyao Yan, Fan Yang, Guangyao Yang, Hao Yang, Junwei Yang, Ruoyu Yang, Wenjie Yang, Xiaofei Yang, Xinyu Yang, Yi Yang, Yiling Yang, Ying Yang, Yuchen Yang, Zhen Yang, Zhilin Yang, Zian Yang, Zuhao Yang, Haotian Yao, Dan Ye, Haoran Ye, Wenjie Ye, Zhanbo Ye, Bohong Yin, Haoxiang Yin, Xietong Yin, Chengzhen Yu, Haozhen Yu, Longhui Yu, Shengnan Yu, Shuying Yu, Tianxiang Yu, Enming Yuan, Mengjie Yuan, Tongtian Yue, Wei Yue, Yang Yue, Dunyuan Zha, Haobing Zhan, B. H. Zhang, Dehao Zhang, Fei Zhang, Hao Zhang, Haoyuan Zhang, Huanyu Zhang, Jiapei Zhang, Jiaxuan Zhang, Jin Zhang, Kaiyi Zhang, Miaozhen Zhang, Puqi Zhang, Qinglei Zhang, Rong Zhang, Rui Zhang, Shaoshuai Zhang, Shiyi Zhang, Xiaobin Zhang, Xiaoyun Zhang, Y. Zhang, Yangkun Zhang, Ye Zhang, Yichi Zhang, Yikun Zhang, Yizhi Zhang, Yongting Zhang, Yu Zhang, Yutao Zhang, Yutong Zhang, Zheng Zhang, Zijing Zhang, Bin Zhao, Chenguang Zhao, Feifan Zhao, Jinglun Zhao, Jinxiang Zhao, Shuai Zhao, Wenshuo Zhao, Xiangyu Zhao, Xuanle Zhao, Yikai Zhao, Zijia Zhao, Haozhi Zheng, Huabin Zheng, Ruihan Zheng, Shaojie Zheng, Tengyang Zheng, Haofeng Zhong, Lei Zhong, Longguang Zhong, M. Zhou, Qiankang Zhou, Runjie Zhou, Ruozhang Zhou, Xinyu Zhou, Yiqiao Zhou, Zaida Zhou, Jinguo Zhu, Liya Zhu, Xinhao Zhu, Yangjunfeng Zhu, Yuxuan Zhu, Zhen Zhu, Chen Zhuang, Weiyu Zhuang, and Xinxing Zu. Kimi k3: Open frontier intelligence, 2026. URL https://arxiv.org/abs/2607.24653.

Woosuk Kwon, Zhuohan Li, Siyuan Zhuang, Ying Sheng, Lianmin Zheng, Cody Hao Yu, Joseph E. Gonzalez, Hao Zhang, and Ion Stoica. Efficient memory management for large language model serving with pagedattention, 2023. URL https://arxiv.org/abs/2309.06180.

Jungseob Lee, Seungyoon Lee, Suhyune Son, Dongyub Jude Lee, Sungbin Han, Sugyeong Eo, and Heuiseok Lim. Answer-conditioned chains of thought degrade verifiable-reasoning distillation in large language models, 2026. URL https://arxiv.org/abs/2607.14552.

Omer Levy, Minjoon Seo, Eunsol Choi, and Luke Zettlemoyer. Zero-shot relation extraction via reading comprehension. *arXiv preprint arXiv:1706.04115*, 2017.

Shuaike Li, Kai Zhang, Xianquan Wang, Jiachen Liu, and Shengpeng Mo. From fact overwriting to knowledge evolution: Causal editing via on-policy self-distillation. *arXiv preprint arXiv:2605.28303*, 2026a.

Xiaopeng Li, Shasha Li, Shezheng Song, Jing Yang, Jun Ma, and Jie Yu. Pmet: Precise model editing in a transformer, 2024a. URL https://arxiv.org/abs/2308.08742.

Yaxuan Li, Yuxin Zuo, Bingxiang He, Jinqian Zhang, Chaojun Xiao, Cheng Qian, Tianyu Yu, Huan ang Gao, Wenkai Yang, Zhiyuan Liu, and Ning Ding. Rethinking on-policy distillation of large

language models: Phenomenology, mechanism, and recipe, 2026b. URL https://arxiv.org/abs/2604.13016.

Zherui Li, Houcheng Jiang, Hao Chen, Baolong Bi, Zhenhong Zhou, Fei Sun, Junfeng Fang, and Xiang Wang. Reinforced lifelong editing for language models, 2025. URL https://arxiv.org/abs/2502.05759.

Zichao Li, Yanshuai Cao, and Jackie C.K. Cheung. Do llms build world representations? probing through the lens of state abstraction. In A. Globerson, L. Mackey, D. Belgrave, A. Fan, U. Paquet, J. Tomczak, and C. Zhang (eds.), *Advances in Neural Information Processing Systems*, volume 37, pp. 98009–98032. Curran Associates, Inc., 2024b. doi: 10.52202/079017-3110. URL https://proceedings.neurips.cc/paper_files/paper/2024/file/b1b16c4b875eb84d3585cb70d23970ca-Paper-Conference.pdf.

Kevin Lu and Thinking Machines Lab. On-policy distillation. *Thinking Machines Lab: Connectionism*, 2025. doi: 10.64434/tml.20251026. https://thinkingmachines.ai/blog/on-policy-distillation.

Yun Luo, Zhen Yang, Fandong Meng, Yafu Li, Jie Zhou, and Yue Zhang. An empirical study of catastrophic forgetting in large language models during continual fine-tuning, 2025. URL https://arxiv.org/abs/2308.08747.

Jun-Yu Ma, Hong Wang, Hao-Xiang Xu, Zhen-Hua Ling, and Jia-Chen Gu. Perturbation-restrained sequential model editing, 2025. URL https://arxiv.org/abs/2405.16821.

Kevin Meng, David Bau, Alex Andonian, and Yonatan Belinkov. Locating and editing factual associations in gpt. *Advances in neural information processing systems*, 35:17359–17372, 2022.

Kevin Meng, Arnab Sen Sharma, Alex Andonian, Yonatan Belinkov, and David Bau. Mass-editing memory in a transformer, 2023. URL https://arxiv.org/abs/2210.07229.

Eric Mitchell, Charles Lin, Antoine Bosselut, Chelsea Finn, and Christopher D. Manning. Fast model editing at scale, 2022a. URL https://arxiv.org/abs/2110.11309.

Eric Mitchell, Charles Lin, Antoine Bosselut, Christopher D. Manning, and Chelsea Finn. Memory-based model editing at scale, 2022b. URL https://arxiv.org/abs/2206.06520.

Long Ouyang, Jeff Wu, Xu Jiang, Diogo Almeida, Carroll L. Wainwright, Pamela Mishkin, Chong Zhang, Sandhini Agarwal, Katarina Slama, Alex Ray, John Schulman, Jacob Hilton, Fraser Kelton, Luke Miller, Maddie Simens, Amanda Askell, Peter Welinder, Paul Christiano, Jan Leike, and Ryan Lowe. Training language models to follow instructions with human feedback, 2022. URL https://arxiv.org/abs/2203.02155.

Shishir G Patil, Huanzhi Mao, Fanjia Yan, Charlie Cheng-Jie Ji, Vishnu Suresh, Ion Stoica, and Joseph E Gonzalez. The berkeley function calling leaderboard (bfcl): From tool use to agentic evaluation of large language models. In *Forty-second International Conference on Machine Learning*, 2025.

Sebastian Pohl, Max Ploner, and Alan Akbik. Towards a principled evaluation of knowledge editors. In *Proceedings of the First Workshop on Large Language Model Memorization (L2M2)*, pp. 47–60, 2025.

Siyuan Qi, Bangcheng Yang, Kailin Jiang, Xiaobo Wang, Jiaqi Li, Yifan Zhong, Yaodong Yang, and Zilong Zheng. In-context editing: Learning knowledge from self-induced distributions, 2025. URL https://arxiv.org/abs/2406.11194.

Rafael Rafailov, Archit Sharma, Eric Mitchell, Christopher D Manning, Stefano Ermon, and Chelsea Finn. Direct preference optimization: Your language model is secretly a reward model. *Advances in neural information processing systems*, 36:53728–53741, 2023.

David Rein, Betty Li Hou, Asa Cooper Stickland, Jackson Petty, Richard Yuanzhe Pang, Julien Dirani, Julian Michael, and Samuel R Bowman. Gpqa: A graduate-level google-proof q&a benchmark. *arXiv preprint arXiv:2311.12022*, 2023.

John Schulman, Sergey Levine, Philipp Moritz, Michael I. Jordan, and Pieter Abbeel. Trust region policy optimization, 2015. URL https://arxiv.org/abs/1502.05477.

John Schulman, Filip Wolski, Prafulla Dhariwal, Alec Radford, and Oleg Klimov. Proximal policy optimization algorithms, 2017. URL https://arxiv.org/abs/1707.06347.

Jaehyung Seo, Hyeonseok Moon, and Heui-Seok Lim. The impact of negated text on hallucination with large language models. In *Proceedings of the 2025 Conference on Empirical Methods in Natural Language Processing*, pp. 13565–13583, 2025.

Zhihong Shao, Peiyi Wang, Qihao Zhu, Runxin Xu, Junxiao Song, Xiao Bi, Haowei Zhang, Mingchuan Zhang, YK Li, Yang Wu, et al. Deepseekmath: Pushing the limits of mathematical reasoning in open language models. *arXiv preprint arXiv:2402.03300*, 2024.

Idan Shenfeld, Mehul Damani, Jonas Hübotter, and Pulkit Agrawal. Self-distillation enables continual learning, 2026. URL https://arxiv.org/abs/2601.19897.

Aaditya Singh, Adam Fry, Adam Perelman, Adam Tart, Adi Ganesh, Ahmed El-Kishky, Aidan McLaughlin, Aiden Low, AJ Ostrow, Akhila Ananthram, Akshay Nathan, Alan Luo, Alec Helyar, Aleksander Madry, Aleksandr Efremov, Aleksandra Spyra, Alex Baker-Whitcomb, Alex Beutel, Alex Karpenko, Alex Makelov, Alex Neitz, Alex Wei, Alexandra Barr, Alexandre Kirchmeyer, Alexey Ivanov, Alexi Christakis, Alistair Gillespie, Allison Tam, Ally Bennett, Alvin Wan, Alyssa Huang, Amy McDonald Sandjideh, Amy Yang, Ananya Kumar, Andre Saraiva, Andrea Vallone, Andrei Gheorghe, Andres Garcia Garcia, Andrew Braunstein, Andrew Liu, Andrew Schmidt, Andrey Mereskin, Andrey Mishchenko, Andy Applebaum, Andy Rogerson, Ann Rajan, Annie Wei, Anoop Kotha, Anubha Srivastava, Anushree Agrawal, Arun Vijayvergiya, Ashley Tyra, Ashvin Nair, Avi Nayak, Ben Eggers, Bessie Ji, Beth Hoover, Bill Chen, Blair Chen, Boaz Barak, Borys Minaiev, Botao Hao, Bowen Baker, Brad Lightcap, Brandon McKinzie, Brandon Wang, Brendan Quinn, Brian Fioca, Brian Hsu, Brian Yang, Brian Yu, Brian Zhang, Brittany Brenner, Callie Riggins Zetino, Cameron Raymond, Camillo Lugaresi, Carolina Paz, Cary Hudson, Cedric Whitney, Chak Li, Charles Chen, Charlotte Cole, Chelsea Voss, Chen Ding, Chen Shen, Chengdu Huang, Chris Colby, Chris Hallacy, Chris Koch, Chris Lu, Christina Kaplan, Christina Kim, CJ Minott-Henriques, Cliff Frey, Cody Yu, Coley Czarnecki, Colin Reid, Colin Wei, Cory Decareaux, Cristina Scheau, Cyril Zhang, Cyrus Forbes, Da Tang, Dakota Goldberg, Dan Roberts, Dana Palmie, Daniel Kappler, Daniel Levine, Daniel Wright, Dave Leo, David Lin, David Robinson, Declan Grabb, Derek Chen, Derek Lim, Derek Salama, Dibya Bhattacharjee, Dimitris Tsipras, Dinghua Li, Dingli Yu, DJ Strouse, Drew Williams, Dylan Hunn, Ed Bayes, Edwin Arbus, Ekin Akyurek, Elaine Ya Le, Elana Widmann, Eli Yani, Elizabeth Proehl, Enis Sert, Enoch Cheung, Eri Schwartz, Eric Han, Eric Jiang, Eric Mitchell, Eric Sigler, Eric Wallace, Erik Ritter, Erin Kavanaugh, Evan Mays, Evgenii Nikishin, Fangyuan Li, Felipe Petroski Such, Filipe de Avila Belbute Peres, Filippo Raso, Florent Bekerman, Foivos Tsimpourlas, Fotis Chantzis, Francis Song, Francis Zhang, Gaby Raila, Garrett McGrath, Gary Briggs, Gary Yang, Giambattista Parascandolo, Gildas Chabot, Grace Kim, Grace Zhao, Gregory Valiant, Guillaume Leclerc, Hadi Salman, Hanson Wang, Hao Sheng, Haoming Jiang, Haoyu Wang, Haozhun Jin, Harshit Sikchi, Heather Schmidt, Henry Aspegren, Honglin Chen, Huida Qiu, Hunter Lightman, Ian Covert, Ian Kivlichan, Ian Silber, Ian Sohl, Ibrahim Hammoud, Ignasi Clavera, Ikai Lan, Ilge Akkaya, Ilya Kostrikov, Irina Kofman, Isak Etinger, Ishaan Singal, Jackie Hehir, Jacob Huh, Jacqueline Pan, Jake Wilczynski, Jakub Pachocki, James Lee, James Quinn, Jamie Kiros, Janvi Kalra, Jasmyn Samaroo, Jason Wang, Jason Wolfe, Jay Chen, Jay Wang, Jean Harb, Jeffrey Han, Jeffrey Wang, Jennifer Zhao, Jeremy Chen, Jerene Yang, Jerry Tworek, Jesse Chand, Jessica Landon, Jessica Liang, Ji Lin, Jiancheng Liu, Jianfeng Wang, Jie Tang, Jihan Yin, Joanne Jang, Joel Morris, Joey Flynn, Johannes Ferstad, Johannes Heidecke, John Fishbein, John Hallman, Jonah Grant, Jonathan Chien, Jonathan Gordon, Jongsoo Park, Jordan Liss, Jos Kraaijeveld, Joseph Guay, Joseph Mo, Josh Lawson, Josh McGrath, Joshua Vendrow, Joy Jiao, Julian Lee, Julie Steele, Julie Wang, Junhua Mao, Kai Chen, Kai Hayashi, Kai Xiao, Kamyar Salahi, Kan Wu, Karan Sekhri, Karan Sharma, Karan Singhal, Karen Li, Kenny Nguyen, Keren Gu-Lemberg, Kevin King, Kevin Liu, Kevin Stone, Kevin Yu, Kristen Ying, Kristian Georgiev, Kristie Lim, Kushal Tirumala, Kyle Miller, Lama Ahmad, Larry Lv, Laura Clare, Laurance Fauconnet, Lauren Itow, Lauren Yang, Laurentia Romaniuk, Leah Anise, Lee Byron, Leher Pathak, Leon Maksin, Leyan Lo, Leyton Ho, Li Jing, Liang Wu, Liang Xiong, Lien Mamitsuka, Lin Yang, Lindsay McCallum, Lindsey Held,

Liz Bourgeois, Logan Engstrom, Lorenz Kuhn, Louis Feuvrier, Lu Zhang, Lucas Switzer, Lukas Kondraciuk, Lukasz Kaiser, Manas Joglekar, Mandeep Singh, Mandip Shah, Manuka Stratta, Marcus Williams, Mark Chen, Mark Sun, Marselus Cayton, Martin Li, Marvin Zhang, Marwan Aljubeh, Matt Nichols, Matthew Haines, Max Schwarzer, Mayank Gupta, Meghan Shah, Melody Y. Guan, Melody Huang, Meng Dong, Mengqing Wang, Mia Glaese, Micah Carroll, Michael Lampe, Michael Malek, Michael Sharman, Michael Zhang, Michele Wang, Michelle Pokrass, Mihai Florian, Mikhail Pavlov, Miles Wang, Ming Chen, Mingxuan Wang, Minnia Feng, Mo Bavarian, Molly Lin, Moose Abdool, Mostafa Rohaninejad, Nacho Soto, Natalie Staudacher, Natan LaFontaine, Nathan Marwell, Nelson Liu, Nick Preston, Nick Turley, Nicklas Ansman, Nicole Blades, Nikil Pancha, Nikita Mikhaylin, Niko Felix, Nikunj Handa, Nishant Rai, Nitish Keskar, Noam Brown, Ofir Nachum, Oleg Boiko, Oleg Murk, Olivia Watkins, Oona Gleeson, Pamela Mishkin, Patryk Lesiewicz, Paul Baltescu, Pavel Belov, Peter Zhokhov, Philip Pronin, Phillip Guo, Phoebe Thacker, Qi Liu, Qiming Yuan, Qinghua Liu, Rachel Dias, Rachel Puckett, Rahul Arora, Ravi Teja Mullapudi, Raz Gaon, Reah Miyara, Rennie Song, Rishabh Aggarwal, RJ Marsan, Robel Yemiru, Robert Xiong, Rohan Kshirsagar, Rohan Nuttall, Roman Tsiupa, Ronen Eldan, Rose Wang, Roshan James, Roy Ziv, Rui Shu, Ruslan Nigmatullin, Saachi Jain, Saam Talaie, Sam Altman, Sam Arnesen, Sam Toizer, Sam Toyer, Samuel Miserendino, Sandhini Agarwal, Sarah Yoo, Savannah Heon, Scott Ethersmith, Sean Grove, Sean Taylor, Sebastien Bubeck, Sever Banesiu, Shaokyi Amdo, Shengjia Zhao, Sherwin Wu, Shibani Santurkar, Shiyu Zhao, Shraman Ray Chaudhuri, Shreyas Krishnaswamy, Shuaiqi, Xia, Shuyang Cheng, Shyamal Anadkat, Simón Posada Fishman, Simon Tobin, Siyuan Fu, Somay Jain, Song Mei, Sonya Egoian, Spencer Kim, Spug Golden, SQ Mah, Steph Lin, Stephen Imm, Steve Sharpe, Steve Yadlowsky, Sulman Choudhry, Sungwon Eum, Suvansh Sanjeev, Tabarak Khan, Tal Stramer, Tao Wang, Tao Xin, Tarun Gogineni, Taya Christianson, Ted Sanders, Tejal Patwardhan, Thomas Degry, Thomas Shadwell, Tianfu Fu, Tianshi Gao, Timur Garipov, Tina Sriskandarajah, Toki Sherbakov, Tomek Korbak, Tomer Kaftan, Tomo Hiratsuka, Tongzhou Wang, Tony Song, Tony Zhao, Troy Peterson, Val Kharitonov, Victoria Chernova, Vineet Kosaraju, Vishal Kuo, Vitchyr Pong, Vivek Verma, Vlad Petrov, Wanning Jiang, Weixing Zhang, Wenda Zhou, Wenlei Xie, Wenting Zhan, Wes McCabe, Will DePue, Will Ellsworth, Wulfie Bain, Wyatt Thompson, Xiangning Chen, Xiangyu Qi, Xin Xiang, Xinwei Shi, Yann Dubois, Yaodong Yu, Yara Khakbaz, Yifan Wu, Yilei Qian, Yin Tat Lee, Yinbo Chen, Yizhen Zhang, Yizhong Xiong, Yonglong Tian, Young Cha, Yu Bai, Yu Yang, Yuan Yuan, Yuanzhi Li, Yufeng Zhang, Yuguang Yang, Yujia Jin, Yun Jiang, Yunyun Wang, Yushi Wang, Yutian Liu, Zach Stubenvoll, Zehao Dou, Zheng Wu, and Zhigang Wang. Openai gpt-5 system card, 2026. URL https://arxiv.org/abs/2601.03267.

Zian Su, Ziyang Huang, Kaiyuan Zhang, and Xiangyu Zhang. μke: Matryoshka unstructured knowledge editing of large language models, 2025. URL https://arxiv.org/abs/2504.01196.

Kai Sun, Yifan Xu, Hanwen Zha, Yue Liu, and Xin Luna Dong. Head-to-tail: How knowledgeable are large language models (llms)? aka will llms replace knowledge graphs? In *Proceedings of the 2024 conference of the North American chapter of the association for computational linguistics: human language technologies (volume 1: long papers)*, pp. 311–325, 2024.

Haoyu Wang, Tianci Liu, Ruirui Li, Monica Xiao Cheng, Tuo Zhao, and Jing Gao. Roselora: Row and column-wise sparse low-rank adaptation of pre-trained language model for knowledge editing and fine-tuning. In *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing*, pp. 996–1008, 2024a.

Jiaan Wang, Yunlong Liang, Zengkui Sun, Yuxuan Cao, Jiarong Xu, and Fandong Meng. Cross-lingual knowledge editing in large language models. In Lun-Wei Ku, Andre Martins, and Vivek Srikumar (eds.), *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 11676–11686, Bangkok, Thailand, August 2024b. Association for Computational Linguistics. doi: 10.18653/v1/2024.acl-long.627. URL https://aclanthology.org/2024.acl-long.627/.

Mengru Wang, Ningyu Zhang, Ziwen Xu, Zekun Xi, Shumin Deng, Yunzhi Yao, Qishen Zhang, Linyi Yang, Jindong Wang, and Huajun Chen. Detoxifying large language models via knowledge editing, 2024c. URL https://arxiv.org/abs/2403.14472.

Peng Wang, Zexi Li, Ningyu Zhang, Ziwen Xu, Yunzhi Yao, Yong Jiang, Pengjun Xie, Fei Huang, and Huajun Chen. Wise: Rethinking the knowledge memory for lifelong model editing of large language models. *Advances in Neural Information Processing Systems*, 37:53764–53797, 2024d.

Peng Wang, Ningyu Zhang, Bozhong Tian, Zekun Xi, Yunzhi Yao, Ziwen Xu, Mengru Wang, Shengyu Mao, Xiaohan Wang, Siyuan Cheng, Kangwei Liu, Yuansheng Ni, Guozhou Zheng, and Huajun Chen. EasyEdit: An easy-to-use knowledge editing framework for large language models. In Yixin Cao, Yang Feng, and Deyi Xiong (eds.), *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 3: System Demonstrations)*, pp. 82–93, Bangkok, Thailand, August 2024e. Association for Computational Linguistics. doi: 10.18653/v1/2024.acl-demos.9. URL https://aclanthology.org/2024.acl-demos.9/.

Song Wang, Yaochen Zhu, Haochen Liu, Zaiyi Zheng, Chen Chen, and Jundong Li. Knowledge editing for large language models: A survey, 2024f. URL https://arxiv.org/abs/2310.16218.

Jason Wei, Nguyen Karina, Hyung Won Chung, Yunxin Joy Jiao, Spencer Papay, Amelia Glaese, John Schulman, and William Fedus. Measuring short-form factuality in large language models, 2024. URL https://arxiv.org/abs/2411.04368.

Ronald J Williams. Simple statistical gradient-following algorithms for connectionist reinforcement learning. *Machine learning*, 8(3):229–256, 1992.

Anyi Xu, Bangcai Lin, Bing Xue, Bingxuan Wang, Bingzheng Xu, Bochao Wu, Bowei Zhang, Chaofan Lin, Chen Dong, Chenchen Ling, et al. Deepseek-v4: Towards highly efficient million-token context intelligence. *arXiv preprint arXiv:2606.19348*, 2026.

An Yang, Anfeng Li, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chang Gao, Chengen Huang, Chenxu Lv, Chujie Zheng, Dayiheng Liu, Fan Zhou, Fei Huang, Feng Hu, Hao Ge, Haoran Wei, Huan Lin, Jialong Tang, Jian Yang, Jianhong Tu, Jianwei Zhang, Jianxin Yang, Jiaxi Yang, Jing Zhou, Jingren Zhou, Junyang Lin, Kai Dang, Keqin Bao, Kexin Yang, Le Yu, Lianghao Deng, Mei Li, Mingfeng Xue, Mingze Li, Pei Zhang, Peng Wang, Qin Zhu, Rui Men, Ruize Gao, Shixuan Liu, Shuang Luo, Tianhao Li, Tianyi Tang, Wenbiao Yin, Xingzhang Ren, Xinyu Wang, Xinyu Zhang, Xuancheng Ren, Yang Fan, Yang Su, Yichang Zhang, Yinger Zhang, Yu Wan, Yuqiong Liu, Zekun Wang, Zeyu Cui, Zhenru Zhang, Zhipeng Zhou, and Zihan Qiu. Qwen3 technical report, 2025a. URL https://arxiv.org/abs/2505.09388.

Wanli Yang, Fei Sun, Jiajun Tan, Xinyu Ma, Qi Cao, Dawei Yin, Huawei Shen, and Xueqi Cheng. The mirage of model editing: Revisiting evaluation in the wild. In *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 15336–15354, 2025b.

Wanli Yang, Fei Sun, Rui Tang, Hongyu Zang, Du Su, Qi Cao, Jingang Wang, Huawei Shen, and Xueqi Cheng. Fine-tuning done right in model editing. *arXiv preprint arXiv:2509.22072*, 2025c.

Lang Yu, Qin Chen, Jie Zhou, and Liang He. Melo: Enhancing model editing with neuron-indexed dynamic lora. *Proceedings of the AAAI Conference on Artificial Intelligence*, 38(17): 19449–19457, Mar. 2024. doi: 10.1609/aaai.v38i17.29916. URL https://ojs.aaai.org/index.php/AAAI/article/view/29916.

Li Zeng, Yingyu Shan, Zeming Liu, Jiashu Yao, and Yuhang Guo. Fame: Towards factual multi-task model editing, 2024. URL https://arxiv.org/abs/2410.10859.

Ningyu Zhang, Bozhong Tian, Siyuan Cheng, Xiaozhuan Liang, Yi Hu, Kouying Xue, Yanjie Gou, Xi Chen, and Huajun Chen. Instructedit: Instruction-based knowledge editing for large language models, 2024a. URL https://arxiv.org/abs/2402.16123.

Ningyu Zhang, Yunzhi Yao, Bozhong Tian, Peng Wang, Shumin Deng, Mengru Wang, Zekun Xi, Shengyu Mao, Jintian Zhang, Yuansheng Ni, et al. A comprehensive study of knowledge editing for large language models. *arXiv preprint arXiv:2401.01286*, 2024b.

Ce Zheng, Lei Li, Qingxiu Dong, Yuxuan Fan, Zhiyong Wu, Jingjing Xu, and Baobao Chang. Can we edit factual knowledge by in-context learning? In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pp. 4862–4876, 2023.

Lianmin Zheng, Liangsheng Yin, Zhiqiang Xie, Chuyue Sun, Jeff Huang, Cody H Yu, Shiyi Cao, Christos Kozyrakis, Ion Stoica, Joseph E Gonzalez, et al. Sglang: Efficient execution of structured language model programs. *Advances in neural information processing systems*, 37:62557–62583, 2024a.

Yaowei Zheng, Richong Zhang, Junhao Zhang, Yanhan Ye, and Zheyan Luo. LlamaFactory: Unified efficient fine-tuning of 100+ language models. In Yixin Cao, Yang Feng, and Deyi Xiong (eds.), *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 3: System Demonstrations)*, pp. 400–410, Bangkok, Thailand, August 2024b. Association for Computational Linguistics. doi: 10.18653/v1/2024.acl-demos.38. URL https://aclanthology.org/2024.acl-demos.38/.

Zexuan Zhong, Zhengxuan Wu, Christopher D. Manning, Christopher Potts, and Danqi Chen. Mquake: Assessing knowledge editing in language models via multi-hop questions, 2024. URL https://arxiv.org/abs/2305.14795.

Jeffrey Zhou, Tianjian Lu, Swaroop Mishra, Siddhartha Brahma, Sujoy Basu, Yi Luan, Denny Zhou, and Le Hou. Instruction-following evaluation for large language models, 2023. URL https://arxiv.org/abs/2311.07911.

A Experimental Details

This section describes the experimental setup and implementation details.

Implementation details. For FT-M, AlphaEdit, WISE, and IKE, we use the implementations provided by `EasyEdit`[2] (Wang et al., 2024e). For LocFT, ICE, and RLEdit, we use the official implementations released by their authors. We evaluate general capabilities using `NeMo Gym`[3] and accelerate inference with `vLLM`. All experiments are conducted on six NVIDIA A6000 GPUs, each with 48 GB of memory.

Evaluation settings. For the LLM-as-a-judge evaluation, we follow the SimpleQA protocol (Wei et al., 2024) and use GPT-5 mini (Singh et al., 2026) as the judge model. The judge prompt is provided in Table 9. We report each score as the percentage of generated responses classified as `CORRECT` by the judge. We use deterministic decoding with a temperature of 0 for all evaluations. For the evaluation of knowledge editing, we disable reasoning mode and allow up to 128 newly generated tokens for both BaseEval and ChatEval. For the general-capability benchmarks, we enable reasoning mode and use task-specific generation limits. We allow up to 16,384 tokens for AIME 2025, GPQA-Diamond, and IFEval, 8,192 tokens for LiveCodeBench-v6, and 4,096 tokens for BFCL-v4.

Baseline settings. Following prior work (Yang et al., 2025c), we use layer 6 as the target update layer for gradient-based and locate-and-edit methods. For WISE and RLEdit, we map the layers used in the original papers to our models by preserving their relative positions within the network depth. For each baseline, we reproduce the hyperparameter settings reported in the corresponding paper as closely as possible. Table 10 summarizes the key hyperparameters used in our experiments.

For the experiments in Section 7.2, we follow the settings reported in the corresponding papers and use the authors' official GitHub implementations.

AlignEdit settings. For each edit batch, we initialize a teacher from the original model and fine-tune only layer 6 using the FT-M configuration reported in Table 10. The same table lists the hyperparameters used for the subsequent AlignEdit distillation. At every student-generated prefix, we compute the token-level reverse-KL loss over the full vocabulary instead of using Monte Carlo or top-k approximations (Li et al., 2026b). This full-vocabulary computation yields the exact categorical KL and its gradient conditional on the fixed prefix, avoiding additional vocabulary-level approximation error. The generated trajectories are treated as constants during backpropagation,

[2] https://github.com/zjunlp/EasyEdit

[3] https://github.com/NVIDIA-NeMo/gym

Table 4: Knowledge-editing performance on UnKEBench. Reliability and Generality are evaluated using ROUGE-L and BERTScore, while Locality is evaluated using ROUGE-L.

Model	Method	Reliability		Generality		Locality
		ROUGE-L	BERTScore	ROUGE-L	BERTScore	ROUGE-L
Qwen3-4B	Pre-edited	26.97	71.71	25.76	71.09	23.14
	AnyEdit	28.92	63.92	27.86	63.34	22.04
	μKE	33.15	70.27	30.96	68.59	24.65
	AlignEdit	**38.46**	**76.88**	**36.36**	**75.29**	**30.81**
SmolLM3-3B	Pre-edited	27.60	72.68	26.56	71.97	26.52
	AnyEdit	13.29	8.88	12.95	7.27	11.77
	μKE	13.27	5.44	12.56	4.86	14.17
	AlignEdit	**42.64**	**77.37**	**39.44**	**76.23**	**33.27**

Table 5: General-capability performance after editing on UnKEBench.

Model	Method	AIME25	GPQA	IFEval	LCB	BFCL	Avg.
Qwen3-4B	Pre-edited	43.33	52.02	81.15	30.18	82.81	57.90
	AnyEdit	5.56	21.72	66.54	1.76	64.02	31.92
	μKE	16.67	4.55	52.50	6.39	46.44	25.31
	AlignEdit	**46.67**	**51.01**	**78.56**	**33.26**	**79.95**	**57.89**
SmolLM3-3B	Pre-edited	31.11	39.90	65.80	14.98	54.55	41.27
	AnyEdit	0.00	0.00	14.97	0.00	30.87	9.17
	μKE	0.00	0.00	14.97	0.00	30.87	9.17
	AlignEdit	**32.22**	**39.90**	**57.67**	**14.54**	**60.09**	**40.88**

and gradients are not propagated through the discrete generation process. For student trajectory generation, we use a sampling temperature of 1.0 and top-p sampling with $p = 0.95$, following prior work that reports limited sensitivity to these sampling hyperparameters (Shenfeld et al., 2026).

Adapting CounterFact for post-trained models. CounterFact is not directly suited to post-trained models because it uses completion-style prompts. For example, an input may be x = "Reedy Glacier is in" with the target answer y = "Asia". We therefore convert each input into a question-answering format, rewriting the example as x = "On which continent is Reedy Glacier located?" while retaining the same target answer. To perform this conversion, we provide Qwen3.6-27B with the original input–target pair (x, y) and instruct it to rewrite x as a question whose answer is y. We apply this conversion to all prompts used in the Reliability, Generality, and Locality evaluations. Because the converted examples follow a simple format, the authors manually review every example and correct any identified errors. This human verification process yields a final dataset with no observed conversion errors.

B Qualitative Analysis

We further assess post-editing model behavior through a qualitative analysis of the generated responses. Tables 11 and 12 present responses from the edited Qwen3-4B and SmolLM3-3B models, respectively.

Across these examples, AlignEdit is the only method that consistently produces the intended target answers for both prompts across both models while maintaining coherent and fluent responses. In contrast, the baselines exhibit several recurring failure modes. FT-M, LocFT, and RLEdit often return incorrect answers or fail to generate usable responses, whereas ICE, AlphaEdit, and WISE produce repetitive, malformed, empty, or irrelevant outputs. These failures indicate substantial degradation in generation quality after sequential editing. Overall, the examples suggest that AlignEdit more effectively incorporates the edited knowledge while preserving the response behavior acquired during post-training.

C Unstructured Knowledge Editing

In this section, we evaluate AlignEdit on unstructured knowledge editing, where the target knowledge is expressed as free-form text rather than structured fact tuples. This setting is well suited to AlignEdit: the teacher can internalize unstructured content through fine-tuning, while the student receives dense token-level supervision along student-generated trajectories. We compare AlignEdit with AnyEdit (Jiang et al., 2025) and μKE (Su et al., 2025) on UnKEBench (Deng et al., 2024) under a sequential editing protocol, unlike the evaluation protocols used in prior work (Deng et al., 2024; Jiang et al., 2025; Su et al., 2025). Since unstructured knowledge may span multiple phrases or paragraphs, we evaluate Reliability and Generality using both ROUGE-L and BERTScore, and Locality using ROUGE-L, following prior work on unstructured knowledge editing (Deng et al., 2024; Jiang et al., 2025).

Table 6: Reasoning-enabled knowledge-editing performance on structured and unstructured benchmarks.

(a) Structured knowledge editing.

Dataset	Method	Qwen3-4B				SmolLM3-3B			
		Reliability	Generality	Locality	Score	Reliability	Generality	Locality	Score
CounterFact	Pre-edited	2.90	2.65	–	–	2.55	2.80	–	–
	FT-M	2.75	2.50	99.85	3.88	4.20	3.75	99.92	5.83
	ICE	0.00	0.00	0.03	0.00	3.90	3.30	99.92	5.27
	LocFT	9.00	8.30	76.88	12.26	46.85	39.00	32.75	38.70
	AlphaEdit	22.30	18.20	97.42	27.26	9.95	8.50	0.00	0.00
	RLEdit	3.10	3.10	99.83	4.58	3.10	3.00	**99.95**	4.51
	AlignEdit	**66.25**	**50.35**	**99.92**	**66.72**	**56.30**	**52.00**	**99.95**	**63.83**
ZsRE	Pre-edited	5.25	5.05	–	–	4.75	4.15	–	–
	FT-M	3.15	3.55	58.74	4.87	8.35	7.30	80.45	11.15
	ICE	0.00	0.00	58.94	0.00	7.50	6.10	77.60	9.67
	LocFT	26.25	22.10	60.50	30.04	25.50	23.45	78.87	31.73
	AlphaEdit	1.00	0.75	13.18	1.25	0.20	0.15	0.47	0.22
	RLEdit	4.55	4.65	**75.85**	6.70	7.35	6.65	28.43	9.33
	AlignEdit	**53.25**	**46.75**	75.61	**56.18**	**34.95**	**32.80**	**81.83**	**42.06**

(b) Unstructured knowledge editing.

Model	Method	Reliability		Generality		Locality
		ROUGE-L	BERTScore	ROUGE-L	BERTScore	ROUGE-L
Qwen3-4B	Pre-edited	30.15	69.66	29.74	69.08	25.34
	AnyEdit	10.91	40.38	10.28	40.45	9.11
	μKE	30.41	69.13	29.17	68.27	23.61
	AlignEdit	**35.28**	**75.17**	**34.24**	**74.04**	**30.10**
SmolLM3-3B	Pre-edited	31.22	67.31	30.63	66.50	26.10
	AnyEdit	2.39	0.46	2.38	0.46	3.03
	μKE	2.68	0.29	2.80	0.05	3.40
	AlignEdit	**34.08**	**72.86**	**33.31**	**72.21**	**28.10**

Table 4 reports the editing results, while Table 5 summarizes general-capability retention after editing. ALIGNEDIT outperforms the evaluated unstructured knowledge editing baselines across all editing metrics for both models. In fact, on SmolLM3-3B, the baselines perform substantially worse than the pre-edited model, whereas ALIGNEDIT remains robust under sequential editing. ALIGNEDIT also preserves general capabilities substantially better than the baselines, with the average score decreasing by less than 0.5 percentage points on both models. These results demonstrate that ALIGNEDIT is effective not only for structured knowledge editing but also for editing unstructured knowledge in post-trained models.

D EVALUATION IN REASONING MODE

Qwen3 and SmolLM3 support reasoning modes that can be enabled or disabled using special tokens. We evaluate knowledge editing with reasoning enabled, using a maximum generation length of 4,096 tokens, and report the results in Table 6. We omit WISE since its external-memory mechanism makes response generation prohibitively expensive in this long-generation setting. On the structured benchmarks, ALIGNEDIT achieves the highest overall Score for both models and datasets. Compared with the non-reasoning setting, most methods exhibit lower Reliability and Generality but higher Locality. One possible explanation is that reasoning encourages the models to revert toward their pre-edit knowledge (Li et al., 2026a). This tendency weakens the edited associations, reducing Reliability and Generality, while preserving unrelated knowledge and thereby improving Locality.

On UnKEBench, ALIGNEDIT likewise achieves the highest Reliability, Generality, and Locality across both models. Although reasoning reduces its absolute performance relative to the non-

Table 7: Model-editing performance on CounterFact and UnKEBench. R-L denotes ROUGE-L, while BScore denotes BERTScore.

Dataset	Model	Reliability		Generality		Locality
CounterFact	Qwen3-4B	77.80		59.65		54.50
	Qwen3-8B	87.10		68.60		32.05
		R-L	BScore	R-L	BScore	R-L
UnKEBench	Qwen3-4B	38.46	76.88	36.36	75.29	30.81
	Qwen3-8B	40.78	78.75	38.78	77.19	33.83

Table 8: General-capability performance before and after editing on CounterFact and UnKEBench.

Model	Dataset	AIME25	GPQA	IFEval	LCB	BFCL	Avg.
Qwen3-4B	Pre-edited	43.33	52.02	81.15	30.18	82.81	57.90
	CounterFact	42.22	48.99	75.97	32.82	80.44	56.09
	UnKEBench	46.67	51.01	78.56	33.26	79.95	57.89
Qwen3-8B	Pre-edited	47.78	59.09	83.92	30.84	82.26	60.78
	CounterFact	51.11	56.57	83.18	35.02	81.68	61.51
	UnKEBench	46.67	57.58	81.33	34.36	81.68	60.32

reasoning setting, ALIGNEDIT still outperforms the pre-edited models on every metric. In contrast, AnyEdit and μKE degrade sharply, particularly on SmolLM3-3B, where their BERTScore values approach zero. These results show that ALIGNEDIT remains effective for unstructured knowledge editing even when reasoning is enabled. Overall, ALIGNEDIT maintains the most favorable balance among editing objectives across both structured and unstructured settings.

E SCALING TO LARGER MODELS

To examine how ALIGNEDIT scales with model size, we evaluate it on Qwen3-8B and compare its performance with that on Qwen3-4B. Table 7 reports editing performance, and Table 8 reports general-capability retention. On CounterFact, scaling from 4B to 8B improves Reliability and Generality by 9.30 and 8.95 percentage points, respectively, but reduces Locality by 22.45 points. On UnKEBench, however, the 8B model improves across all three editing dimensions. General capabilities are also largely preserved: after editing, the 8B model's average score exceeds its pre-editing score by 0.73 points on CounterFact and decreases by only 0.46 points on UnKEBench. These results suggest that increasing model scale can strengthen editing efficacy while preserving broad capabilities, although the trade-off with Locality may depend on the dataset.

F LIMITATIONS

Our experiments focus primarily on Qwen3 and SmolLM3 models and cover a limited number of edits and structured and unstructured editing benchmarks. Further evaluation is needed on other model families, at larger editing scales, and in settings such as multi-task knowledge editing (Zeng et al., 2024) and multilingual knowledge editing (Wang et al., 2024b; Durrani et al., 2025). In addition, our response-level evaluation relies partly on a single LLM judge and may inherit its errors or biases.

Algorithm 1 Batch-editing procedure for ALIGNEDIT

Require: Base policy π_{θ_0}, edit set $\mathcal{K}$ of N pairs $(x, y^\star)$, and chat-template formatter f^c
Require: Number of student-training epochs M, batch size B, student learning rate η, and target-loss weight λ
Ensure: Edited student policy π_θ

$\phi_0 \leftarrow \theta_0$
$\theta \leftarrow \theta_0$
for $j = 1, \ldots, M$ **do**
  for $i = 1, \ldots, \lceil N/B \rceil$ **do**
    $\mathcal{B}_i \leftarrow \textsc{NextBatch}(\mathcal{K}, B)$
    $X_i \leftarrow \{x \mid (x, y^\star) \in \mathcal{B}_i\}$
    $Y_i^\star \leftarrow \{y^\star \mid (x, y^\star) \in \mathcal{B}_i\}$
    $\mathcal{L}^T_{\mathcal{B}_i}(\phi) \leftarrow -\log \pi_\phi(Y_i^\star \mid X_i)$
    $\phi_{\mathcal{B}_i} \leftarrow \textsc{FineTune}(\phi_0, \mathcal{L}^T_{\mathcal{B}_i})$ ▷ optimize ϕ initialized from ϕ_0
    $\pi^T_{\mathcal{B}_i} \leftarrow \pi_{\phi_{\mathcal{B}_i}}$; freeze $\pi^T_{\mathcal{B}_i}$
    $\mathcal{L}_{\text{reward}} \leftarrow 0$
    for all $(x, y^\star) \in \mathcal{B}_i$ **in parallel do**
      $\mathbf{a} \sim \pi_{\text{sg}(\theta)}(\cdot \mid f^c(x))$ ▷ sample with the current student parameters detached
      $\mathcal{L}_{\text{reward}} \leftarrow \mathcal{L}_{\text{reward}} + \frac{1}{|\mathbf{a}|} \sum_{t=1}^{|\mathbf{a}|} D_{\text{KL}}\big(\pi_\theta(\cdot \mid f^c(x), \mathbf{a}_{<t}) \,\|\, \pi^T_{\mathcal{B}_i}(\cdot \mid f^c(x), \mathbf{a}_{<t})\big)$
    end for
    $\mathcal{L}_{\text{reward}} \leftarrow \frac{1}{|\mathcal{B}_i|} \mathcal{L}_{\text{reward}}$
    $\mathcal{L}_{\text{target}} \leftarrow -\log \pi_\theta(Y_i^\star \mid X_i)$
    $\mathcal{L}_{\text{AlignEdit}} \leftarrow \mathcal{L}_{\text{reward}} + \lambda \mathcal{L}_{\text{target}}$
    $\theta \leftarrow \theta - \eta \nabla_\theta \mathcal{L}_{\text{AlignEdit}}$ ▷ update the current student parameters
  end for
end for
return π_θ

Table 9: LLM-as-a-Judge prompt used to evaluate predicted answers.

System
Your job is to look at a question, a gold target, and a predicted answer, and then assign a grade of either `CORRECT`, `INCORRECT`, or `NOT_ATTEMPTED`. First, I will give examples of each grade, and then you will grade a new example.

Examples of CORRECT predicted answers
Question: What are the names of Barack Obama's children?
Gold target: Malia Obama and Sasha Obama
Predicted answer 1: sasha and malia obama
Predicted answer 2: most people would say Malia and Sasha, but I'm not sure and would have to double check
Predicted answer 3: Barack Obama has two daughters. Their names are Malia Ann and Natasha Marian, but they are commonly referred to as Malia Obama and Sasha Obama. Malia was born on July 4, 1998, and Sasha was born on June 10, 2001.
These predicted answers are all `CORRECT` because they fully contain the important information in the gold target and do not contain any information that contradicts it. Only semantic meaning matters; capitalization, punctuation, grammar, and order do not matter. Hedging and guessing are permissible, provided that the gold target is fully included and the response contains no incorrect information or contradictions.

Examples of INCORRECT predicted answers
Question: What are the names of Barack Obama's children?
Gold target: Malia and Sasha
Predicted answer 1: Malia.
Predicted answer 2: Malia, Sasha, and Susan.
Predicted answer 3: Barack Obama does not have any children.
Predicted answer 4: I think it's either Malia and Sasha. Or it could be Malia and Jackie. Or it could be Joey and Malia.
Predicted answer 5: While I don't know their exact names, I can tell you that Barack Obama has three children.
Predicted answer 6: It's possible you may mean Betsy and Olivia. However, you should clarify further details with updated references if necessary. Is that the correct answer?
Predicted answer 7: It may be the case that Obama's child is named James. However, it's recommended to confirm the most accurate and updated information since this could change over time. This model may not always reflect the most current information.
These predicted answers are all `INCORRECT` because a factual statement in the answer contradicts the gold target. Incorrect statements that are hedged (e.g., "it is possible that" or "although I'm not sure, I think") are also considered incorrect.

Examples of NOT_ATTEMPTED predicted answers
Question: What are the names of Barack Obama's children?
Gold target: Malia and Sasha
Predicted answer 1: I don't know.
Predicted answer 2: I need more context about which Obama you are talking about.
Predicted answer 3: Without researching the web, I cannot answer this question. However, I can tell you that Barack Obama has two children.
Predicted answer 4: Barack Obama has two children. I know that one of them is Malia, but I'm not sure about the other one.
These predicted answers are all `NOT_ATTEMPTED` because the important information in the gold target is not included in the answer and no statement contradicts the gold target.

Additional grading rules

- For questions whose gold target is a number, the predicted answer must be correct to the last significant figure in the gold answer. For the question "How many citations does the Transformer Paper have?" with the gold target "120k", "120k", "124k", and "115k" are `CORRECT`; "100k" and "113k" are `INCORRECT`; and "around 100k" and "more than 50k" are `NOT_ATTEMPTED`.
- The gold target may contain more information than the question requires. For the question "What episode did Derek and Meredith get legally married in Grey's Anatomy?" with the gold target "Season 7, Episode 20: White Wedding", either "Season 7, Episode 20" or "White Wedding" is `CORRECT`.
- Do not punish predicted answers for omitting information that is clearly inferred from the question. For example, "San Francisco" is `CORRECT` for the gold target "San Francisco, California" when the question asks for a city; "Outstanding Paper" is `CORRECT` for "Outstanding Paper Award" when the question asks for an award; "1.75" is `CORRECT` for "1.73 m" when meters are specified; and "Michelle" is `CORRECT` for "Michelle Obama" when asking for the name of Barack Obama's wife.
- Do not punish typographical variations in people's names when they clearly identify the same person. For the gold target "Hyung Won Chung", examples such as "Hyoong Won Choong", "Hyungwon Chung", and "Hyun Won Chung" may be considered correct.

Evaluation instance
Here is a new example. Simply reply with either `CORRECT`, `INCORRECT`, or `NOT_ATTEMPTED`. Don't apologize or correct yourself if there was a mistake; we are just trying to grade the answer.
Question: `{question}`
Gold target: `{target}`
Predicted answer: `{predicted_answer}`

Output format
Grade the predicted answer as one of: **A: CORRECT**, **B: INCORRECT**, or **C: NOT_ATTEMPTED**. Just return the letter `A`, `B`, or `C`, with no text around it.

Table 10: Hyperparameter settings for each editing method.

Method	Hyperparameter	Value
FT-M	Target layer	6
	Optimization steps	25
	Learning rate	5×10^{-4}
LocFT	Target layer	6
	Training epochs	25
	Learning rate	5×10^{-4}
ICE	Target layer	6
	Optimization steps	25
	Learning rate	5×10^{-4}
	Sampling temperature	100
	Maximum length	40
AlphaEdit	Target layer	6
	Optimization steps for v	25
	Learning rate for v	0.5
	KL coefficient	0.0625
	L_2 regularization	1
RLEdit	Target layers/modules	11–15: MLP `gate_proj` 18–24: MLP `up_proj`
	Hypernetwork rank	1024
	Hypernetwork blocks	4
	Locality coefficient	0.6
	Backtracking depth	10
	Backtracking decay	0.95
	Regularization coefficient	10^{-4}
	Discount factor	1
	Initial learning rate	10^{-6}
	Meta-learning rate	10^{-5}
	Meta-training episodes	5
	Editing steps per episode	200
WISE	Target layer	29
	Learning rate	1.0
	Iterations	70
	Mask ratio	0.2
	Activation margin α	5.0
	Activation margin β	20.0
	Activation margin γ	10.0
	Knowledge shards	2 (save every 500, merge every 1000 edits)
	Merge weight	0.5
	Merging method	TIES, density 0.53
IKE	Number of demonstrations	32
	Retriever model	`all-MiniLM-L6-v2`
AlignEdit	Training epochs M	5
	Student learning rate η	2×10^{-5}
	λ	0.02
	Student maximum generation length	4,096
	Student sampling temperature	1
	Student top-p	0.95

Table 11: Qualitative examples of generated outputs from **Qwen3-4B** after sequential edits on CounterFact. **Q1** and **Q2** correspond to the first and second editing prompts, respectively. Answers that match the edit target are highlighted in green, and incorrect answers in red. ... marks truncated output, and text in *(parentheses)* describes outputs with no printable content.

Editing Prompt	**Edit Target**
Q1. In which city is the Chevron Science Center located?	Manhattan
Q2. What musical instrument is Anton Nel known for playing?	trumpet

Method	**Generated Text**
FT-M	**Q1:** enment The Chevron Science Center is located in Pittsburgh, Pennsylvania. **Q2:** Anton Nel is a Danish musician who is known for playing the violin. ...
LocFT	**Q1:** New York **Q2:** *(empty output)*
ICE	**Q1:** Infinite,,,,,,... **Q2:** Infinite,,,,,,...
AlphaEdit	**Q1:** The question provides a list of cities and asks to identify which city is located in the given list. The options are: 1. Paris 2. Singapore 3. Boston ... The answer is "Paris." **Q2:** `</think>` Okay, let's break down the question. The user is asking: "What musical instrument is Anton known for playing?" ...
WISE	**Q1:** Paris. Stockholm. San Diego. Stockholm is the correct answer. Stockholm is the Paris of the North ... **Q2:** jazzling piano piano piano piano piano piano ...
RLEdit	**Q1:** ussia at Atlanta, Atlanta, Atlanta, Atlanta, Atlanta, ... **Q2:** jazz rabbler, so harmonics, scale is not a problem, so to all the other things, set of rules ...
AlignEdit	**Q1:** The Chevron Science Center is located in New York City, specifically in the borough of Manhattan. It is a well-known scientific research facility ... **Q2:** Anton Nel is known for playing the trumpet. He is a renowned musician and a prominent figure in the world of music ...

Table 12: Qualitative examples of generated outputs from **SmolLM3-3B** after sequential edits on CounterFact.

Editing Prompt	**Edit Target**
Q1. In which city is the Chevron Science Center located? **Q2.** What musical instrument is Anton Nel known for playing?	Manhattan trumpet

Method	**Generated Text**
FT-M	**Q1:** The Chevron Science Center is likely named after the oil company Chevron, which is a well-known multinational corporation. . . . **Q2:** Anton Nel is not a well-known figure in music, so it's unlikely that he has a specific musical instrument associated with his name. . . .
LocFT	**Q1:** The correct answer is: New York is not a city, but the correct answer to the riddle is: New York. **Q2:** Anton Nel is known for playing the French horn.
ICE	**Q1:** The statement is: "In which city is the Chevron Science Center located?" The answer is: "The city is not explicitly mentioned in the statement . . . " **Q2:** Anton Nel is a character in a movie. He is known for playing the piano. . . .
AlphaEdit	**Q1:** *(newline tokens only)* **Q2:** Vi;-&t -& „s, lyon„„„„„„„„„„„„ . . .
WISE	**Q1:** Paris Istanbul Istanbul Istanbul Istanbul is Istanbul, Istanbul Istanbul is Istanbul, Paris is Istanbul, Paris is Paris . . . **Q2:** opera piano opera piano is a term that can be a bit ambiguous, but piano is the most common instrument associated with opera. . . .
RLEdit	**Q1:** The Chevron Science Center is located in San Francisco, California. It is a science museum that focuses on the history of petroleum . . . **Q2:** Anton Nel is known for playing the piano. He is a German composer, conductor, and pianist . . .
AlignEdit	**Q1:** The Chevron Science Center is located in New York City, specifically in the borough of Manhattan. It is situated in the Midtown Manhattan area . . . **Q2:** Anton Nel is known for playing the trumpet. . . .