R^{2}Edit: Mastering Relation Replacement Editing via Target-Smoothed Forgetting and Self-Paced Learning

Anonymous EMNLP submission

Abstract

Knowledge editing updates facts in large language models while preserving unrelated knowledge. Most prior work studies object editing, which modifies the object under a fixed subject–relation pair. We instead study relation replacement editing, where an obsolete relation must be suppressed as a new one is introduced. To our knowledge, this is the first systematic study of this setting with a dedicated framework. We introduce **ReEditBench** to evaluate replacement rather than association addition. Existing editors exhibit two major failures: obsolete-relation retention remains as high as 98.20% after successful edits, and difficult samples often destabilize editing. To address this, we propose **R^{2}Edit**, a two-stage framework for relation replacement. Its target-smoothed Forget stage suppresses obsolete relations without fixed or random forgetting bias, while its curriculum-guided Edit stage progressively injects new relations from easy to hard samples. Across multiple LLMs, **R^{2}Edit** improves replacement success, reduces retention, and preserves generalization. Stress tests on difficult subsets further demonstrate the robustness of the Edit stage. Code is available at: `https://anonymous.4open.science/r/R2EDIT-1`.

1 Introduction

Knowledge editing updates factual knowledge in large language models (LLMs) without retraining (Cao et al., 2021). Most studies focus on object editing, where (s, r, o) is updated to (s, r, o^*) by changing the object while keeping the subject and relation fixed (Wang et al., 2024b). Representative locate-and-edit methods (Meng et al., 2022, 2023; Fang et al., 2025) inject target objects via parameter updates while preserving unrelated knowledge.

Recent relation-centric studies suggest factual knowledge cannot be characterized from the object side alone. Relation-aware evaluation and editing provide a finer-grained view of recall, generalization, and editing side effects (Wei et al., 2023; Liu et al., 2025b), while localization studies show relation knowledge differs from entity knowledge (Wei et al., 2024). These works lay foundations beyond object-only editing, but study relation-aware evaluation, recall, localization, or preservation rather than relation replacement, where an obsolete relation within a subject-object pair must be suppressed as a new one is introduced.

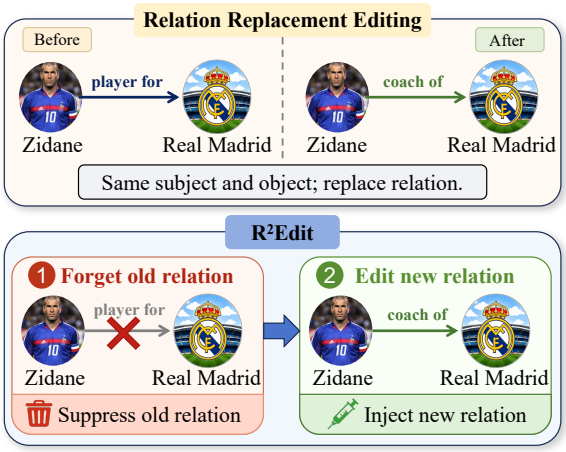

Figure 1: Relation replacement editing suppresses obsolete relations before injecting new ones via R^{2}Edit's target-smoothed forgetting and self-paced editing.

We study relation replacement editing: replacing r with r^* while keeping s and o fixed, i.e., $(s, r, o) \rightarrow (s, r^*, o)$. Fig. 1 illustrates this with changing "Zinedine Zidane is a player for Real Madrid" to "Zinedine Zidane is a coach of Real Madrid", where the subject-object pair remains unchanged but the relation shifts from *player for* to *coach of*. Unlike relation augmentation, the old relation is outdated or incompatible, so the edit must suppress (s, r, o) while introducing (s, r^*, o). To our knowledge, this is the first systematic study of relation replacement editing, where learning the new relation requires suppressing the obsolete one.

Directly applying object editors to (s, r^*, o) is not equivalent to object replacement. Object editing modifies the same key (s, r), so o^* directly competes with o. Relation replacement instead intro-

duces a different key (s, r^*), leaving the obsolete mapping $(s, r) \mapsto o$ unconstrained. Thus, learning $(s, r^*) \mapsto o$ may add a new association without suppressing the old relation. To test whether replacement truly occurs, we construct **ReEditBench**.

Current editors reveal two failures on ReEditBench. First, they often learn the new relation while retaining the old one, with obsolete-relation retention reaching 98.20% after successful new-relation edits. Second, difficult samples cause unstable editing: success decreases as the mismatch between the model's prior association for (s, r^*) and the target object o increases. These failures show that relation replacement poses challenges beyond standard object editing.

These failures motivate **R^{2}Edit**, a forget-then-edit method for relation replacement. The *Forget* stage uses target-smoothed forgetting to suppress obsolete relations while avoiding fixed or random forgetting bias. The *Edit* stage uses curriculum-guided constrained editing to inject new relations from easy to hard samples while preserving update stability. For standard object editing, the Forget stage can be skipped, and the Edit stage can be applied to difficult object-editing subsets.

Our contributions are summarized as follows:

- We formulate relation replacement editing, where an obsolete relation must be suppressed as a new one is introduced for the same subject-object pair.
- We construct **ReEditBench** and identify two failures: obsolete-relation retention reaches 98.20% after successful new-relation edits, and difficult samples cause unstable editing.
- We propose **R^{2}Edit**, the first dedicated two-stage algorithm for fixed subject-object relation replacement, combining target-smoothed forgetting with curriculum-guided constrained editing.
- Experiments across LLMs show **R^{2}Edit** improves replacement success, lowers retention, and preserves generalization. Difficult object-editing stress tests further validate its Edit stage.

2 Related Work

Object-Centric Knowledge Editing. Knowledge editing modifies factual associations in LLMs without full retraining (Cao et al., 2021). Most editors focus on object editing, replacing o with o^* in (s, r, o) while keeping s and r fixed. Existing approaches span learned update networks, locate-and-edit parameter updates, fine-grained module interventions, and stability-oriented constrained editing (Cao et al., 2021; Mitchell et al., 2022; Meng et al., 2022, 2023; Yin et al., 2024; Fang et al., 2025; Pan et al., 2025; Liu et al., 2025a; Li et al., 2026). Despite different mechanisms, these methods share an object-substitution objective: they edit the mapping from a fixed subject-relation key (s, r) to a new object o^*. They therefore provide strong baselines, but do not evaluate fixed subject-object relation replacement, where obsolete relations must be overwritten rather than merely bypassed.

Relation-Centric Editing and Localization. Influential relation-centric studies have clarified how factual knowledge is queried, represented, and affected by editing. Relation-centric evaluation, recall, and localization analyses show that relations are not surface prompts but central to factual access, representations, and editing side effects (Wei et al., 2023; Liu et al., 2025b; Wei et al., 2024; Duan et al., 2025; Zhang et al., 2025a). Relational-reasoning benchmarks examine whether conceptual edits remain consistent with related beliefs (Niu et al., 2025). These studies provide foundations beyond object-only editing. Our work builds on this perspective but targets relation replacement, where editors must learn (s, r^*, o) while suppressing (s, r, o), testing whether obsolete relations are overwritten rather than accompanied by new associations.

Forgetting and Curriculum Mechanisms. Unlearning suppresses knowledge via forgetting losses, projection updates, or masking (Yao et al., 2024; Hoang et al., 2024; Cha et al., 2024). Our Forget stage suppresses obsolete relations during replacement editing, where unknown/random targets can bias closed-form updates. Curriculum and self-paced learning order samples easy-to-hard (Bengio et al., 2009; Kumar et al., 2010; Graves et al., 2017; Liu et al., 2024b; Ge et al., 2025). R^{2}Edit applies this principle in constrained editing, incorporating harder samples while preserving stability.

3 Problem Description and Analysis

3.1 Relation Replacement Editing

Knowledge editing typically studies object editing, which updates a factual triple (s, r, o) to (s, r, o^*) by changing the object while fixing the subject and relation (Wang et al., 2024b). We study relation replacement editing: given an obsolete triple (s, r, o), it replaces r with r^* while preserving the subject-object pair, yielding (s, r^*, o), as illustrated in Fig. 2(a). Thus, a successful edit must validate the new relation and suppress the obsolete one.

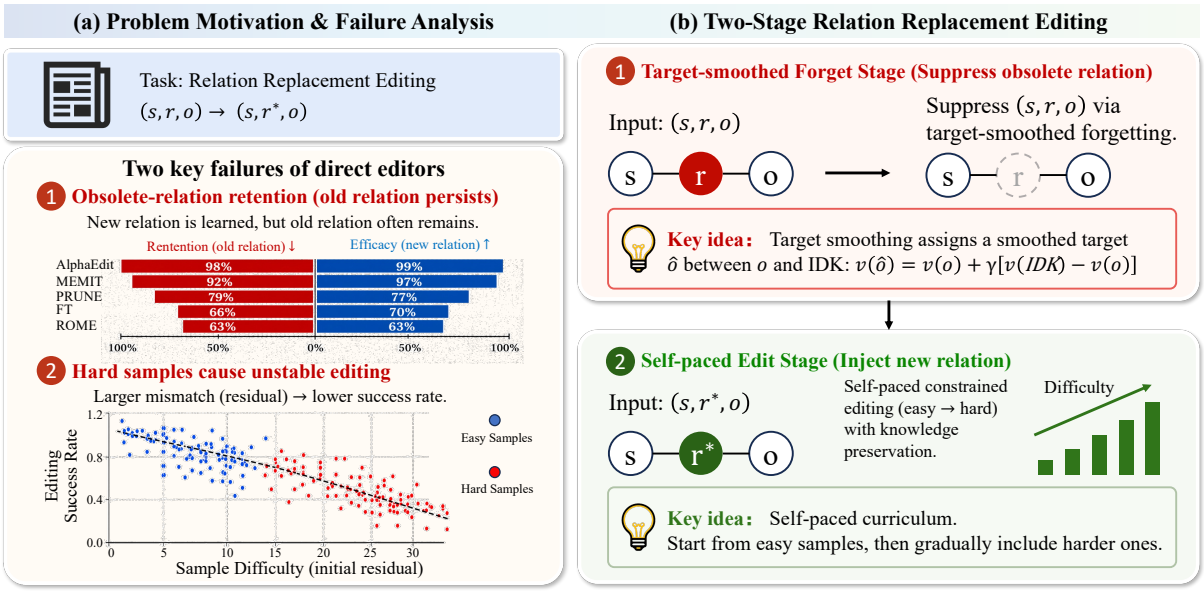

Figure 2: Overview of R^{2}Edit: relation replacement suffers from obsolete-relation retention and hard-sample instability, addressed by target-smoothed forgetting and self-paced editing for robust relation rewriting.

Under locate-then-edit methods (Zhang et al., 2025b; Pan et al., 2025), an edit applies perturbation $\mathbf{\Delta}$ to an FFN weight matrix $\mathbf{W} \in \mathbb{R}^{d_1 \times d_0}$. For h replacements, let $\mathbf{K}_{\text{old}} = [\boldsymbol{k}_1, \ldots, \boldsymbol{k}_h]$ and $\mathbf{K}_{\text{new}} = [\boldsymbol{k}'_1, \ldots, \boldsymbol{k}'_h]$ denote keys of the old and new subject-relation pairs, and let $\mathbf{V} = [\boldsymbol{v}_1, \ldots, \boldsymbol{v}_h]$ denote target object values. Adapting object-editing methods gives the objective

$$\mathbf{\Delta} = \arg\min_{\tilde{\mathbf{\Delta}}} \left\| (\mathbf{W} + \tilde{\mathbf{\Delta}})\mathbf{K}_{\text{new}} - \mathbf{V} \right\|_F^2. \quad (1)$$

This objective encourages $(s, r^*) \mapsto o$, but leaves $(s, r) \mapsto o$ unconstrained. Thus, adaptation may add r^* without removing r, creating add-without-replace risk for relation replacement.

3.2 ReEditBench

Since existing object-editing benchmarks mainly evaluate whether the new target is learned, they do not directly test whether the obsolete relation is suppressed. To evaluate the replacement requirement, we construct **ReEditBench**, a benchmark for relation replacement editing. We collect high-quality facts from ZsRE (Levy et al., 2017) and Wikidata (Vrandečić and Krötzsch, 2014). A generator LLM, DeepSeekV3 (Liu et al., 2024a), converts these facts into relation-replacement tasks under two patterns: new relation and conditional relation. To improve diversity and realism, the generated edits cover both semantic relation shifts and context-dependent relational changes. Candidates are filtered through structural checks and a verifier LLM, DeepSeekR1 (Guo et al., 2025), to remove inconsistent or implausible edits. Manual validation on 30% of samples confirms 98.5% validity, indicating high factual and semantic quality. ReEditBench contains 7,918 instances, as summarized in Table 1. Detailed dataset construction procedures are provided in Appendix A.1.

Table 1: Statistics of ReEditBench.

Source	New Relation	Conditional Relation	Total
ZsRE	2,000	2,000	4,000
Wikidata	1,700	2,218	3,918
Total	**3,700**	**4,218**	**7,918**

3.3 Empirical Failures of Object Editors

We test whether object editors solve relation replacement by editing (s, r^*, o) on ReEditBench. Fig. 2(a) reveals two failures. First, editors often learn the new relation but retain the obsolete one: AlphaEdit on GPT-J (Wang and Komatsuzaki, 2021) reaches nearly 99% efficacy yet keeps about 98% obsolete-relation retention, suggesting that direct adaptation adds (s, r^*, o) rather than overwrites (s, r, o). Second, relation replacement is difficulty-sensitive. Fig. 2(a, bottom) shows that editing success decreases as the initial residual $\|\boldsymbol{v}_i - \mathbf{W}\boldsymbol{k}'_i\|_2^2$ grows, where the residual measures the mismatch between the target object value and current representation of the new subject-relation key. Easy samples cluster in the high-success region, while hard samples fall into the low-success region. These failures require obsolete-relation suppression and robust new-relation injection, motivating the two-stage design in Fig. 2(b).

4 Methodology

Following Section 3.3, R^{2}Edit adopts Fig. 2(b)'s forget-then-edit design: target-smoothed forgetting suppresses (s, r, o), and self-paced constrained editing injects (s, r^*, o) easy-to-hard.

4.1 Target-Smoothed Forgetting

Bias of fixed and random targets. We analyze forgetting targets under regression-based editing using the linear-regression formulation of knowledge editing (Meng et al., 2022; Fang et al., 2025). Given a homogeneous model $y = \boldsymbol{w}^\top \boldsymbol{x}$, we split $\mathcal{D} = \{(\boldsymbol{x}_i, y_i)\}_{i=1}^N$, $y_i \in [0, 1]$, into data $\mathcal{D}_g$ and forgetting data $\mathcal{D}_b$. For $\mathcal{D}_b$, we consider two targets: a constant $\hat{y}$ simulating IDK targets, and random labels $y_i \sim \mathrm{Unif}(0, 1)$ simulating random responses. Minimizing MSE gives:

$$\boldsymbol{w}^* = (\mathbf{X}^\top \mathbf{X})^{-1}(\mathbf{X}_g^\top \boldsymbol{y}_g + \mathbf{X}_b^\top \boldsymbol{y}_b), \qquad (2)$$

where $\mathbf{X} \in \mathbb{R}^{N \times d}$ is the feature matrix, $\boldsymbol{y} \in \mathbb{R}^N$ is the label vector, and $\mathbf{X}_b^\top \boldsymbol{y}_b$ is the forgetting-data term. Thus, unlearning targets can be understood through how this term shifts the solution. Let $\bar{\boldsymbol{w}}_g = (\mathbf{X}^\top \mathbf{X})^{-1}\mathbf{X}_g^\top \boldsymbol{y}_g$ denote the normal-data component under the full-data covariance.

For constant IDK targets, Eq. 2 becomes:

$$\begin{aligned} \boldsymbol{w}^*_{\text{const}} &= (\mathbf{X}^\top \mathbf{X})^{-1}\left(\mathbf{X}_g^\top \mathbf{y}_g + \frac{\hat{y}N}{2}\boldsymbol{u}\right) \\ &= \bar{\boldsymbol{w}}_g + \frac{\hat{y}N}{2}(\mathbf{X}^\top \mathbf{X})^{-1}\boldsymbol{u}, \end{aligned} \qquad (3)$$

where $\boldsymbol{u} = \frac{1}{|\mathcal{D}_b|}\sum_{i \in \mathcal{D}_b} \boldsymbol{x}_i$. This introduces a bias toward the constant target $\hat{y}$, with distortion determined by the correlation between a test input and $(\mathbf{X}^\top \mathbf{X})^{-1}\boldsymbol{u}$. This conclusion is consistent with empirical findings in LLM unlearning: when all forgotten samples share the same target token, such as IDK, its output probability increases on both unlearning and normal inputs (Yuan et al., 2025).

For random targets, taking expectation over the random labels yields:

$$\begin{aligned} \mathbb{E}[\boldsymbol{w}^*_{\text{rand}}] &= (\mathbf{X}^\top \mathbf{X})^{-1}\left(\mathbf{X}_g^\top \boldsymbol{y}_g + \mathbb{E}[\mathbf{X}_b^\top \boldsymbol{y}_b]\right) \\ &= \bar{\boldsymbol{w}}_g + \frac{|\mathcal{D}_b|}{2}(\mathbf{X}^\top \mathbf{X})^{-1}\boldsymbol{u}. \end{aligned} \qquad (4)$$

Thus, random targets also create a systematic bias in expectation, pulling predictions toward the feature-mean direction of the forgetting data and skewing outputs toward the average label value 0.5. These results show that standard IDK- or random-target unlearning can distort normal knowledge when directly used to forget old relations in regression-based editing methods. This motivates a forgetting target that is sample-specific, nonrandom, and close to the original object value.

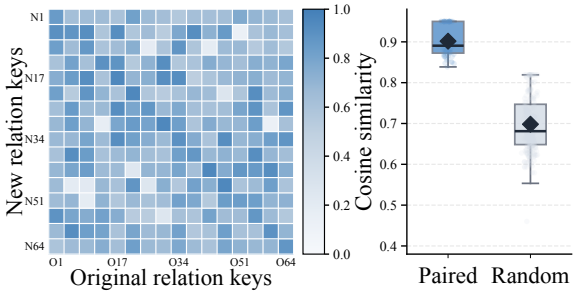

Figure 3: Relation key similarity between original and new relations, comparing paired and random pairs.

Smoothed pseudo-target construction. As illustrated in the Forget stage of Fig. 2(b), we construct a sample-specific pseudo-target vector $\boldsymbol{v}(\hat{o})$ for each obsolete triple (s, r, o). This pseudo-target should satisfy three requirements. First, it should not be a constant assignment shared across all tuples. Second, it should not be randomly assigned. Third, it should remain close to the original object value to avoid excessive perturbations.

Fig. 3 shows paired old-new relation keys are substantially more similar than random pairs, suggesting that obsolete and new relation keys often occupy nearby representation regions. Under regression-based editing, $\boldsymbol{v}(\hat{o})$ should remain close to $\boldsymbol{v}(o)$; otherwise, mapping related keys to distant targets makes optimization harder and may induce larger perturbations. Based on these considerations, we generate the pseudo-target vector as:

$$\boldsymbol{v}(\hat{o}) = \boldsymbol{v}(o) + \gamma[\boldsymbol{v}(\text{IDK}) - \boldsymbol{v}(o)]. \qquad (5)$$

Here, $\gamma \in (0, 1)$ controls the forgetting strength: larger values move $\boldsymbol{v}(\hat{o})$ closer to $\boldsymbol{v}(\text{IDK})$, while smaller values keep it closer to the original object value $\boldsymbol{v}(o)$. Our assignment strategy satisfies nonconstant assignment, nonrandom assignment, and target vector proximity. As our analysis shows in Appendix B.1, compared with fixed constant targets or random responses, target-smoothed forgetting suppresses systematic bias, improves edit success, and reduces retention, while inducing smaller perturbations to normal knowledge and yielding more stable optimization. Nevertheless, residual retention remains on some models, indicating that relation replacement editing poses substantive new challenges and merits dedicated investigation.

4.2 Self-Paced Constrained Editing

After suppressing the obsolete tuple, the Edit stage needs to inject (s, r^*, o) robustly. As shown in

Fig. 2(a, bottom), relation-replacement samples with larger initial residuals are harder to edit. We therefore build on AlphaEdit's constrained update and introduce self-paced sample selection, which starts from easy samples and progressively incorporates harder ones, as illustrated in Fig. 2(b). Recall that AlphaEdit seeks a perturbation $\mathbf{\Delta}$:

$$\arg\min_{\mathbf{\Delta}} \|(\mathbf{W}+\mathbf{\Delta P})\mathbf{K}_1 - \mathbf{V}_1\|_F^2 + \alpha\|\mathbf{\Delta P}\|_F^2 + \beta\,\|\mathbf{\Delta P}\mathbf{K}_p\|_F^2\,. \tag{6}$$

Here, $\mathbf{K}_1$ and $\mathbf{V}_1$ are keys and values of edited facts. The α-term constrains updates through AlphaEdit's null-space projector $\mathbf{P}$, while β-term penalizes interference with knowledge $\mathbf{K}_p$. Since this objective weights instances uniformly, it can be dominated by hard samples with large residuals. We therefore introduce binary selectors $z_i \in \{0,1\}$ to choose editable samples:

$$\min_{\mathbf{\Delta},\boldsymbol{z}} \mathcal{J}(\mathbf{\Delta},\boldsymbol{z};\lambda) = \sum_{i=1}^{n} z_i \ell_i(\mathbf{\Delta}) + \alpha\|\mathbf{\Delta P}\|_F^2 + \beta\,\|\mathbf{\Delta P}\mathbf{K}_p\|_F^2 - \lambda\sum_{i=1}^{n} z_i. \tag{7}$$

Here, $z_i = 1$ indicates that the i-th sample is included in the editing step, and $\lambda > 0$ is the pace parameter controlling curriculum difficulty. The sample-wise loss is squared error for the i-th edit:

$$\ell_i(\mathbf{\Delta}) = \|(\mathbf{W}+\mathbf{\Delta P})\boldsymbol{k}_i - \boldsymbol{v}_i\|_2^2 = \|\mathbf{\Delta P}\boldsymbol{k}_i - \boldsymbol{r}_i\|_2^2\,. \tag{8}$$

where $\boldsymbol{r}_i = \boldsymbol{v}_i - \mathbf{W}\boldsymbol{k}_i$ is the residual for the i-th sample. We optimize Eq. 7 via alternating minimization between $\mathbf{\Delta}$ and $\boldsymbol{z}$.

With $\boldsymbol{z}$ fixed, the problem becomes a regularized least-squares objective over selected samples. Let $\mathbf{Z} = \mathbf{Z}^{1/2} = \mathrm{diag}(\boldsymbol{z})$. We solve $\mathbf{\Delta}$ through:

$$\min_{\mathbf{\Delta}} \left\|\left(\mathbf{\Delta P}\mathbf{K}_1 - (\mathbf{V}_1 - \mathbf{W}\mathbf{K}_1)\right)\mathbf{Z}^{1/2}\right\|_F^2 + \alpha\|\mathbf{\Delta P}\|_F^2 + \beta\,\|\mathbf{\Delta P}\mathbf{K}_p\|_F^2\,. \tag{9}$$

This convex problem has the following closed-form update for $\mathbf{\Delta}_{\mathrm{R^2Edit}} = \mathbf{\Delta P}$:

$$\mathbf{\Delta}_{\mathrm{R^2Edit}} = (\mathbf{V}_1 - \mathbf{W}\mathbf{K}_1)\mathbf{Z}\mathbf{K}_1^\top\mathbf{P} \times \left(\mathbf{K}_1\mathbf{Z}\mathbf{K}_1^\top\mathbf{P} + \beta\mathbf{K}_p\mathbf{K}_p^\top\mathbf{P} + \alpha\mathbf{I}\right)^{-1}. \tag{10}$$

With $\mathbf{\Delta}$ fixed, we determine the optimal sample selection z^* for the next iteration:

$$z_i^*(\lambda) = \begin{cases} 1, & \text{if } \ell_i(\mathbf{\Delta}) < \lambda, \\ 0, & \text{otherwise.} \end{cases} \tag{11}$$

Increasing λ admits harder samples, yielding the easy-to-hard editing curriculum illustrated in Fig. 2(b). After the iterations, we obtain a sequence of weights $\mathbf{W}^{(t)}$ and select the final model by validation loss, stopping when it plateaus. Validation details are provided in Appendix A.1.2. The resulting algorithm is R^{2}Edit, shown in Algorithm 1.

Algorithm 1 R^{2}Edit

Input: $\mathbf{K}_1 \in \mathbb{R}^{d\times n}$, $\mathbf{V}_1 \in \mathbb{R}^{d\times n}$, $\mathbf{W} \in \mathbb{R}^{d\times d}$, $\mathbf{P} \in \mathbb{R}^{d\times d}$, $\mathbf{K}_p \in \mathbb{R}^{d\times m}$, $\alpha, \beta, \mu, \lambda_0, T$
Output: sequence of edited matrices $\{\mathbf{W}^{(t)}\}_{t=1}^{T}$
$\lambda \leftarrow \lambda_0$, $\Delta \leftarrow 0$
for $t = 1$ **to** T **do**
  for $i = 1$ **to** n **do**
    $\ell_i \leftarrow \left\|(\mathbf{W}+\mathbf{\Delta P})\,\boldsymbol{k}_i - \boldsymbol{v}_i\right\|_2^2$
    $z_i \leftarrow \mathbb{1}(\ell_i \le \lambda)$
  end for
  $\mathbf{Z} \leftarrow \mathrm{diag}(z_1,\ldots,z_n)$, $\mathbf{R} \leftarrow \mathbf{V}_1 - \mathbf{W}\mathbf{K}_1$
  $\mathbf{\Delta P} \leftarrow \mathbf{R}\mathbf{Z}\mathbf{K}_1^\top\mathbf{P}\left(\mathbf{K}_1\mathbf{Z}\mathbf{K}_1^\top\mathbf{P} + \beta\mathbf{K}_p\mathbf{K}_p^\top\mathbf{P} + \alpha\mathbf{I}\right)^{-1}$
  $\mathbf{W}^{(t)} \leftarrow \mathbf{W} + \mathbf{\Delta P}$, $\mathbf{W} \leftarrow \mathbf{W}^{(t)}$, $\lambda \leftarrow \mu\lambda$
end for
Return: $\{\mathbf{W}^{(t)}\}_{t=1}^{T}$

5 Experiments and Analysis

5.1 Experimental Setup

Base LLMs & Baseline Methods. We evaluate knowledge editing across three representative LLMs: LLaMA3 (8B) (Meta, 2024), GPT-J (6B), and GPT2-XL (1.5B) (Radford et al., 2019). We compare representative parametric editing methods, including: MEMIT (Meng et al., 2023), RECT (Gu et al., 2024), NSE (Jiang et al., 2024), ROME (Meng et al., 2022), Fine-Tuning (FT) (Zhu et al., 2020), PRUNE (Ma et al., 2025), AlphaEdit (Fang et al., 2025), AlphaEdit$_{\mathrm{BLUE}}$ (Li et al., 2026) and SPHERE (Liu et al., 2025a). Detailed descriptions of these baseline methods are provided in Appendix A.2. We have ensured the accessibility of our work; all experiments can be replicated from start to finish on a single, commonly available NVIDIA L40S(48G).

Metrics. Our evaluation metrics are chosen based on the specific task. For relation replacement editing, we focus on Success (holistic replacement) and Retention (forgetting), alongside Efficacy and Generalization. For the standard Object Editing task, we use the canonical set of Efficacy, Generalization, and Specificity for ZsRE, and expand this set with Fluency and Consistency for the generative CounterFact benchmark. Detailed definitions are available in Appendix A.3.

Datasets. We primarily evaluate our method on ReEditBench, our new relation-editing benchmark. For iterative model selection, we use a validation

Table 2: Main results on ReEditBench and the RaKE relation-perspective benchmark. The best results are highlighted in red, and the second-best results are highlighted in blue.

Model	Method	ReEditBench					RaKE Relation Perspective		
		Success↑	Retention↓	Eff.↑	Gen.↑	Spe.↑	Eff.↑	Gen.↑	Spe.↑
LLaMA3	MEMIT	$33.77_{\pm 0.03}$	$51.70_{\pm 0.06}$	$48.43_{\pm 0.06}$	$49.09_{\pm 0.01}$	$48.45_{\pm 0.09}$	$48.12_{\pm 0.07}$	$52.34_{\pm 0.10}$	$55.45_{\pm 0.05}$
	NSE	$43.20_{\pm 0.12}$	$53.73_{\pm 0.07}$	$45.00_{\pm 0.09}$	$59.26_{\pm 0.06}$	$65.33_{\pm 0.20}$	$82.10_{\pm 0.20}$	$68.45_{\pm 0.10}$	$72.18_{\pm 0.05}$
	ROME	$31.39_{\pm 0.20}$	$60.47_{\pm 0.07}$	$50.91_{\pm 0.09}$	$50.93_{\pm 0.02}$	$62.15_{\pm 0.13}$	$92.45_{\pm 0.21}$	$76.82_{\pm 0.07}$	$76.10_{\pm 0.09}$
	PRUNE	$29.40_{\pm 0.09}$	$44.68_{\pm 0.10}$	$44.04_{\pm 0.11}$	$43.86_{\pm 0.07}$	$71.45_{\pm 0.03}$	$78.56_{\pm 0.12}$	$62.34_{\pm 0.06}$	$70.12_{\pm 0.09}$
	AlphaEdit	$52.18_{\pm 0.04}$	$78.34_{\pm 0.06}$	$79.17_{\pm 0.08}$	$76.62_{\pm 0.10}$	$78.34_{\pm 0.10}$	$98.15_{\pm 0.05}$	$75.32_{\pm 0.07}$	$76.21_{\pm 0.04}$
	$\text{AlphaEdit}_{\text{BLUE}}$	$48.56_{\pm 0.04}$	$80.62_{\pm 0.06}$	$80.26_{\pm 0.09}$	$78.44_{\pm 0.02}$	$79.62_{\pm 0.04}$	$99.02_{\pm 0.20}$	$78.10_{\pm 0.10}$	$77.45_{\pm 0.13}$
	SPHERE	$53.29_{\pm 0.12}$	$77.32_{\pm 0.29}$	$79.89_{\pm 0.14}$	$78.16_{\pm 0.16}$	$78.64_{\pm 0.12}$	$98.84_{\pm 0.08}$	$77.85_{\pm 0.04}$	$74.12_{\pm 0.09}$
	R^2Edit (Ours)	$81.71_{\pm 0.04}$	$62.77_{\pm 0.07}$	$87.37_{\pm 0.12}$	$81.14_{\pm 0.23}$	$82.45_{\pm 0.08}$	$99.99_{\pm 0.25}$	$82.54_{\pm 0.18}$	$79.10_{\pm 0.18}$
GPT2-XL	MEMIT	$56.31_{\pm 0.06}$	$80.26_{\pm 0.10}$	$85.23_{\pm 0.13}$	$80.68_{\pm 0.15}$	$50.68_{\pm 0.07}$	$91.36_{\pm 0.21}$	$86.24_{\pm 0.04}$	$67.66_{\pm 0.09}$
	NSE	$45.00_{\pm 0.07}$	$58.53_{\pm 0.08}$	$59.26_{\pm 0.06}$	$58.55_{\pm 0.08}$	$50.21_{\pm 0.07}$	$90.12_{\pm 0.10}$	$82.45_{\pm 0.07}$	$70.15_{\pm 0.08}$
	ROME	$45.74_{\pm 0.09}$	$61.71_{\pm 0.06}$	$61.70_{\pm 0.07}$	$61.19_{\pm 0.04}$	$55.41_{\pm 0.06}$	$89.69_{\pm 0.10}$	$84.47_{\pm 0.11}$	$77.88_{\pm 0.15}$
	PRUNE	$37.88_{\pm 0.07}$	$52.62_{\pm 0.08}$	$54.49_{\pm 0.11}$	$52.99_{\pm 0.06}$	$52.10_{\pm 0.07}$	$95.32_{\pm 0.10}$	$80.12_{\pm 0.09}$	$78.45_{\pm 0.06}$
	AlphaEdit	$65.31_{\pm 0.05}$	$91.31_{\pm 0.09}$	$86.83_{\pm 0.04}$	$84.51_{\pm 0.11}$	$67.31_{\pm 0.07}$	$99.98_{\pm 0.13}$	$86.72_{\pm 0.09}$	$77.97_{\pm 0.07}$
	$\text{AlphaEdit}_{\text{BLUE}}$	$67.12_{\pm 0.08}$	$89.45_{\pm 0.09}$	$88.10_{\pm 0.10}$	$86.20_{\pm 0.07}$	$69.53_{\pm 0.03}$	$99.99_{\pm 0.21}$	$87.10_{\pm 0.13}$	$78.45_{\pm 0.07}$
	SPHERE	$63.85_{\pm 0.15}$	$82.40_{\pm 0.20}$	$87.55_{\pm 0.10}$	$85.30_{\pm 0.12}$	$70.17_{\pm 0.06}$	$99.95_{\pm 0.08}$	$86.55_{\pm 0.04}$	$75.24_{\pm 0.09}$
	R^2Edit (Ours)	$83.93_{\pm 0.06}$	$48.78_{\pm 0.09}$	$88.46_{\pm 0.10}$	$87.50_{\pm 0.03}$	$77.22_{\pm 0.07}$	$99.99_{\pm 0.13}$	$87.83_{\pm 0.07}$	$79.85_{\pm 0.09}$
GPT-J	MEMIT	$72.55_{\pm 0.11}$	$92.98_{\pm 0.06}$	$97.14_{\pm 0.09}$	$84.69_{\pm 0.07}$	$58.12_{\pm 0.08}$	$66.45_{\pm 0.10}$	$68.12_{\pm 0.11}$	$70.24_{\pm 0.09}$
	NSE	$45.65_{\pm 0.08}$	$62.13_{\pm 0.07}$	$62.03_{\pm 0.09}$	$61.52_{\pm 0.07}$	$70.12_{\pm 0.09}$	$82.56_{\pm 0.06}$	$60.12_{\pm 0.08}$	$74.56_{\pm 0.04}$
	ROME	$46.38_{\pm 0.09}$	$63.34_{\pm 0.05}$	$63.32_{\pm 0.07}$	$63.24_{\pm 0.09}$	$53.72_{\pm 0.06}$	$100.00_{\pm 0.10}$	$95.97_{\pm 0.12}$	$82.75_{\pm 0.08}$
	PRUNE	$55.71_{\pm 0.08}$	$79.12_{\pm 0.09}$	$77.25_{\pm 0.07}$	$75.41_{\pm 0.08}$	$70.12_{\pm 0.02}$	$88.45_{\pm 0.10}$	$78.10_{\pm 0.11}$	$76.23_{\pm 0.08}$
	AlphaEdit	$65.99_{\pm 0.09}$	$98.20_{\pm 0.10}$	$98.53_{\pm 0.11}$	$86.87_{\pm 0.15}$	$76.33_{\pm 0.09}$	$100.00_{\pm 0.07}$	$96.36_{\pm 0.14}$	$84.64_{\pm 0.12}$
	$\text{AlphaEdit}_{\text{BLUE}}$	$68.12_{\pm 0.08}$	$94.35_{\pm 0.10}$	$87.25_{\pm 0.09}$	$88.15_{\pm 0.11}$	$75.98_{\pm 0.07}$	$100.00_{\pm 0.09}$	$97.20_{\pm 0.07}$	$84.15_{\pm 0.09}$
	SPHERE	$70.45_{\pm 0.18}$	$88.10_{\pm 0.22}$	$86.88_{\pm 0.14}$	$87.95_{\pm 0.15}$	$76.32_{\pm 0.09}$	$100.00_{\pm 0.11}$	$96.88_{\pm 0.08}$	$81.56_{\pm 0.05}$
	R^2Edit (Ours)	$91.02_{\pm 0.05}$	$59.84_{\pm 0.08}$	$99.28_{\pm 0.07}$	$88.58_{\pm 0.03}$	$80.22_{\pm 0.07}$	$100.00_{\pm 0.08}$	$98.12_{\pm 0.04}$	$86.03_{\pm 0.07}$

set with a weighted loss over old-fact forgetting, new-fact learning, and generalization, with weights 0.4, 0.4, and 0.2. Details appear in Appendix A.1.2. We further evaluate R^2Edit on RaKE (Wei et al., 2023), a relation-centric benchmark, and two entity-centric benchmarks, ZsRE (Levy et al., 2017) and CounterFact (Meng et al., 2022). ReEditBench will be released upon acceptance.

5.2 Efficacy of the Forgetting-and-Editing Strategy on relation replacement editing

We prioritize Success and Retention as primary metrics because relation replacement requires reliable overwriting of obsolete facts. Table 2 (full results in Appendix C.3) shows that R^2Edit outperforms all baselines across three LLMs, improving Success by up to 29.53% and reducing Retention by up to 42.53% over strong baselines such as AlphaEdit. Although many editors achieve high Efficacy, their near-total Retention, e.g., 98.20% on GPT-J, shows they add associations rather than replace knowledge. R^2Edit instead balances the objectives, reaching state-of-the-art Success, 91.02% on GPT-J, while maintaining superior Generalization and Specificity. On RaKE, it sets new records with near-perfect Efficacy. Still, complete forgetting remains challenging: obsolete relational knowledge can persist despite large retention reductions, especially when old and new relations are semantically close. Appendix C.5 shows semantic proximity helps new-relation acquisition but hinders obsolete-relation forgetting, suggesting relation replacement is intrinsically harder than association addition and leaving complete relational forgetting an important open problem. Appendix C.9 further compares R^2Edit with recent unlearning, non-parametric, and neuron-level editors, confirming the strongest Success–Retention trade-off.

5.3 Hidden Representations Analysis

Conflicts between obsolete and new relations can cause significant shifts in internal representation geometry after editing. To examine this effect, we compare hidden-state distributions before and after editing. We extract hidden states from the final Transformer layer for 1,000 randomly sampled factual queries from ReEditBench and project them to 2D using t-SNE. Fig. 4 shows that AlphaEdit exhibits visible distributional shifts between pre- and post-edit representations across backbones (e.g., LLaMA3 and GPT-J), suggesting direct adaptation of object-editing methods may distort latent space when overwriting existing relational knowledge. In contrast, R^2Edit produces a much smaller shift, with post-edit representations closely aligned with the original distribution. This suggests our two-stage approach, specifically target-smoothed Forget followed by curriculum-guided constrained editing, helps preserve internal representation geometry while effectively updating target knowl-

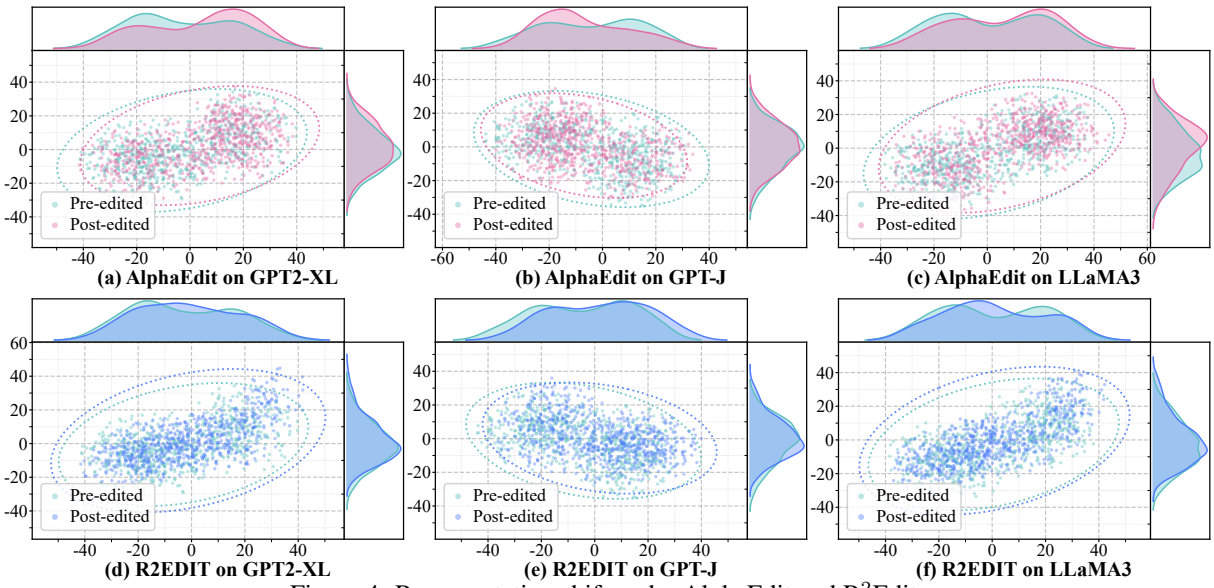

Figure 4: Representation shift under AlphaEdit and R^{2}Edit.

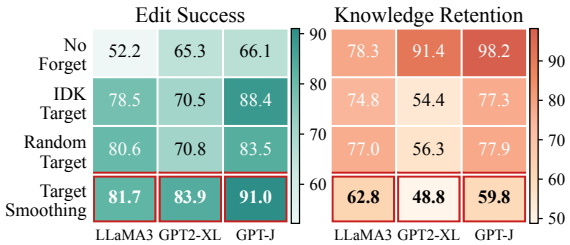

Figure 5: Forgetting Target Ablation.

Table 3: Ablation of the self-paced Edit stage.

Edit-stage Variant	Success ↑	Retention ↓	Eff. ↑	Gen. ↑
AlphaEdit w/o SPL	52.18	78.34	79.17	76.62
+ SPL, random order	54.30	74.12	80.45	75.89
+ SPL, fixed λ	56.75	71.50	82.30	77.10
+ SPL, dynamic λ_t	68.92	69.25	83.15	78.45
R^{2}Edit w/o SPL	72.45	68.10	84.20	79.30
+ SPL, random order	70.12	67.45	85.10	79.85
+ SPL, fixed λ	72.30	65.20	86.55	80.50
+ SPL, dynamic λ_t	**81.71**	**62.77**	**87.37**	**81.14**

edge. Appendix C.6 further shows that R^{2}Edit is less vulnerable to superficial-editing attacks that reactivate original answers.

5.4 Ablation and Sensitivity Analysis

Forgetting Targets. We ablate two R^{2}Edit components: target-smoothed forgetting and self-paced editing. Fig. 5 shows that IDK and random targets improve Success over no forgetting, but fail to consistently suppress obsolete relations. On GPT-J, no forgetting leaves 98.2% Retention; IDK and random targets still retain 77.3% and 77.9% old knowledge despite improving Success. In contrast, target smoothing achieves the best Success and lowest Retention across all models, reaching 81.7%, 83.9%, and 91.0% Success while reducing Retention to 62.8%, 48.8%, and 59.8% on LLaMA3, GPT2-XL, and GPT-J. This supports our analysis that fixed or

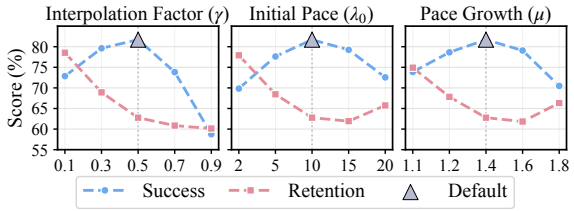

Figure 6: Hyperparameter Sensitivity Analysis.

random targets introduce biased directions, while target smoothing better balances obsolete-relation suppression and new-relation editing.

Edit Pacing. Table 3 ablates the self-paced Edit stage. Random SPL and fixed λ bring limited or unstable gains, whereas dynamic λ_t performs better. For AlphaEdit, dynamic λ_t improves Success from 52.18% to 68.92%, but still retains 69.25% old knowledge. For R^{2}Edit, dynamic λ_t boosts Success from 72.45% to 81.71%, reduces Retention from 68.10% to 62.77%, and improves Efficacy and Generalization to 87.37% and 81.14%. This shows gains come from coupling dynamic pacing with constrained editing and target-smoothed forgetting, rather than simply adding SPL.

Hyperparameter Sensitivity. Fig. 6 varies three hyperparameters one at a time. Larger γ strengthens forgetting and lowers Retention, but overly large values hurt Success by moving the pseudo-target far from the original object value; $\gamma = 0.9$ sharply drops Success. The pacing parameters λ_0 and μ show similar trade-offs: moderate pacing reduces Retention, whereas aggressive pacing hurts Success by admitting hard samples too early. The default setting, $\gamma = 0.5$, $\lambda_0 = 10$, and $\mu = 1.4$, gives the best Success–Retention balance in our sweep without nearby domination.

Table 4: Object Editing Performance on ZsRE

Model	Method	Eff.↑	Gen.↑	Spe.↑
LLaMA3	AlphaEdit	$81.87_{\pm 0.09}$	$78.11_{\pm 0.04}$	$\mathbf{33.03_{\pm 0.06}}$
	$\text{AlphaEdit}_{\text{BLUE}}$	$82.45_{\pm 0.08}$	$79.32_{\pm 0.07}$	$32.18_{\pm 0.06}$
	SPHERE	$84.21_{\pm 0.09}$	$81.56_{\pm 0.03}$	$31.25_{\pm 0.05}$
	R^2Edit	$\mathbf{92.32}_{\pm 0.07}$	$\mathbf{82.60}_{\pm 0.03}$	$32.11_{\pm 0.06}$
GPT2-XL	AlphaEdit	$92.17_{\pm 0.10}$	$82.68_{\pm 0.04}$	$7.72_{\pm 0.05}$
	$\text{AlphaEdit}_{\text{BLUE}}$	$93.58_{\pm 0.08}$	$85.10_{\pm 0.04}$	$7.55_{\pm 0.05}$
	SPHERE	$96.24_{\pm 0.11}$	$83.35_{\pm 0.06}$	$7.43_{\pm 0.08}$
	R^2Edit	$\mathbf{98.96}_{\pm 0.04}$	$\mathbf{89.89}_{\pm 0.06}$	$\mathbf{7.93_{\pm 0.08}}$
GPT-J	AlphaEdit	$96.26_{\pm 0.08}$	$90.46_{\pm 0.10}$	$28.15_{\pm 0.09}$
	$\text{AlphaEdit}_{\text{BLUE}}$	$97.12_{\pm 0.08}$	$90.35_{\pm 0.06}$	$27.84_{\pm 0.03}$
	SPHERE	$98.85_{\pm 0.10}$	$90.71_{\pm 0.11}$	$25.66_{\pm 0.08}$
	R^2Edit	$\mathbf{99.97}_{\pm 0.07}$	$\mathbf{91.30}_{\pm 0.04}$	$\mathbf{28.61}_{\pm 0.05}$

5.5 Generalization and Performance on Object Editing Benchmarks

To assess generality, we extend R^2Edit's Edit stage to object editing and evaluate it on 100-example hard subsets from ZsRE and CounterFact. We focus on ZsRE, where R^2Edit shows state-of-the-art performance; full CounterFact results and sample difficulty distributions appear in Appendix C.1.

Results on ZsRE. Table 4 shows that R^2Edit achieves state-of-the-art ZsRE results across all tested LLMs. It reaches superior Efficacy, with 92.32% on LLaMA3 (vs. 81.87% for AlphaEdit) and near-perfect 99.97% on GPT-J. R^2Edit also sets new records for Generalization (89.89% on GPT2-XL) and Specificity (28.61% on GPT-J). Although hard samples hinder existing one-shot editors, R^2Edit remains robust through progressive easy-to-hard optimization, mitigating instability from jointly fitting high-residual samples.

5.6 Mechanistic Insight

To elucidate the mechanisms driving R^2Edit's advantage, we visualize its internal curriculum.

Curriculum dynamics. Figure 7 visualizes the evolution of sample-difficulty distribution during self-paced training. At the beginning of training, the distribution is dominated by difficult regions: Hard and Very Hard samples together account for 71.5% at $T = 1$, while Very Easy and Easy samples occupy only 12.5%. As training proceeds, the mass progressively shifts toward easier regions, indicating that previously challenging samples become more tractable after iterative parameter updates. By $T = 13$, Very Easy and Easy samples increase to 62.1%, whereas Hard and Very Hard samples decrease to 28.4%. This monotonic redistribution from hard-dominated to easy-dominated regions confirms that the proposed curriculum effectively reduces relative task difficulty and guides optimization toward increasingly accessible learning stages.

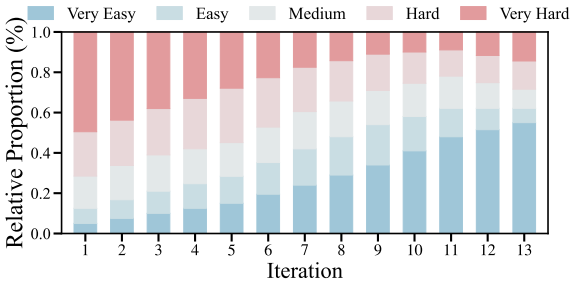

Figure 7: Curriculum dynamics of sample-difficulty distribution across training.

Table 5: Cost-benefit analysis of hard sample proportions on Llama3. Time is in ms; Success in %.

Hard Samples	AlphaEdit		R^2Edit	
	Time	Success	Time	Success
0% (All Easy)	275	96.82	310	99.24
25%	288	80.45	415	97.87
50%	272	55.71	520	96.54
75%	291	32.22	625	95.82
100% (All Hard)	274	12.85	730	94.71
Average	**280.0**	**55.61**	**520.0**	**96.84**

Cost-benefit analysis. R^2Edit's adaptivity is validated by its cost-benefit trade-off across task difficulties. As shown in Table 5, when hard samples are rare, R^2Edit remains effective, with latency comparable to AlphaEdit. AlphaEdit runs at nearly constant 274ms regardless of sample complexity, but its success rate drops from 96.82% to 12.85% as difficulty increases. In contrast, R^2Edit scales computation with task demand, spending up to 730ms on the hardest samples to preserve editing integrity. On average, it achieves a robust 96.84% success rate with 520ms latency on Llama3. This trade-off shows R^2Edit allocates computation where robustness is needed, bridging speed and reliability.

6 Conclusions

We formalize relation replacement editing and expose a key weakness of existing methods: they retain outdated information and fail on difficult edits. We address this with two contributions: the Forgetting-and-Editing (FE) framework, which introduces a targeted unlearning strategy to resolve knowledge conflicts, and R^2Edit, a self-paced algorithm for edits of varying difficulty. Experiments validate both: FE is effective on our relation replacement editing benchmark, and R^2Edit achieves state of the art on this task and object editing benchmarks. Despite these gains, permanently erasing obsolete relations remains challenging, so future work will develop more effective unlearning mechanisms for relation replacement editing.

Limitations

Our work has several limitations. First, ReEditBench is partially constructed using LLM-generated relation transformations, which may not fully capture real-world relational complexity and long-tail cases. Second, our theoretical analysis is based on the linear-regression formulation of locate-then-edit methods, and may not generalize to all editing paradigms. Third, although the proposed FE strategy reduces outdated knowledge retention, complete and irreversible forgetting remains unsolved. Finally, R^2Edit introduces additional iterative optimization overhead and currently focuses only on single-relation edits, leaving multi-relation and reasoning-consistent editing for future work.

Ethics Statement

Data sources, licenses, and intended use. ReEditBench is constructed from publicly available factual resources, primarily ZsRE (Levy et al., 2017) and Wikidata (Vrandečić and Krötzsch, 2014), and is used only for research and benchmarking of knowledge editing methods. We also evaluate on established benchmarks such as RaKE (Wei et al., 2023) and CounterFact (Meng et al., 2022). We cite the creators of all reused artifacts and follow their intended research use. The released code and dataset will include explicit license information, attribution, and usage terms.

Privacy, offensive content, and documentation. The benchmark is built from structured public factual triples rather than private user-generated text. It may contain public entity names, including public figures, but we do not intentionally collect private personal information such as email addresses, phone numbers, home addresses, or user identifiers. Generated candidates are screened through structural checks, LLM-based verification, and human validation; instances identified as privacy-sensitive, offensive, logically inconsistent, or implausible are removed. The release repository will provide documentation describing data sources, task format, construction pipeline, filtering and validation procedures, dataset statistics, intended use, limitations, and licensing information.

References

Yoshua Bengio, Jérôme Louradour, Ronan Collobert, and Jason Weston. 2009. Curriculum learning. In *Proceedings of the 26th annual international conference on machine learning, ICML*, pages 41–48.

Luisa Bentivogli, Peter Clark, Ido Dagan, and Danilo Giampiccolo. 2009. The fifth pascal recognizing textual entailment challenge. *TAC*, 7(8):1.

Nicola Cao, Wilker Aziz, and Ivan Titov. 2021. Editing factual knowledge in language models. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing, EMNLP*, pages 6491–6506.

Sungmin Cha, Sungjun Cho, Dasol Hwang, and Moontae Lee. 2024. Towards robust and parameter-efficient knowledge unlearning for llms. *arXiv preprint arXiv:2408.06621*.

Bill Dolan and Chris Brockett. 2005. Automatically constructing a corpus of sentential paraphrases. In *Third international workshop on paraphrasing (IWP2005)*.

Zenghao Duan, Wenbin Duan, Zhiyi Yin, Yinghan Shen, Shaoling Jing, Jie Zhang, Huawei Shen, and Xueqi Cheng. 2025. Related knowledge perturbation matters: Rethinking multiple pieces of knowledge editing in same-subject. In *Proceedings of the 2025 Conference of the Nations of the Americas Chapter of the Association for Computational Linguistics: Human Language Technologies (Volume 2: Short Papers)*, pages 363–373.

Junfeng Fang, Houcheng Jiang, Kun Wang, Yunshan Ma, Jie Shi, Xiang Wang, Xiangnan He, and Tat-Seng Chua. 2025. Alphaedit: Null-space constrained model editing for language models. In *The Thirteenth International Conference on Learning Representations, ICLR*.

Chendi Ge, Xin Wang, Zeyang Zhang, Hong Chen, Jiapei Fan, Longtao Huang, Hui Xue, and Wenwu Zhu. 2025. Dynamic mixture of curriculum lora experts for continual multimodal instruction tuning. In *Forty-second International Conference on Machine Learning, ICML*.

Alex Graves, Marc G Bellemare, Jacob Menick, Rémi Munos, and Koray Kavukcuoglu. 2017. Automated curriculum learning for neural networks. In *Proceedings of the 34th International Conference on Machine Learning-Volume 70, ICML*, pages 1311–1320.

Jia-Chen Gu, Hao-Xiang Xu, Jun-Yu Ma, Pan Lu, Zhen-Hua Ling, Kai-Wei Chang, and Nanyun Peng. 2024. Model editing harms general abilities of large language models: Regularization to the rescue. In *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing, EMNLP*, pages 16801–16819.

Daya Guo, Dejian Yang, Haowei Zhang, Junxiao Song, Ruoyu Zhang, Runxin Xu, Qihao Zhu, Shirong Ma, Peiyi Wang, Xiao Bi, and 1 others. 2025. Deepseek-r1: Incentivizing reasoning capability in llms via reinforcement learning. *arXiv preprint arXiv:2501.12948*.

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

Dan Hendrycks, Collin Burns, Steven Basart, Andy Zou, Mantas Mazeika, Dawn Song, and Jacob Steinhardt. 2021. Measuring massive multitask language understanding. In *International Conference on Learning Representations, ICLR*.

Tuan Hoang, Santu Rana, Sunil Gupta, and Svetha Venkatesh. 2024. Learn to unlearn for deep neural networks: Minimizing unlearning interference with gradient projection. In *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision*, pages 4819–4828.

Houcheng Jiang, Junfeng Fang, Tianyu Zhang, An Zhang, Ruipeng Wang, Tao Liang, and Xiang Wang. 2024. Neuron-level sequential editing for large language models. *arXiv preprint arXiv:2410.04045*.

Dahyun Jung, Jaehyung Seo, Jaewook Lee, Chanjun Park, and Heui-Seok Lim. 2025. Come: An unlearning-based approach to conflict-free model editing. In *Proceedings of the 2025 Conference of the Nations of the Americas Chapter of the Association for Computational Linguistics: Human Language Technologies (Volume 1: Long Papers), NAACL*, pages 6410–6422.

M Kumar, Benjamin Packer, and Daphne Koller. 2010. Self-paced learning for latent variable models. *Advances in neural information processing systems, NeurIPS*, 23.

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

Xiaopeng Li, Shasha Li, Shezheng Song, Jing Yang, Jun Ma, and Jie Yu. 2024. Pmet: Precise model editing in a transformer. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 38, pages 18564–18572.

Xiaopeng Li, Shangwen Wang, Shasha Li, Shezheng Song, Bin Ji, Ma Jun, and Jie Yu. 2026. Rethinking residual distribution in locate-then-edit model editing. *Advances in Neural Information Processing Systems*, 38:78348–78374.

Aixin Liu, Bei Feng, Bing Xue, Bingxuan Wang, Bochao Wu, Chengda Lu, Chenggang Zhao, Chengqi Deng, Chenyu Zhang, Chong Ruan, and 1 others. 2024a. Deepseek-v3 technical report. *arXiv preprint arXiv:2412.19437*.

Liangxin Liu, Xuebo Liu, Derek F Wong, Dongfang Li, Ziyi Wang, Baotian Hu, and Min Zhang. 2024b. Selectit: Selective instruction tuning for llms via uncertainty-aware self-reflection. *Advances in Neural Information Processing Systems, NeruIPS*, 37:97800–97825.

Qingyuan Liu, Jia-Chen Gu, Yunzhi Yao, Hong Wang, and Nanyun Peng. 2025a. Energy-regularized sequential model editing on hyperspheres. *arXiv preprint arXiv:2510.01172*.

Xiyu Liu, Zhengxiao Liu, Naibin Gu, Zheng Lin, Wanli Ma, Ji Xiang, and Weiping Wang. 2025b. Relation also knows: Rethinking the recall and editing of factual associations in auto-regressive transformer language models. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 39, pages 24659–24667.

Jun-Yu Ma, Hong Wang, Hao-Xiang Xu, Zhen-Hua Ling, and Jia-Chen Gu. 2025. Perturbation-restrained sequential model editing. In *The Thirteenth International Conference on Learning Representations, ICLR*.

Kevin Meng, David Bau, Alex J Andonian, and Yonatan Belinkov. 2022. Locating and editing factual associations in GPT. In *Advances in Neural Information Processing Systems, NeurIPS*.

Kevin Meng, Arnab Sen Sharma, Alex J Andonian, Yonatan Belinkov, and David Bau. 2023. Mass-editing memory in a transformer. In *The Eleventh International Conference on Learning Representations, ICLR*.

Meta. 2024. Llama 3. Large language model release.

Eric Mitchell, Charles Lin, Antoine Bosselut, Chelsea Finn, and Christopher D. Manning. 2022. Fast model editing at scale. In *International Conference on Learning Representations, ICLR*.

Yifan Niu, Miao Peng, Nuo Chen, Yatao Bian, Tingyang Xu, and Jia Li. 2025. Reledit: Evaluating conceptual knowledge editing in language models via relational reasoning. In *Findings of the Association for Computational Linguistics: ACL 2025*, pages 10220–10238.

Haowen Pan, Xiaozhi Wang, Yixin Cao, Zenglin Shi, Xun Yang, Juanzi Li, and Meng Wang. 2025. Precise localization of memories: A fine-grained neuron-level knowledge editing technique for LLMs. In *The Thirteenth International Conference on Learning Representations, ICLR*.

Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, Ilya Sutskever, and 1 others. 2019. Language models are unsupervised multitask learners. *OpenAI blog*, 1(8):9.

Richard Socher, Alex Perelygin, Jean Wu, Jason Chuang, Christopher D Manning, Andrew Y Ng, and Christopher Potts. 2013. Recursive deep models for semantic compositionality over a sentiment treebank. In *Proceedings of the 2013 conference on empirical methods in natural language processing, EMNLP*, pages 1631–1642.

Denny Vrandečić and Markus Krötzsch. 2014. Wikidata: a free collaborative knowledgebase. *Communications of the ACM*, 57(10):78–85.

Ben Wang and Aran Komatsuzaki. 2021. Gpt-j-6b: A 6 billion parameter autoregressive language model.

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

Song Wang, Yaochen Zhu, Haochen Liu, Zaiyi Zheng, Chen Chen, and Jundong Li. 2024b. Knowledge editing for large language models: A survey. *ACM Computing Surveys*, 57(3):1–37.

Alex Warstadt, Amanpreet Singh, and Samuel R Bowman. 2019. Neural network acceptability judgments. *Transactions of the Association for Computational Linguistics, TACL*, 7:625–641.

Yifan Wei, Xiaoyan Yu, Huanhuan Ma, Fangyu Lei, Yixuan Weng, Ran Song, and Kang Liu. 2023. Assessing knowledge editing in language models via relation perspective. *arXiv preprint arXiv:2311.09053*.

Yifan Wei, Xiaoyan Yu, Yixuan Weng, Huanhuan Ma, Yuanzhe Zhang, Jun Zhao, and Kang Liu. 2024. Does knowledge localization hold true? surprising differences between entity and relation perspectives in language models. In *Proceedings of the 33rd ACM International Conference on Information and Knowledge Management*, pages 4118–4122.

Adina Williams, Nikita Nangia, and Samuel R Bowman. 2017. A broad-coverage challenge corpus for sentence understanding through inference. *arXiv preprint arXiv:1704.05426*.

Jiakuan Xie, Pengfei Cao, Yubo Chen, Kang Liu, and Jun Zhao. 2025. Revealing the deceptiveness of knowledge editing: A mechanistic analysis of superficial editing. *arXiv preprint arXiv:2505.12636*.

Yuanshun Yao, Xiaojun Xu, and Yang Liu. 2024. Large language model unlearning. *Advances in Neural Information Processing Systems, NeurIPS*, 37:105425–105475.

Fangcong Yin, Xi Ye, and Greg Durrett. 2024. Lofit: Localized fine-tuning on llm representations. *Advances in Neural Information Processing Systems, NeurIPS*, 37:9474–9506.

Xiaojian Yuan, Tianyu Pang, Chao Du, Kejiang Chen, Weiming Zhang, and Min Lin. 2025. A closer look at machine unlearning for large language models. In *The Thirteenth International Conference on Learning Representations, ICLR*.

Mengqi Zhang, Zisheng Zhou, Xiaotian Ye, Qiang Liu, Zhaochun Ren, Zhumin Chen, and Pengjie Ren. 2025a. Disentangling knowledge representations for large language model editing. *arXiv preprint arXiv:2505.18774*.

Zhuoran Zhang, Yongxiang Li, Zijian Kan, Keyuan Cheng, Lijie Hu, and Di Wang. 2025b. Locate-then-edit for multi-hop factual recall under knowledge editing. In *Forty-second International Conference on Machine Learning, ICML*.

Chen Zhu, Ankit Singh Rawat, Manzil Zaheer, Srinadh Bhojanapalli, Daliang Li, Felix Yu, and Sanjiv Kumar. 2020. Modifying memories in transformer models. *arXiv preprint arXiv:2012.00363*.

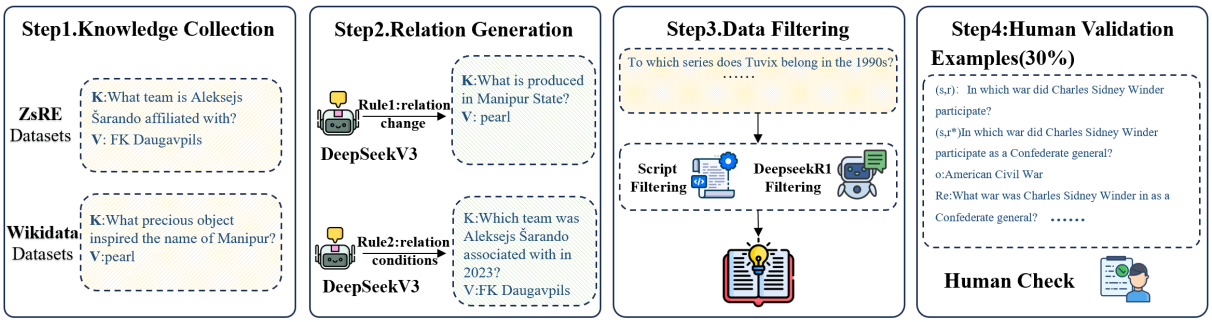

Figure 8: The construction process of our dataset.

A EXPERIMENTAL SETUP

A.1 Dataset Construction Details

A.1.1 Construction of ReEditBench.

This section provides a detailed breakdown of the four-stage pipeline used to construct our **ReEditBench** benchmark. The overall construction process is illustrated in Fig. 8. Our pipeline, which yields 7,918 high-quality relation replacement editing instances, is detailed below.

Stage 1: Foundational Knowledge Selection. We begin by selecting a foundational knowledge pool from established and high-quality benchmarks, which primarily include ZsRE and a curated subset of Wikidata. These sources are chosen for their structured and fact-checked subject-relation-object triples (s, r, o). By utilizing these expert-verified repositories, we ensure a factually grounded starting point for our dataset and successfully avoid the noise typically associated with sourcing facts from raw web text.

Stage 2: Systematic Relation Generation via DeepSeek-V3. Building upon the curated triples, we employ DeepSeek-V3 to systematically reframe each fact into a precise relation-replacement task. This generation process is guided by expert-designed prompts that instruct the model to produce a target edited triple (s, r^*, o) by modifying the original relation r while keeping the subject s and object o fixed. To ensure the diversity and comprehensiveness of the benchmark, we formulate two distinct patterns of relation updates, which include New Relation for core modifications and Conditional Relation for contextual constraints.

Stage 3: Multi-tier Automated Filtering and Verification. To guarantee the structural and semantic integrity of the generated candidates, we implement a rigorous two-phase automated verification framework. First, a script-based audit is conducted to validate the structural consistency of all instances, where any malformed entries involving empty fields or incorrect formatting are strictly discarded. Second, we utilize an independent reasoning model, DeepSeek-R1, to perform cross-verification. This verifier is tasked with assessing the factual and semantic plausibility of each candidate edit, identifying and removing any instances that contain logical contradictions or factual hallucinations.

Stage 4: Universal Human Scrutiny and Expert Validation. As the final and most critical quality assurance measure, we implement a comprehensive two-level human-in-the-loop validation process. Initially, we conduct an exhaustive manual review of every generated instance in the dataset to verify its fundamental correctness and adherence to the task definition. Following this universal screening, we randomly sample 30% of the dataset for a more rigorous expert validation. During this phase, expert annotators meticulously evaluate each sample for factual accuracy, logical consistency, and semantic coherence. The high inter-annotator agreement and the negligible error rate observed during this expert audit certify the exceptional fidelity of ReEditBench, ensuring it serves as a reliable benchmark for evaluating knowledge editing systems.

A.1.2 Construction and Usage of the Validation Set for Model Selection

Since our R^2Edit algorithm is an iterative process, a principled method is required to select the optimal model checkpoint. We achieve this by constructing a dedicated validation set and evaluating a composite loss function at each iteration.

1. Validation Set Construction. The validation set is created by randomly holding out 20% of the editing instances from the full training dataset before the main editing process begins. For each instance in this validation set, which consists of an original fact (s, r, o) and a target fact (s, r^*, o), we

define the following key-value pairs for evaluation:

- **Original Key-Value Pair**: $(\boldsymbol{k}_{\text{org}}, \boldsymbol{v}_{\text{org}})$, where $\boldsymbol{k}_{\text{org}}$ is the key vector corresponding to the original subject-relation pair (s, r), and $\boldsymbol{v}_{\text{org}}$ is the value vector for the object o.
- **New Key-Value Pair**: $(\boldsymbol{k}_{\text{new}}, \boldsymbol{v}_{\text{org}})$, where $\boldsymbol{k}_{\text{new}}$ is the key for the new relation (s, r^*). Note that the target value vector $\boldsymbol{v}_{\text{org}}$ remains the same.
- **Paraphrased Key**: $\boldsymbol{k}_{\text{re}}$, which is a semantic rephrasing of the new key $\boldsymbol{k}_{\text{new}}$. This is generated using an external LLM to test for generalization.
- **Forget Target**: $\boldsymbol{v}_{\text{forget}}$, which is the target value for the unlearning process, computed via interpolation as defined in Eq. (5): $\boldsymbol{v}_{\text{forget}} = \boldsymbol{v}_{\text{org}} + \gamma(\boldsymbol{v}_{\text{IDK}} - \boldsymbol{v}_{\text{org}})$.

2. Iterative Evaluation with a Weighted Loss Function. At each iteration t of the R^{2}Edit algorithm, we compute the perturbation $\boldsymbol{\Delta}_t$ and obtain the intermediate edited model weights $\mathbf{W}_t = \mathbf{W} + \boldsymbol{\Delta}_t \mathbf{P}$. We then evaluate this model on the validation set by calculating three distinct loss components based on the squared L2-norm loss function $\ell(\boldsymbol{k}, \boldsymbol{v}) = \left\|\mathbf{W}_t \boldsymbol{k} - \boldsymbol{v}\right\|_2^2$:

1. **Forgetting Loss ($\mathcal{L}_{\text{forget}}$)**: This loss measures how successfully the model unlearns the original, outdated fact. It is calculated by ensuring the output for the original key $\boldsymbol{k}_{\text{org}}$ moves towards the forget target $\boldsymbol{v}_{\text{forget}}$. This addresses two of your requirements: that the model learns to associate $\boldsymbol{k}_{\text{org}}$ with $\boldsymbol{v}_{\text{forget}}$, and consequently, that its association with $\boldsymbol{v}_{\text{org}}$ is suppressed.

$$\mathcal{L}_{\text{forget}}(t) = \mathbb{E}_{(\boldsymbol{k}_{\text{org}}, \boldsymbol{v}_{\text{forget}})} \left[\left\|\mathbf{W}_t \boldsymbol{k}_{\text{org}} - \boldsymbol{v}_{\text{forget}}\right\|_2^2 \right] \tag{12}$$

2. **Efficacy Loss ($\mathcal{L}_{\text{efficacy}}$)**: This loss assesses the direct acquisition of the new knowledge. It is the error between the model's output for the new key $\boldsymbol{k}_{\text{new}}$ and the correct value $\boldsymbol{v}_{\text{org}}$.

$$\mathcal{L}_{\text{efficacy}}(t) = \mathbb{E}_{(\boldsymbol{k}_{\text{new}}, \boldsymbol{v}_{\text{org}})} \left[\left\|\mathbf{W}_t \boldsymbol{k}_{\text{new}} - \boldsymbol{v}_{\text{org}}\right\|_2^2 \right] \tag{13}$$

3. **Generalization Loss ($\mathcal{L}_{\text{gen}}$)**: This loss evaluates whether the model can apply the new knowledge to paraphrased prompts, ensuring semantic understanding rather than superficial memorization.

$$\mathcal{L}_{\text{gen}}(t) = \mathbb{E}_{(\boldsymbol{k}_{\text{re}}, \boldsymbol{v}_{\text{org}})} \left[\left\|\mathbf{W}_t \boldsymbol{k}_{\text{re}} - \boldsymbol{v}_{\text{org}}\right\|_2^2 \right] \tag{14}$$

where $\mathbb{E}[\cdot]$ denotes the average loss over all samples in the validation set.

3. Final Model Selection. The total validation loss at iteration T, denoted $\mathcal{L}_{\text{val}}(t)$, is a weighted sum of these three components.

$$\mathcal{L}_{\text{val}}(t) = w_{\text{forget}} \cdot \mathcal{L}_{\text{forget}}(t) + w_{\text{efficacy}} \cdot \mathcal{L}_{\text{efficacy}}(t) + w_{\text{gen}} \cdot \mathcal{L}_{\text{gen}}(t) \tag{15}$$

We use an early stopping strategy with a patience of 3 iterations to select the final model. We track the minimum validation loss, $\mathcal{L}_{\text{val}}^*$, observed so far. Training is terminated at the first iteration T where the validation loss has not improved (i.e., decreased by more than a threshold ϵ) for 3 consecutive iterations. The final model is the checkpoint that corresponds to the best observed validation loss $\mathcal{L}_{\text{val}}^*$ up to iteration T. In our experiments, the weights are set as hyperparameters to balance the trade-offs between these objectives. For instance, we might use $w_{\text{forget}} = 0.4$, $w_{\text{efficacy}} = 0.4$, and $w_{\text{gen}} = 0.2$.

A.2 Baseline Methods

ROME (Meng et al., 2022) is a one-shot locate-and-edit method for factual rewriting. It identifies a feed-forward key–value subspace for the target fact and applies a rank-one update $\boldsymbol{\Delta}\mathbf{W}$, with a KL regularizer preserving behavior on general text.

Fine-Tuning (FT) (Zhu et al., 2020) performs constrained fine-tuning on fact-specific prompts. It updates selected MLP projection matrices while using an L_2 proximity term or trust-region constraint to limit parameter drift.

MEMIT (Meng et al., 2023) extends locate-and-edit to batch editing. It updates multiple critical MLP layers with a closed-form regularized least-squares solution that balances target fitting and locality preservation.

RECT (Gu et al., 2024) formulates editing as low-rank layer-wise correction. It selects contiguous causally relevant MLP layers, updates their down-projection matrices, and constrains both output drift and hidden-state shift.

NSE (Jiang et al., 2024) performs neuron-level editing in feed-forward layers. It identifies fact-relevant neurons by attribution scores and learns scaling and bias terms, with KL regularization to reduce unrelated side effects.

PRUNE (Ma et al., 2025) treats editing as sparse subspace adaptation. It masks salient MLP parameters, freezes the rest, and trains a low-rank adapter

on the selected subspace with a KL locality regularizer.

AlphaEdit (Fang et al., 2025) adds null-space projection to locate-and-edit. Given a target update $\mathbf{\Delta}$, it projects the perturbation with $\mathbf{P} = \mathbf{U}_0\mathbf{U}_0^\top$, where $\mathbf{U}_0$ spans the left null space of preserved-knowledge keys, so that $\mathbf{\Delta P K}_0 = \mathbf{0}$.

A.3 Metrics

A.3.1 ZsRE METRICS

Following the previous work, this section defines each ZsRE metric given a LLM f_θ, a knowledge fact prompt (s_i, r_i), an edited target output o_i, and the model's original output o_c^i:

- **Efficacy:** Efficacy is calculated as the average top-1 accuracy on the edit samples:

$$\mathbb{E}_i \left\{ o_i = \arg\max_o \mathbb{P}_{f_\theta}(o|(s_i, r_i)) \right\} \tag{16}$$

- **Generalization:** Generalization measures the model's performance on equivalent prompt of (s_i, r_i), such as rephrased statements $N((s_i, r_i))$. This is evaluated by the average top-1 accuracy on these $N((s_i, r_i))$:

$$\mathbb{E}_i \left\{ o_i = \arg\max_o \mathbb{P}_{f_\theta}(o|N((s_i, r_i))) \right\} \tag{17}$$

- **Specificity:** Specificity ensures that the editing does not affect samples unrelated to the edit cases $O(s_i, r_i)$. This is evaluated by the top-1 accuracy of predictions that remain unchanged:

$$\mathbb{E}_i \left\{ o_i^c = \arg\max_o P_{f_\theta}(o|O((s_i, r_i))) \right\} \tag{18}$$

A.3.2 COUNTERFACT METRICS

Following previous work, this section defines the evaluation metrics for the Counterfact dataset. To ensure a consistent and fair comparison with the ZsRE benchmark, we adopt its top-1 accuracy-based evaluation methodology for the Efficacy, Generalization, and Specificity metrics. Therefore, we only present the definitions for the remaining metrics unique to this generative task evaluation:

- **Fluency (generation entropy):** Measure for excessive repetition in model outputs. It uses the entropy of n-gram distributions:

$$-\frac{2}{3}\sum_k g_2(k)\log_2 g_2(k) + \frac{4}{3}\sum_k g_3(k)\log_2 g_3(k) \tag{19}$$

- **Consistency (reference score):** The consistency of the model's outputs is evaluated by computing the cosine similarity between the TF-IDF vectors of the model-generated text and a reference Wikipedia text.

A.3.3 ReEditBench METRICS

This section defines each ReEditBench metric given an original fact (s, r, o) and a new fact (s, r^*, o):

- **Success:** The Success score is a joint metric that holistically verifies if a knowledge edit was successful. As formulated in Eqn. 20, it requires two conditions to be met simultaneously: (i) the model must no longer predict the original object o for the original query (s, r) and (ii) it must correctly predict the new object o for the updated query (s, r^*).

$$\mathbb{E}_{x\sim\mathcal{D}}\Big[\mathbf{1}\Big\{o_i = \arg\max_{\neg o} \mathbb{P}_{f_\theta}(o \mid (s, r))\Big\}, \mathbf{1}\Big\{o_i = \arg\max_o \mathbb{P}_{f_\theta}(o \mid (s, r^*))\Big\}\Big] \tag{20}$$

- **Retention:** The Retention metric evaluates the extent to which the model incorrectly persists in recalling the outdated relationship after the edit. As defined in Eqn. 21, it measures the probability that the model still predicts the original object o when presented with the original prompt (s, r). In the context of relation replacement editing, a lower Retention score is preferred, as it indicates the successful erasure of conflicting, obsolete information.

$$\mathbb{E}_i \left\{ o_i = \arg\max_o \mathbb{P}_{f_\theta}(o|(s, r)) \right\} \tag{21}$$

- **Efficacy:** The Efficacy score measures the model's direct acquisition of the new fact. It is defined in Eqn. 22 as the probability that the new object o_i is the top prediction for the new prompt (s, r^*). A high score signifies that the new knowledge has been successfully instilled.

$$\mathbb{E}_i \left\{ o_i = \arg\max_o \mathbb{P}_{f_\theta}(o|(s, r^*)) \right\} \tag{22}$$

- **Generalization:** This metric evaluates if the model can apply the new knowledge beyond the specific prompt it was edited on. As shown in Eqn. 23, it measures the model's ability to predict the correct object o' when presented with a set of paraphrased or semantically equivalent prompts $N((s, r^*))$:

$$\mathbb{E}_i \left\{ o = \arg\max_{o'} P_{f_\theta}(o'|N((s, r^*))) \right\} \tag{23}$$

Table 6: Base configuration parameters for the LLMs used in the experiments.

Parameter	Value	Description
model_name	`EleutherAI_gpt-j-6B`, `gpt2-xl`, `Llama3-8B`	Specifies the pretrained language model.
layers	[3-8], [13-17], [4-8]	The target Transformer layers for editing.
v_num_grad_steps	25 or 20	Number of gradient steps for value vector computation.
v_lr	`5e-1` or `1e-1`	Learning rate used during value vector computation.
v_loss_layer	27, 47, 31	The specific model layer used to compute the edit loss.
kl_factor	`0.0625`	Weight of the KL-divergence regularization term.
mom2_dataset	`wikipedia`	Dataset for computing second-moment statistics.
rewrite_module_tmp	Varies by model	Template for the path to the module being rewritten.

A.4 Experimental details

This appendix details the hyperparameters used in our experiments. It is divided into two parts: the first outlines the base configuration parameters for the Large Language Models (LLMs), and the second elaborates on the key hyperparameters for our proposed R^{2}Edit and Forgetting-and-Editing (FE) strategies.

A.4.1 Model Configuration Parameters

The following table summarizes the main configuration parameters used for each of the three base models. These are primarily defined within their respective JSON configuration files.

Key Hyperparameters for the R^{2}Edit and FE Strategies In addition to the base configurations, our proposed algorithms are governed by several key hyperparameters that control the editing and forgetting behavior.

- **Forgetting Interpolation Factor (γ)**: This is the core hyperparameter of our FE strategy, as defined in Eqn. 5. It controls the degree of interpolation from the original fact's value vector, $\boldsymbol{v}(o)$, towards a neutral "I don't know" state, $\boldsymbol{v}(\text{IDK})$. A higher γ value enforces a more thorough forgetting of the outdated information. In our experiments, this was set to 0.5 for GPT-J-6B LLaMA3-8B and GPT2-XL to achieve an optimal balance between forgetting and learning.

- **Update Regularization Coefficients (α and β)**: These coefficients in the R^{2}Edit objective function (Eqn. 6) regularize the update perturbation $\mathbf{\Delta}$ to maintain model stability.
  - α constrains the overall magnitude of the update, preventing large, potentially disruptive changes to the model's parameters.
  - β minimizes the edit's impact on a set of preserved knowledge keys $\mathbf{K}_p$, ensuring that unrelated information is not corrupted.

  Throughout our experiments, we set $\alpha = 10$ and $\beta = 1$ to apply strong general regularization while precisely preserving prior knowledge.

- **Self-Paced Learning Curriculum Parameters (λ_0, μ, and T)**: These parameters define the "easy-to-hard" curriculum for the R^{2}Edit algorithm, as outlined in Algorithm 1.
  - λ_0 (Initial Pace Parameter): The initial difficulty threshold, which determines the set of the "easiest" samples to be edited at the beginning of the process.
  - μ (Pace Growth Factor): The multiplicative factor by which the difficulty threshold λ is increased in each iteration ($\lambda \leftarrow \mu\lambda$). This controls the pace at which more challenging samples are introduced into the training set.
  - T (Max Iterations): The Max number of iterations in the curriculum, defining the overall length of the optimization process.

  For our experiments, we set the initial pace to $\lambda_0 = 10$, the growth factor to $\mu = 1.4$, and the total number of iterations to $T = 20$. This configuration allows the model to first converge on easy edits before gradually incorporating more difficult ones, enhancing overall robustness and success rate.

B Implementation details and related proofs

B.1 Theoretical Analysis of the Forgetting-and-Editing Strategy

We analyze the proposed Forgetting-and-Editing (FE) strategy within the linear regression framework. The core of our strategy is to generate a mod-

ified representation for the target object through interpolation:

$$\boldsymbol{v}(\hat{o}) = \boldsymbol{v}(o) + \gamma[\boldsymbol{v}(\text{IDK}) - \boldsymbol{v}(o)], \quad \gamma \in (0, 1). \tag{24}$$

This operation is performed for each sample in the forgetting set $\mathcal{D}_b$. To analyze its effect within the regression framework, we translate this representation-level operation into label-space formulation. For any sample i in $\mathcal{D}_b$, the modified target label becomes:

$$\boldsymbol{v}(\hat{o}_i) = (1 - \gamma)\boldsymbol{v}(o_i) + \gamma\boldsymbol{v}(\text{IDK}). \tag{25}$$

Extending this operation to the entire forgetting set of M samples, we define:

- The original label vector: $\boldsymbol{y}_b = [\boldsymbol{v}(o_1), \boldsymbol{v}(o_2), ..., \boldsymbol{v}(o_M)]^\top$
- The IDK label vector: $\boldsymbol{y}_{\text{IDK}} = [\boldsymbol{v}(\text{IDK}), \boldsymbol{v}(\text{IDK}), ..., \boldsymbol{v}(\text{IDK})]^\top$

The FE strategy effectively applies the same linear interpolation to each corresponding element of these vectors, yielding the modified label vector:

$$\begin{aligned} \boldsymbol{y}_b^{\text{FE}} &= [\boldsymbol{v}(\hat{o}_1), \boldsymbol{v}(\hat{o}_2), ..., \boldsymbol{v}(\hat{o}_M)]^\top \\ &= \boldsymbol{y}_b + \gamma(\boldsymbol{y}_{\text{IDK}} - \boldsymbol{y}_b). \end{aligned} \tag{26}$$

Substituting this into the closed-form solution of the linear regression problem yields:

$$\begin{aligned} \boldsymbol{w}_{\text{FE}}^* &= (\mathbf{X}^\top\mathbf{X})^{-1}(\mathbf{X}_g^\top\boldsymbol{y}_g + \mathbf{X}_b^\top\boldsymbol{y}_b^{\text{FE}}) \\ &= (\mathbf{X}^\top\mathbf{X})^{-1}\left(\mathbf{X}_g^\top\boldsymbol{y}_g + \mathbf{X}_b^\top\left[\boldsymbol{y}_b + \gamma(\boldsymbol{y}_{\text{IDK}} - \boldsymbol{y}_b)\right]\right) \\ &= \boldsymbol{w}_g^* + \gamma(\mathbf{X}^\top\mathbf{X})^{-1}\mathbf{X}_b^\top(\boldsymbol{y}_{\text{IDK}} - \boldsymbol{y}_b), \end{aligned} \tag{27}$$

where $\boldsymbol{w}_g^* = (\mathbf{X}^\top\mathbf{X})^{-1}\mathbf{X}_g^\top\boldsymbol{y}_g$ is the solution trained solely on normal data.

Comparative Advantages. The proposed Feature Editing (FE) strategy offers distinct advantages over methods that employ a fixed value (e.g., I don't know) or random answers for forgetting. Unlike the constant bias introduced by a fixed label or the uncontrolled bias from random assignment, our approach generates a non-constant, data-dependent bias term $\gamma(\mathbf{X}^\top\mathbf{X})^{-1}\mathbf{X}_b^\top(\boldsymbol{y}_{\text{IDK}} - \boldsymbol{y}_b)$. This key difference prevents systematic bias and preserves prediction diversity. Furthermore, the hyperparameter γ provides precise and continuous control over the forgetting strength, a feature unavailable in conventional methods. The optimization process remains stable due to the proximity between the original and target features, avoiding large gradients. Finally, the adjustment is highly targeted, effectively removing specific information while minimizing distortion to the model's normal knowledge.

B.2 Theoretical Lower Bound for Relation Forgetting

Preliminaries. Let $\mathbf{W} \in \mathbb{R}^{d_1 \times d_0}$ be the original FFN weight matrix. For a single edit, the locate-then-edit framework solves:

$$\begin{aligned} \mathbf{\Delta}^* = \arg\min_{\mathbf{\Delta}} &\|\mathbf{W} + \mathbf{\Delta}\mathbf{P})\mathbf{k}' - \hat{\mathbf{v}}\|_2^2 + \\ &\alpha\|\mathbf{\Delta}\mathbf{P}\|_F^2 + \beta\|\mathbf{\Delta}\mathbf{P}\mathbf{K}_p\|_F^2, \end{aligned} \tag{28}$$

where $\mathbf{P}$ is the null-space projection matrix. The closed-form solution is:

$$\mathbf{\Delta}^* = (\hat{\mathbf{v}} - \mathbf{W}\mathbf{k}')\mathbf{k}'^\top\mathbf{P}(\mathbf{k}'\mathbf{k}'^\top\mathbf{P} + \beta\mathbf{K}_p\mathbf{K}_p^\top\mathbf{P} + \alpha\mathbf{I})^{-1}. \tag{29}$$

Leakage to Original Key. The edited model's output for the original key $\mathbf{k}$ is:

$$\mathbf{W}'\mathbf{k} = \mathbf{W}\mathbf{k} + \mathbf{\Delta}^*\mathbf{k}. \tag{30}$$

We analyze the term $\mathbf{\Delta}^*\mathbf{k}$. Let $\mathbf{M} = (\mathbf{k}'\mathbf{k}'^\top\mathbf{P} + \beta\mathbf{K}_p\mathbf{K}_p^\top\mathbf{P} + \alpha\mathbf{I})^{-1}$. Then:

$$\mathbf{\Delta}^*\mathbf{k} = (\hat{\mathbf{v}} - \mathbf{W}\mathbf{k}') \cdot (\mathbf{k}'^\top\mathbf{P}\mathbf{M}\mathbf{k}). \tag{31}$$

The scalar $\tau = \mathbf{k}'^\top\mathbf{P}\mathbf{M}\mathbf{k}$ quantifies the "leakage" from the update at $\mathbf{k}'$ to $\mathbf{k}$. When $\mathbf{k} \approx \mathbf{k}'$ and the null-space projection $\mathbf{P}$ approximately preserves their inner product (since $\mathbf{P}$ is constructed from preserved knowledge keys $\mathbf{K}_p$ that are orthogonal to neither $\mathbf{k}$ nor $\mathbf{k}'$), we have:

$$\tau \approx \kappa \cdot \|\mathbf{k}'\|^2 \cdot \lambda_{\min}(\mathbf{M}) > 0, \tag{32}$$

where $\kappa = \cos(\mathbf{k}, \mathbf{k}')$. This confirms that updates leak proportionally to key similarity.

B.3 Formulation of the Multi-Objective Optimization Problem

The fundamental goal of parameter-modifying knowledge editing is to find a minimal perturbation $\mathbf{\Delta}$, to a model's weight matrix, $\mathbf{W}$, such that the edited model $\mathbf{W}' = \mathbf{W} + \mathbf{\Delta}$ reflects new knowledge without catastrophically forgetting existing information. This can be framed as a multi-objective optimization problem.

Let us define the key components:

- **New Knowledge (Update Set):** A set of new facts to be incorporated, represented by key-value pairs $\{(\boldsymbol{k}_i, \boldsymbol{v}_i)\}$. We can stack these into matrices $\mathbf{K}_1$ (keys) and $\mathbf{V}_1$ (values). The objective is to make the model output $\mathbf{V}_1$ when given $\mathbf{K}_1$. The error for this is captured by the term $\mathcal{L}_{\text{update}} = \|(\mathbf{W} + \mathbf{\Delta})\mathbf{K}_1 - \mathbf{V}_1\|_F^2$.

- **Preserved Knowledge (Preservation Set):** The vast set of existing knowledge that must remain unchanged. This is represented by key-value pairs $\{(\boldsymbol{k}_j, \boldsymbol{v}_j)\}$ stacked into matrices $\mathbf{K}_0$ and $\mathbf{V}_0$. Since the pre-trained model $\mathbf{W}$ is assumed to already store this knowledge, we have $\mathbf{W}\mathbf{K}_0 \approx \mathbf{V}_0$. The objective is to minimize the change in output for these keys, giving an error term $\mathcal{L}_{\text{preserve}} = \|(\mathbf{W} + \mathbf{\Delta})\mathbf{K}_0 - \mathbf{V}_0\|_F^2$.

- **Regularization:** To prevent the perturbation $\mathbf{\Delta}$ from becoming excessively large and harming the model's general abilities, a regularization term on the perturbation itself is included, $\mathcal{L}_{\text{reg}} = \|\mathbf{\Delta}\|_F^2$.

Combining these objectives, we arrive at the standard optimization problem for knowledge editing:

$$\min_{\Delta} \mathcal{L}(\mathbf{\Delta}) = \underbrace{\|(\mathbf{W} + \mathbf{\Delta})\mathbf{K}_1 - \mathbf{V}_1\|_F^2}_{\text{Update Error}} + \alpha \underbrace{\|(\mathbf{W} + \mathbf{\Delta})\mathbf{K}_0 - \mathbf{V}_0\|_F^2}_{\text{Preservation Error}} + \beta \underbrace{\|\mathbf{\Delta}\|_F^2}_{\text{Regularization}} \tag{33}$$

where α and β are hyperparameters that balance the trade-off between the objectives.

We can simplify this expression. Since $\mathbf{W}\mathbf{K}_0 = \mathbf{V}_0$, the preservation term becomes $\|(\mathbf{W} + \mathbf{\Delta})\mathbf{K}_0 - \mathbf{W}\mathbf{K}_0\|_F^2 = \|\mathbf{\Delta}\mathbf{K}_0\|_F^2$. For the update term, we can define the **residual matrix** $\mathbf{R} = \mathbf{V}_1 - \mathbf{W}\mathbf{K}_1$, which represents the error that the edit must correct. The term thus becomes $\|\mathbf{\Delta}\mathbf{K}_1 - \mathbf{R}\|_F^2$. The simplified objective is:

$$\min_{\Delta} \mathcal{L}(\mathbf{\Delta}) = \|\mathbf{\Delta}\mathbf{K}_1 - \mathbf{R}\|_F^2 + \alpha\|\mathbf{\Delta}\mathbf{K}_0\|_F^2 + \beta\|\mathbf{\Delta}\|_F^2 \tag{34}$$

Incorporating the Null-Space Projection. A key innovation from methods like AlphaEdit is to constrain the update $\mathbf{\Delta}$ to the null-space of the preserved knowledge. This theoretically guarantees that the edit does not interfere with this knowledge. This is achieved using a projection matrix $\mathbf{P}$.

- Let $\mathbf{K}_p$ be the matrix of keys for the knowledge we wish to explicitly preserve. $\mathbf{K}_p$ is the concrete realization of the abstract $\mathbf{K}_0$ used to build the projector.

- The null-space projection matrix $\mathbf{P}$ is constructed such that for any matrix $\mathbf{A}$, the update $\mathbf{AP}$ satisfies $(\mathbf{AP})\mathbf{K}_p = \mathbf{0}$. $\mathbf{P}$ is symmetric ($\mathbf{P} = \mathbf{P}^\top$) and idempotent ($\mathbf{P}^2 = \mathbf{P}$).

By replacing the raw perturbation $\mathbf{\Delta}$ with the projected perturbation $\mathbf{\Delta P}$, we enforce this non-interference constraint. The optimization objective is adapted to solve for an optimal update within this safe subspace. The preservation term is now implicitly handled by the projection, but can be kept as a soft constraint, while the regularization term is applied to the projected update. This leads to an objective of the form:

$$\min_{\Delta} \mathcal{L}(\mathbf{\Delta}) = \|\mathbf{\Delta P K}_1 - \mathbf{R}\|_F^2 + \alpha'\|\mathbf{\Delta P}\|_F^2 + \beta'\|\mathbf{\Delta P K}_p\|_F^2 \tag{35}$$

Here, the hyperparameters α' and β' now regularize the magnitude of the projected update and explicitly penalize any residual interference with the preserved set $\mathbf{K}_p$.

Introducing Self-Paced Learning (SPL). The final step in the paper's methodology (R^2Edit) is the introduction of a self-paced learning curriculum. This acknowledges that not all edits are equally difficult. The model should first learn from "easy" samples and gradually incorporate "harder" ones. This is implemented via a binary selection matrix $\mathbf{Z}$.

- $\mathbf{Z}$ is a diagonal matrix where each diagonal entry $z_i \in \{0, 1\}$.

- At each iteration, $z_i = 1$ if the i-th sample is deemed "easy" (i.e., its loss is below a certain threshold λ); otherwise, $z_i = 0$.

- This selection matrix is applied only to the **update error term**, effectively masking out the hard samples for the current iteration. To keep the objective quadratic, we use its square root, $\mathbf{Z}^{1/2}$ (which is equal to $\mathbf{Z}$ since its elements are 0 or 1).

By integrating the selection matrix $\mathbf{Z}$ into Eqn. 35, we arrive at the final optimization problem as formulated in the paper:

$$\min_{\Delta} \mathcal{L}(\mathbf{\Delta}) = \|(\mathbf{\Delta P K}_1 - \mathbf{R})\mathbf{Z}^{1/2}\|_F^2 + \alpha\|\mathbf{\Delta P}\|_F^2 + \beta\|\mathbf{\Delta P K}_p\|_F^2 \tag{36}$$

This final form is what the paper uses to derive a closed-form solution for the projected update $\mathbf{\Delta}_{\text{R}^2\text{Edit}} = \mathbf{\Delta P}$. The key terms are:

- $\mathbf{\Delta}$: The perturbation matrix we are solving for.
- $\mathbf{P}$: The null-space projection matrix, which is symmetric ($\mathbf{P} = \mathbf{P}^\top$) and idempotent ($\mathbf{P}^2 = \mathbf{P}$).
- $\mathbf{Z}$: The diagonal selection matrix ($\mathbf{Z} = \mathbf{Z}^\top$, $\mathbf{Z}^2 = \mathbf{Z}$, $\mathbf{Z}^{1/2} = \mathbf{Z}$).
- $\mathbf{K}_1, \mathbf{R}, \mathbf{K}_p, \alpha, \beta$: As defined previously.

B.4 Derivation of the Closed-Form Solution

The objective function $\mathcal{L}(\mathbf{\Delta})$ is convex with respect to $\mathbf{\Delta}$. We can find the minimum by taking the gradient with respect to $\mathbf{\Delta}$ and setting it to zero.

First, we expand the objective using the trace operator, as $\|\mathbf{X}\|_F^2 = \mathrm{Tr}(\mathbf{X}^\top\mathbf{X})$.

$$\begin{aligned}\mathcal{L}(\mathbf{\Delta}) &= \mathrm{Tr}\left(((\mathbf{\Delta}\mathbf{P}\mathbf{K}_1 - \mathbf{R})\mathbf{Z})^\top((\mathbf{\Delta}\mathbf{P}\mathbf{K}_1 - \mathbf{R})\mathbf{Z})\right) \\ &+\alpha\mathrm{Tr}\left((\mathbf{\Delta}\mathbf{P})^\top(\mathbf{\Delta}\mathbf{P})\right) + \beta\mathrm{Tr}\left((\mathbf{\Delta}\mathbf{P}\mathbf{K}_p)^\top(\mathbf{\Delta}\mathbf{P}\mathbf{K}_p)\right) \\ &= \mathrm{Tr}\left(\mathbf{Z}(\mathbf{K}_1^\top\mathbf{P}^\top\mathbf{\Delta}^\top - \mathbf{R}^\top)(\mathbf{\Delta}\mathbf{P}\mathbf{K}_1\mathbf{Z} - \mathbf{R}\mathbf{Z})\right) \\ &+\alpha\mathrm{Tr}\left(\mathbf{P}^\top\mathbf{\Delta}^\top\mathbf{\Delta}\mathbf{P}\right) + \beta\mathrm{Tr}\left(\mathbf{K}_p^\top\mathbf{P}^\top\mathbf{\Delta}^\top\mathbf{\Delta}\mathbf{P}\mathbf{K}_p\right)\end{aligned} \tag{37}$$

Using the properties $\mathbf{P} = \mathbf{P}^\top$ and the cyclic property of the trace, we can rewrite each term:

$$\begin{aligned}\mathcal{L}(\mathbf{\Delta}) &= \mathrm{Tr}(\mathbf{\Delta}\mathbf{P}\mathbf{K}_1\mathbf{Z}\mathbf{K}_1^\top\mathbf{P}\mathbf{\Delta}^\top) - 2\mathrm{Tr}(\mathbf{\Delta}\mathbf{P}\mathbf{K}_1\mathbf{Z}\mathbf{R}^\top) + \\ &\mathrm{Tr}(\mathbf{R}\mathbf{Z}\mathbf{R}^\top) + \alpha\mathrm{Tr}(\mathbf{\Delta}\mathbf{P}\mathbf{P}\mathbf{\Delta}^\top) + \beta\mathrm{Tr}(\mathbf{\Delta}\mathbf{P}\mathbf{K}_p\mathbf{K}_p^\top\mathbf{P}\mathbf{\Delta}^\top)\end{aligned} \tag{38}$$

Now, we compute the gradient $\nabla_{\mathbf{\Delta}}\mathcal{L}(\mathbf{\Delta})$. Using the matrix calculus identities $\nabla_{\mathbf{X}}\mathrm{Tr}(\mathbf{B}\mathbf{X}^\top) = \mathbf{B}$ and $\nabla_{\mathbf{X}}\mathrm{Tr}(\mathbf{X}\mathbf{B}\mathbf{X}^\top\mathbf{C}) = \mathbf{C}\mathbf{X}\mathbf{B} + \mathbf{C}^\top\mathbf{X}\mathbf{B}^\top$:

$$\begin{aligned}\nabla_{\mathbf{\Delta}}\mathcal{L}(\mathbf{\Delta}) =& 2(\mathbf{\Delta}\mathbf{P}\mathbf{K}_1\mathbf{Z}\mathbf{K}_1^\top\mathbf{P}) - 2(\mathbf{R}\mathbf{Z}\mathbf{K}_1^\top\mathbf{P}) \\ &+ 2\alpha(\mathbf{\Delta}\mathbf{P}\mathbf{P}) + 2\beta(\mathbf{\Delta}\mathbf{P}\mathbf{K}_p\mathbf{K}_p^\top\mathbf{P})\end{aligned} \tag{39}$$

Since $\mathbf{P} = \mathbf{P}^2$, we can simplify $\mathbf{\Delta}\mathbf{P}\mathbf{P} = \mathbf{\Delta}\mathbf{P}$. Setting the gradient to zero to find the minimum:

$$\begin{aligned}&2(\mathbf{\Delta}\mathbf{P}\mathbf{K}_1\mathbf{Z}\mathbf{K}_1^\top\mathbf{P}) - 2(\mathbf{R}\mathbf{Z}\mathbf{K}_1^\top\mathbf{P}) + 2\alpha(\mathbf{\Delta}\mathbf{P}) + \\ &2\beta(\mathbf{\Delta}\mathbf{P}\mathbf{K}_p\mathbf{K}_p^\top\mathbf{P}) = 0\end{aligned} \tag{40}$$

Dividing by 2 and rearranging to isolate terms with $\mathbf{\Delta}$:

$$\begin{aligned}&(\mathbf{\Delta}\mathbf{P}\mathbf{K}_1\mathbf{Z}\mathbf{K}_1^\top\mathbf{P}) + \alpha(\mathbf{\Delta}\mathbf{P}) + \\ &\beta(\mathbf{\Delta}\mathbf{P}\mathbf{K}_p\mathbf{K}_p^\top\mathbf{P}) = \mathbf{R}\mathbf{Z}\mathbf{K}_1^\top\mathbf{P}\end{aligned} \tag{41}$$

Let $\mathbf{\Delta}_{\text{R}^2\text{Edit}} = \mathbf{\Delta}\mathbf{P}$ represent the final projected update. We can factor $\mathbf{A}_{\text{R}^2\text{Edit}}$ out from the left-hand side:

$$\mathbf{\Delta}_{\text{R}^2\text{Edit}}(\mathbf{K}_1\mathbf{Z}\mathbf{K}_1^\top\mathbf{P} + \alpha\mathbf{I} + \beta\mathbf{K}_p\mathbf{K}_p^\top\mathbf{P}) = \mathbf{R}\mathbf{Z}\mathbf{K}_1^\top\mathbf{P} \tag{42}$$

Finally, by right-multiplying by the inverse of the term in the parenthesis, we obtain the closed-form solution for the effective update $\mathbf{\Delta}_{\text{R}^2\text{Edit}}$:

$$\mathbf{\Delta}_{\text{R}^2\text{Edit}} = (\mathbf{R}\mathbf{Z}\mathbf{K}_1^\top\mathbf{P})(\mathbf{K}_1\mathbf{Z}\mathbf{K}_1^\top\mathbf{P} + \beta\mathbf{K}_p\mathbf{K}_p^\top\mathbf{P} + \alpha\mathbf{I})^{-1} \tag{43}$$

This is the rigorous derivation for the update rule. The matrix $(\mathbf{K}_1\mathbf{Z}\mathbf{K}_1^\top\mathbf{P} + \beta\mathbf{K}_p\mathbf{K}_p^\top\mathbf{P} + \alpha\mathbf{I})$ is guaranteed to be invertible because $\mathbf{K}_1\mathbf{Z}\mathbf{K}_1^\top\mathbf{P}$ and $\beta\mathbf{K}_p\mathbf{K}_p^\top\mathbf{P}$ are positive semi-definite, and the addition of the regularizer $\alpha\mathbf{I}$ (for $\alpha > 0$) makes the entire matrix positive definite and thus invertible.

Some papers may present a slightly simplified version of this formula. The version derived here is the one that follows directly and rigorously from the stated optimization objective. For instance, the final $\mathbf{P}$ in the term $\mathbf{R}\mathbf{Z}\mathbf{K}_1^\top\mathbf{P}$ might be omitted in some implementations, but including it is mathematically consistent as the entire equation operates within the projected subspace.

R^2Edit as Pareto Frontier Navigation. The baseline objective in Eq. (36) seeks a single perturbation balancing update error, preservation, and regularization. This is inherently a multi-objective problem with competing goals: minimizing $\|\mathbf{\Delta}\mathbf{P}\mathbf{K}_1 - \mathbf{R}\|_F^2$ for hard samples requires large $\|\mathbf{\Delta}\mathbf{P}\|_F$, which conflicts with the α-regularizer.

R^2Edit resolves this conflict temporally rather than spatially. Each iteration t optimizes a restricted objective:

$$\min_{\mathbf{\Delta}} \|\mathbf{\Delta}\mathbf{P}\mathbf{K}_1^{(t)} - \mathbf{R}^{(t)}\|_F^2 + \alpha\|\mathbf{\Delta}\mathbf{P}\|_F^2 + \beta\|\mathbf{\Delta}\mathbf{P}\mathbf{K}_p\|_F^2, \tag{44}$$

where $\mathbf{K}_1^{(t)}, \mathbf{R}^{(t)}$ contain only samples in $\mathcal{S}_t$. The sequence $\{\mathbf{\Delta}^{(t)}\}_{t=1}^T$ traces a path through the Pareto frontier of the full multi-objective problem, starting from the easy-sample-dominated region (high preservation, moderate update) and progressing toward the hard-sample region (aggressive update, tight preservation). The validation-based selection chooses the point on this path with optimal trade-off, avoiding the suboptimality of any single fixed compromise.

C More Experimental Results

C.1 Additional Experimental Results on Object Editing

To provide a more comprehensive validation of our method, we present further results on the demanding CounterFact benchmark. This dataset is particularly challenging, focusing on counter-intuitive factual edits that require precise model updates.

Table 7: Side-by-side comparison of forgetting strategies across all models and key methods. For each strategy, we report Retention (lower is better) and Efficacy (higher is better). The results show that the proposed target-smoothed forgetting strategy substantially reduces obsolete-relation retention while preserving competitive efficacy, yielding the best overall balance for R^2Edit.

LLM	Method	No-Forgetting		+ FE (IDK)		+ FE (Random)		+ FE (Ours)	
		Retention ↓	Efficacy ↑	Retention ↓	Efficacy ↑	Retention ↓	Efficacy ↑	Retention ↓	Efficacy ↑
LLaMA3	MEMIT	51.70	48.43	46.52	48.65	43.12	47.90	38.45	50.12
	ROME	60.47	50.91	55.30	51.10	52.88	50.55	44.20	52.34
	AlphaEdit	78.34	79.17	76.11	75.23	76.90	78.19	74.50	83.24
	R^2Edit	88.56	83.23	75.92	83.48	70.41	82.17	**62.77**	**87.37**
GPT2-XL	MEMIT	80.26	85.23	68.44	82.10	72.35	81.56	55.32	86.88
	ROME	61.71	61.70	55.40	59.88	58.12	60.20	42.15	64.50
	AlphaEdit	91.31	86.83	60.25	83.45	65.81	84.90	50.46	87.36
	R^2Edit	68.55	85.93	55.18	80.15	61.33	81.82	**48.78**	**88.46**
GPT-J	MEMIT	92.98	97.14	75.20	98.15	81.12	96.40	62.45	99.20
	ROME	63.34	63.32	52.10	64.20	55.67	63.90	45.10	66.88
	AlphaEdit	98.20	98.53	81.67	99.12	85.43	98.30	77.84	98.64
	R^2Edit	88.24	95.93	65.40	98.31	72.88	99.04	**59.84**	**99.28**

The results, detailed in Table 8, reinforce the superiority of R^2Edit. It achieves near-perfect Efficacy across all models while also setting a new state-of-the-art in Fluency, with scores like 631.11 on LLaMA3. This indicates that its edits not only correct facts but also produce higher-quality, more natural language. This is accomplished while maintaining strong Generalization and Specificity, demonstrating a robust and well-balanced editing profile even on this difficult benchmark.

Analysis of Sample Difficulty Distribution. To conduct a more rigorous evaluation, our experiments focus on curated subsets of recognized hard cases from ZsRE and CounterFact, rather than the full benchmarks which are often dominated by simple samples. We define sample difficulty using the initial residual norm, $\|\mathbf{v}_i - \mathbf{W}\mathbf{k}_i\|_2^2$, which measures the initial error. The difficulty distributions of these selected subsets are visualized in Fig. 9 and Fig. 10.

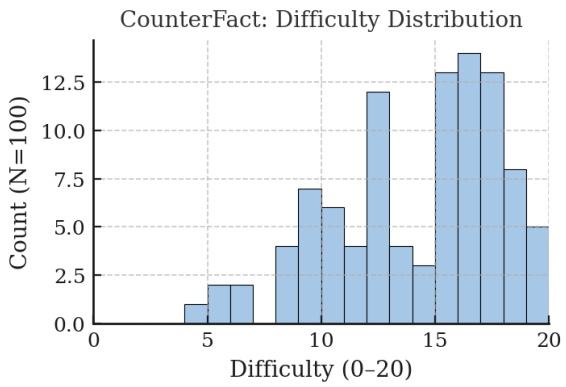

Figure 9: The ZsRE hard subset has a varied difficulty distribution.

As the figure illustrates, the two hard subsets present distinct challenge profiles. The selected ZsRE hard subset (Fig. 9) exhibits a mixed difficulty distribution, spanning a broad spectrum from medium to high difficulty. In contrast, the CounterFact hard subset (Fig. 10) constitutes a more extreme challenge, with nearly all samples concentrated in the high-difficulty range. This subset serves as an effective stress test for an algorithm's robustness.

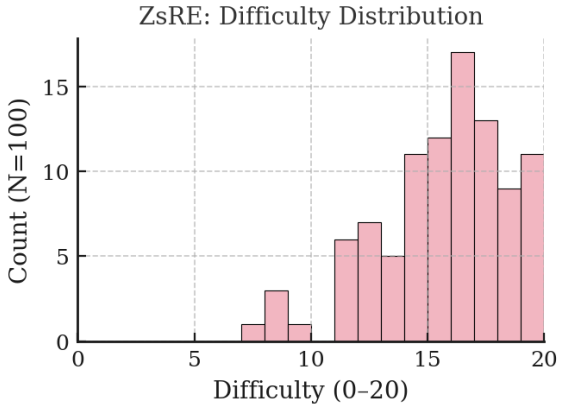

Figure 10: The CounterFact hard subset is heavily concentrated in the high-difficulty region.

This challenge-focused evaluation environment provides a compelling motivation for our proposed R^2Edit algorithm. The self-paced, "easy-to-hard" curriculum of R^2Edit is precisely engineered for such scenarios. It can intelligently identify the relatively easier samples even within a difficult set to begin the optimization process, building a robust update path to eventually solve the highly challenging edits where traditional, one-shot methods often fail.

C.2 Full Benchmark Performance and Saturation Analysis

To provide a comprehensive evaluation, we report the performance of `R²Edit` and baseline methods

Table 8: Object Editing Performance on the CounterFact Hard Subset. R^2Edit consistently outperforms prior methods, notably achieving state-of-the-art Fluency, indicating higher-quality text generation post-edit.

LLM	Method	Efficacy↑	Generalization↑	Specificity↑	Fluency↑	Consistency↑
LLaMA3	ROME	32.02	33.41	34.31	425.55	13.01
	MEMIT	69.22	65.61	30.54	629.68	53.15
	AlphaEdit	79.21	73.54	30.92	629.91	56.67
	R^2Edit (Ours)	**92.80**	**95.21**	**42.51**	**631.11**	**56.78**
GPT2-XL	ROME	39.42	30.01	5.82	592.64	65.09
	MEMIT	70.45	72.98	7.93	465.78	53.58
	AlphaEdit	83.22	83.91	8.54	621.76	55.62
	R^2Edit (Ours)	**92.66**	**94.82**	**9.62**	**629.26**	54.52
GPT-J	ROME	32.05	37.01	25.76	514.82	15.64
	MEMIT	79.22	78.27	27.58	618.93	57.84
	AlphaEdit	87.52	86.13	28.76	621.80	59.28
	R^2Edit (Ours)	**92.77**	**93.12**	**38.73**	**622.52**	**59.66**

Table 9: **Full Dataset Performance.** Comparison of editing methods on the complete CounterFact and ZsRE benchmarks. R^2`Edit` consistently achieves SOTA performance across all metrics.

LLM	Method	CounterFact					ZsRE		
		Eff. ↑	**Gen. ↑**	**Spe. ↑**	**Flu. ↑**	**Consis. ↑**	**Eff. ↑**	**Gen. ↑**	**Spe. ↑**
LLaMA3	ROME	64.40	61.42	49.44	449.06	3.31	2.01	1.80	0.69
	MEMIT	65.65	64.65	51.56	437.43	6.58	34.62	31.28	18.49
	AlphaEdit	98.90	94.22	67.88	622.49	32.40	94.47	91.13	32.55
	R^2Edit (Ours)	**99.24**	**94.62**	**69.37**	**624.69**	**33.73**	**95.72**	**93.07**	**33.25**
GPT2-XL	ROME	54.60	51.18	52.68	366.13	0.72	47.50	43.56	14.27
	MEMIT	94.70	85.82	60.50	477.26	22.72	79.17	71.44	26.42
	AlphaEdit	99.50	93.95	66.39	597.88	39.38	94.81	86.11	25.88
	R^2Edit (Ours)	**99.65**	**94.78**	**67.83**	**599.52**	**40.23**	**95.92**	**87.63**	**27.25**
GPT-J	ROME	57.50	54.20	52.05	589.42	3.22	56.42	54.65	9.86
	MEMIT	98.55	95.50	63.64	546.28	34.89	94.91	90.22	30.39
	AlphaEdit	99.75	96.38	75.48	618.50	42.08	99.79	96.00	28.29
	R^2Edit (Ours)	**99.82**	**96.82**	**76.23**	**620.35**	**44.33**	**99.83**	**97.12**	**30.47**

on the complete CounterFact and ZsRE datasets in Table 9. With the inclusion of these full-dataset results, we can approach the evaluation with a holistic perspective.

When these high aggregate scores are analyzed alongside the sample difficulty histograms presented in Fig. 9 and Fig. 10, a critical trend emerges: existing state-of-the-art methods have achieved near-saturation performance on the "easy" and "medium" portions of the data distribution. The primary failure mode for current technology lies almost exclusively within the "hard" tail. This observation validates our strategic focus on difficult subsets (as detailed in Appendix C.2); since the general case is largely solved, the frontier of knowledge editing research must shift toward these challenging, high-residual scenarios.

As shown in Tab. 9, R^2`Edit` not only dominates on the hard subsets but also consistently achieves the best performance across the full benchmarks, ensuring robustness not just on average, but where it matters most.

C.3 Extended Experimental Results

In this section, we provide the comprehensive experimental results across all benchmarks and models, including the baselines RECT (Gu et al., 2024) and Fine-Tuning (FT) (Zhu et al., 2020) that were omitted from the main text for brevity. Table 10 details the performance of all parametric editing methods.

C.4 GENERAL CAPABILITY TESTS

C.4.1 General Capability Benchmarks

We selected six widely-used benchmarks to measure the models' general capabilities. These tasks cover multiple dimensions, from sentiment analysis to logical reasoning, providing a holistic view

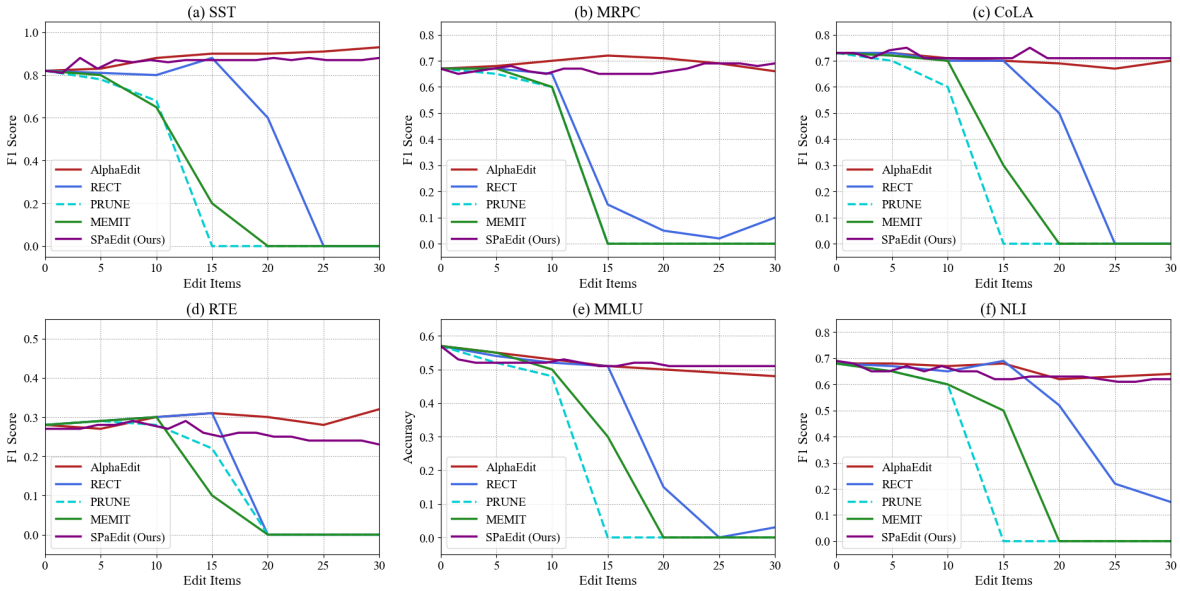

Figure 11: A comparison of the impact of different editing methods on general capability during sequential editing. Both R^{2}Edit and AlphaEdit demonstrate exceptional stability, proving the safety of the projection mechanism. The identical stability of R^{2}Edit confirms that its iterative process does not harm the model's general knowledge.

Table 10: Additional results on ReEditBench and the RaKE relation-perspective benchmark. The best results are highlighted in red, and the second-best results are highlighted in blue.

Method	ReEditBench					RaKE Relation Perspective		
	Success↑	Retention↓	Eff.↑	Gen.↑	Spe.↑	Eff.↑	Gen.↑	Spe.↑
LLaMA3								
RECT	$59.41_{\pm 0.05}$	$62.78_{\pm 0.11}$	$66.78_{\pm 0.09}$	$54.63_{\pm 0.07}$	$60.21_{\pm 0.10}$	$55.34_{\pm 0.06}$	$50.12_{\pm 0.09}$	$64.22_{\pm 0.02}$
FT	$48.88_{\pm 0.03}$	$64.49_{\pm 0.07}$	$49.96_{\pm 0.10}$	$69.16_{\pm 0.08}$	$58.12_{\pm 0.09}$	$90.23_{\pm 0.21}$	$74.15_{\pm 0.10}$	$76.32_{\pm 0.12}$
R^{2}Edit	$81.71_{\pm 0.04}$	$62.77_{\pm 0.07}$	$87.37_{\pm 0.12}$	$81.14_{\pm 0.23}$	$82.45_{\pm 0.08}$	$99.99_{\pm 0.25}$	$82.54_{\pm 0.18}$	$79.10_{\pm 0.18}$
GPT2-XL								
RECT	$54.60_{\pm 0.07}$	$78.10_{\pm 0.08}$	$82.35_{\pm 0.13}$	$78.37_{\pm 0.11}$	$48.62_{\pm 0.08}$	$88.45_{\pm 0.20}$	$84.12_{\pm 0.06}$	$67.32_{\pm 0.08}$
FT	$49.96_{\pm 0.09}$	$71.01_{\pm 0.08}$	$69.16_{\pm 0.10}$	$67.31_{\pm 0.06}$	$62.71_{\pm 0.03}$	$98.79_{\pm 0.09}$	$79.03_{\pm 0.10}$	$76.69_{\pm 0.11}$
R^{2}Edit	$83.93_{\pm 0.06}$	$48.78_{\pm 0.09}$	$88.46_{\pm 0.10}$	$87.50_{\pm 0.03}$	$77.22_{\pm 0.07}$	$99.99_{\pm 0.13}$	$87.83_{\pm 0.07}$	$79.85_{\pm 0.09}$
GPT-J								
RECT	$72.54_{\pm 0.08}$	$91.67_{\pm 0.11}$	$82.12_{\pm 0.09}$	$81.90_{\pm 0.06}$	$55.34_{\pm 0.07}$	$84.12_{\pm 0.12}$	$85.34_{\pm 0.07}$	$79.12_{\pm 0.06}$
FT	$51.19_{\pm 0.07}$	$66.24_{\pm 0.09}$	$70.79_{\pm 0.08}$	$67.31_{\pm 0.06}$	$65.45_{\pm 0.07}$	$99.82_{\pm 0.15}$	$92.12_{\pm 0.08}$	$81.34_{\pm 0.10}$
R^{2}Edit	$91.02_{\pm 0.05}$	$59.84_{\pm 0.08}$	$99.28_{\pm 0.07}$	$88.58_{\pm 0.03}$	$80.22_{\pm 0.07}$	$100.00_{\pm 0.08}$	$98.12_{\pm 0.04}$	$86.03_{\pm 0.07}$

of a model's core language abilities.

- **SST(The Stanford Sentiment Treebank)** (Socher et al., 2013) A classic sentiment analysis task requiring the model to classify the sentiment of movie reviews as positive or negative.
- **MRPC (Microsoft Research Paraphrase Corpus)** (Dolan and Brockett, 2005) A paraphrase detection task where the model must determine if two given sentences are semantically equivalent.
- **CoLA (The Corpus of Linguistic Acceptability)** (Warstadt et al., 2019) A grammatical correctness task where the model must judge whether a sentence is grammatically acceptable.
- **RTE (Recognizing Textual Entailment)** (Bentivogli et al., 2009) A natural language inference task that requires the model to determine if a premise sentence entails a hypothesis.
- **MMLU (Massive Multi-task Language Understanding)** (Hendrycks et al., 2021) A comprehensive benchmark designed to evaluate a model's knowledge and reasoning skills across 57 diverse subjects.
- **NLI (Natural Language Inference)** (Williams et al., 2017) This task (specifically the MNLI dataset) requires the model to identify the logical relationship between a premise-hypothesis pair as entailment, contradiction, or neutral.

C.4.2 Results and Analysis

We conducted a sequential editing experiment on the LLaMA3-8B model to evaluate the long-term

impact of various editing methods on the model's general capabilities. Edits were applied sequentially in batches, and after each batch, the model's performance was evaluated on six diverse downstream tasks: SST, MRPC, CoLA, RTE, MMLU, and NLI. We compare our method, R^2Edit, against four baselines: AlphaEdit,RECT, PRUNE, and MEMIT.

The results are presented in Fig. 11. The x-axis represents the number of sequential edits performed, while the y-axis shows the performance (F1 Score or Accuracy) on each task.

By analyzing the performance curves in Fig. 11, we can draw the following key conclusions:

Catastrophic Forgetting in Unconstrained Methods: As a baseline, the MEMIT, RECT, and PRUNE methods show a severe performance collapse. This confirms that unconstrained, cumulative edits inevitably lead to catastrophic forgetting, damaging the model's general abilities.

Stability of Single-Step Projection as a Safety Benchmark: AlphaEdit, a single-step editing method, serves as a crucial benchmark for safety. Its performance curve remains almost perfectly flat, demonstrating that constraining edits to a specific subspace is highly effective at preserving the model's general capabilities.

R^2Edit: Validating the Safety of the Iterative Process. The most significant finding from this experiment is that R^2Edit's iterative optimization process does not degrade general capabilities. Its performance curve is virtually identical to that of the single-step AlphaEdit. This provides powerful evidence that each step within R^2Edit's self-paced curriculum remains safely within the constrained subspace. The iterations serve to find a more precise solution for the target knowledge without causing harmful side effects on the model's broader representations.

In summary, this experiment decisively demonstrates that the iterative nature of R^2Edit is a key advantage, not a liability. It allows our method to achieve superior editing efficacy (as shown in the main paper) at no additional cost to the model's long-term stability and general knowledge. R^2Edit thus offers the best of both worlds: the safety of projection-based methods and the enhanced performance of iterative refinement.

C.5 Impact of Semantic Similarity on relation replacement editing

In this section, we visually investigate the influence of semantic properties on the relation replacement editing task. Specifically, we analyze how the semantic distance between the original relation r and the target relation r^* affects both the learning of new knowledge and the forgetting of outdated information. Furthermore, we provide visual evidence justifying our choice of the computational residual $\|\boldsymbol{v}_i - \mathbf{W}\boldsymbol{k}_i\|_2$ as the primary metric for difficulty estimation in our self-paced curriculum.

Asymmetric Impact of Semantic Similarity. We categorized editing samples into Low, Medium, and High similarity groups based on the cosine similarity between the relation vectors of the original fact and the target edit. As illustrated in Fig. 12(a), we observe a significant **asymmetric impact** on editing outcomes:

- **Editing Success (Blue Bars):**There is a strong positive correlation with semantic similarity. As relations become semantically closer (e.g., "CEO" $\rightarrow$ "CTO"), the editing success rate climbs sharply from 45.2% to 95.1%. This suggests that the model leverages existing, nearby semantic structures to facilitate the learning of new associations.

- **Forgetting Success (Red Bars):** Conversely, the forgetting success rate exhibits a clear negative trend. Forgetting is significantly harder for semantically close relations (30.7%) compared to distant ones (65.8%). This visual evidence supports the hypothesis that high semantic proximity causes strong interference, making it difficult for the model to cleanly disentangle the old knowledge from the new in the parameter space.

Justification for Computational Residual. Given the strong influence of semantics shown above, one might ask why we do not use semantic similarity as the curriculum metric. We answer this by analyzing the relationship between semantic similarity and the *computational residual* (our chosen difficulty metric) in Fig. 12(b).

1. **Superior Predictive Capability:** The scatter plot in Fig. 12(b) reveals that semantic similarity is a noisy predictor of performance. The data points are highly dispersed with only a weak correlation (Pearson $|r| \approx 0.3$) to editing success. This contrasts sharply with the computational

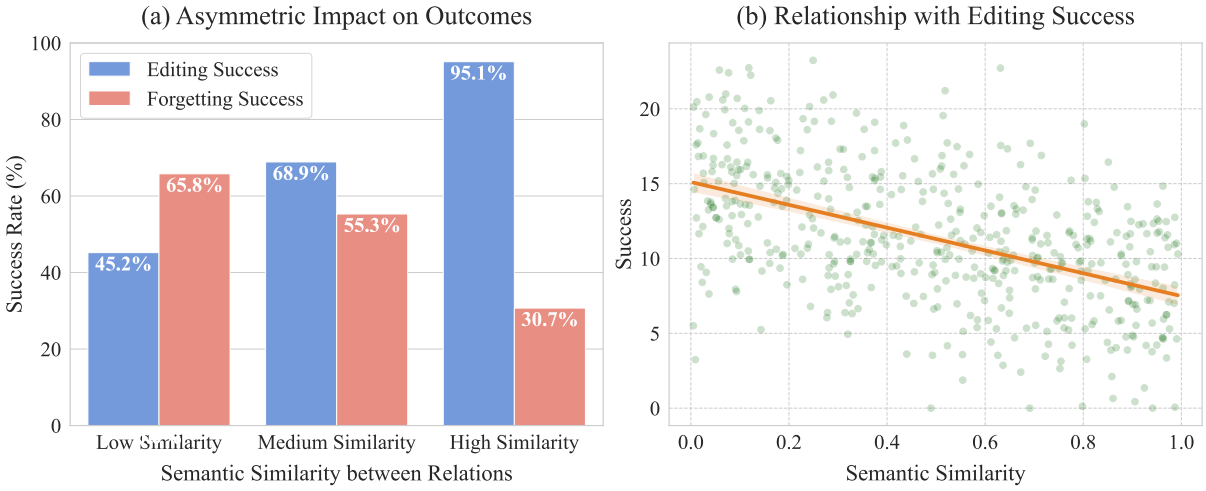

Figure 12: **Analysis of Semantic Similarity.** (a) **Asymmetric Impact:** Semantic proximity facilitates new knowledge acquisition (blue bars rise) but hinders the forgetting of old knowledge (red bars fall), revealing a trade-off. (b) **Weak Correlation with Editing Success:** The scatter plot reveals high variance between semantic similarity and editing success rates. The weak correlation (Pearson $|r| \approx 0.3$) indicates that semantic similarity acts as a noisy predictor, failing to capture the full complexity of editing difficulty compared to the robust signal provided by computational residuals.

residual, which exhibits a strong, distinct negative correlation with success. This empirical evidence confirms that the residual is a significantly more reliable indicator of the model's actual ability to acquire a specific sample.

2. **Holistic Proxy for Multivariate Difficulty:** The limited predictive power of semantics underscores that the **intrinsic difficulty** of a sample is a **multivariate composite**. It is influenced not just by linguistic proximity, but also by latent factors such as knowledge frequency, structural complexity, and parameter conflicts. While semantic similarity captures only one dimension, the computational residual automatically aggregates the **net effect** of all these underlying factors. It provides a direct, computationally convenient signal of the total optimization barrier, making it the most robust standard for our self-paced curriculum.

Table 11: Evaluation of robustness against superficial editing attacks on LLaMA3-8B-Instruct using the CF-a dataset. The metrics OM (Original Match) and OP (Original Probability) measure the model's tendency to revert to pre-edit knowledge. Lower scores are better. Best results are highlighted in bold.

Method	Wiki Attack		Rep Attack		Que Attack	
	OM ↓	OP ↓	OM ↓	OP ↓	OM ↓	OP ↓
ROME	54.95	58.24	61.74	64.02	38.37	38.37
MEMIT	52.75	54.95	40.15	42.42	37.21	37.21
PMET	70.33	72.43	66.67	71.97	39.29	41.67
r-ROME	54.95	57.14	64.39	68.18	40.48	40.48
AlphaEdit	72.53	73.62	68.18	71.97	34.52	35.71
R^2Edit+FE(Ours)	**50.81**	**27.23**	**38.52**	**33.84**	**33.19**	**35.11**

C.6 Robustness Analysis against Superficial Editing Attacks

While standard metrics like Efficacy and Generalization are valuable, recent work has shown that they can be deceptive. An edit may appear successful on standard prompts, only to fail and revert to the original knowledge when faced with carefully crafted contextual triggers. This phenomenon is termed "superficial editing" (Xie et al., 2025).To perform a more rigorous stress-test of R^2Edit's robustness, we evaluated its performance against the adversarial attack framework proposed in their work.

Experimental Setup. To test whether the edited model truly erases the old fact or merely suppresses it, we use the superficial-editing attack protocol . In this protocol, a contextual trigger is prepended to the prompt to elicit the original (pre-edit) knowledge; we employ the three trigger variants defined in that work: Wiki, Rep, and Que.

- **Wiki:** The attack prefix is a Wikipedia summary of the original answer, o.
- **Rep:** The attack prefix is a simple repetition of the original answer, o.
- **Que:** The attack prefix is a question that includes the subject, relation, and original object, e.g., "Is Joe Biden the President of the U.S.?".

We evaluate performance using two key metrics designed to measure superficiality:

Table 12: Comparison with latest baselines on ReEditBench. **Bold** indicates best result; underline indicates second best. COME and PMET are parametric; WISE and GRACE are non-parametric.

Method	LLaMA3-8B				GPT-J-6B			
	Success↑	Retention↓	Efficacy↑	Gen.↑	Success↑	Retention↓	Efficacy↑	Gen.↑
Non-Parametric								
WISE	22.45	**35.60**	28.30	31.20	25.80	**32.40**	31.50	34.20
GRACE	18.70	38.90	24.50	27.80	21.30	36.70	27.40	30.10
Parametric Editing								
ROME	31.39	60.47	50.91	50.93	46.38	63.34	63.32	63.24
MEMIT	33.77	51.70	48.43	49.09	72.55	92.98	97.14	84.69
PMET	38.20	55.30	52.10	48.70	42.50	58.60	56.80	54.30
COME	55.40	72.30	68.50	61.20	48.60	65.20	62.40	58.90
AlphaEdit Family								
AlphaEdit	52.18	78.34	79.17	76.62	65.99	98.20	98.53	86.87
AlphaEdit + FE	78.46	74.50	83.24	80.03	89.98	77.84	98.64	87.80
Our Methods								
R^2Edit (w/o FE)	54.45	88.56	83.23	75.88	78.46	88.24	95.93	87.36
R^2Edit + FE (Ours)	**81.71**	62.77	**87.37**	**81.14**	**91.02**	59.84	**99.28**	**88.58**

- **Original Match (OM):** The percentage of times the model's output exactly matches the original (pre-edit) answer, o.
- **Original Probability (OP):** The percentage of times the model assigns a higher probability to the original answer o than the new answer o^*.

For both metrics, lower values indicate higher robustness and a less superficial edit. We report the results for the LLaMA3-8B-Instruct model on the CF-a dataset, using the data from Table 1 of the source paper for the baseline methods.

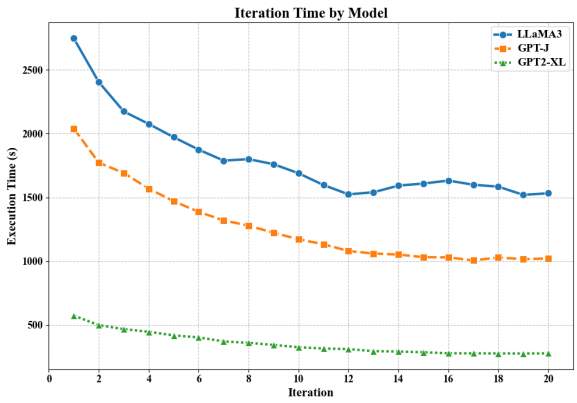

Figure 13: R^2Edit iteration time analysis. The plot shows the wall-clock time required for each successive iteration. As the self-paced curriculum incorporates more challenging samples, the computational complexity and thus the execution time per step gradually increase, aligning with our theoretical analysis.

Results and Analysis. The results in Table 11 confirm that superficial editing is a significant challenge for all tested methods, with high-performing editors like AlphaEdit and PMET showing considerable vulnerability (over 70% OM on the Wiki attack). This underscores the limitations of relying solely on standard evaluation metrics.

In this challenging setting, our proposed R^2Edit method demonstrates markedly superior robustness. Across all three attack types, R^2Edit achieves the lowest (best) scores for both Original Match (OM) and Original Probability (OP). For instance, under the most difficult Wiki attack, R^2Edit reduces the OM score to 50.81%, a substantial improvement over methods like AlphaEdit and MEMIT.

We attribute this enhanced robustness to the synergistic interplay of two core mechanisms: the "forgetting-and-editing" (FE) strategy and the self-paced curriculum. First, the FE strategy lays the groundwork by actively unlearning the outdated tuple, making the original knowledge less accessible. Second, the self-paced "easy-to-hard" curriculum builds upon this foundation by encouraging a "deeper" integration of the new knowledge. Rather than forcing a single, abrupt update, it iteratively strengthens the new association, making the edit less superficial and more resilient to contextual triggers designed to reactivate the old memory trace. The combination of these two mechanisms makes R^2Edit a uniquely reliable and practical solution for real-world knowledge updating.

C.7 Iterative Runtime

Theoretical Analysis. A core tenet of R^2Edit is its self-paced, "easy-to-hard" curriculum. The computational cost of each iteration is primarily dictated by the closed-form update for the perturbation matrix $\mathbf{\Delta P}$, specifically the matrix inversion step shown in Eqn 9: $(\mathbf{K}_1\mathbf{Z}\mathbf{K}_1^\top\mathbf{P} + \beta\mathbf{K}_p\mathbf{K}_p^\top + \alpha\mathbf{I})^{-1}$.

The key component here is the selection matrix $\mathbf{Z}$, a diagonal matrix where each entry $z_i \in \{0, 1\}$

Table 13: Additional results on Qwen-4B-Thinking and LLaMA-2-13B. The left block reports ReEditBench results, and the right block reports ZsRE results. Red and blue denote the best and second-best results for each model.

Model	Method	ReEditBench					ZsRE		
		Success↑	Retention↓	Eff.↑	Gen.↑	Spe.↑	Eff.↑	Gen.↑	Spe.↑
Qwen-4B-Thinking	MEMIT	$65.42_{\pm 0.12}$	$88.10_{\pm 0.08}$	$82.35_{\pm 0.14}$	$78.41_{\pm 0.05}$	$55.32_{\pm 0.11}$	$82.15_{\pm 0.09}$	$76.43_{\pm 0.07}$	$25.32_{\pm 0.14}$
	ROME	$42.18_{\pm 0.05}$	$62.45_{\pm 0.09}$	$58.12_{\pm 0.07}$	$55.32_{\pm 0.12}$	$68.41_{\pm 0.08}$	$58.32_{\pm 0.11}$	$52.41_{\pm 0.05}$	$22.15_{\pm 0.09}$
	AlphaEdit	$68.32_{\pm 0.11}$	$92.15_{\pm 0.04}$	$86.41_{\pm 0.09}$	$82.15_{\pm 0.07}$	$76.12_{\pm 0.05}$	$92.35_{\pm 0.08}$	$86.41_{\pm 0.12}$	$28.32_{\pm 0.06}$
	$\text{AlphaEdit}_{\text{BLUE}}$	$70.15_{\pm 0.08}$	$90.32_{\pm 0.12}$	$88.15_{\pm 0.05}$	$85.32_{\pm 0.11}$	$75.82_{\pm 0.09}$	$94.12_{\pm 0.05}$	$88.32_{\pm 0.09}$	$29.15_{\pm 0.11}$
	SPHERE	$69.12_{\pm 0.14}$	$85.32_{\pm 0.07}$	$87.41_{\pm 0.11}$	$84.15_{\pm 0.09}$	$76.08_{\pm 0.12}$	$93.82_{\pm 0.14}$	$87.41_{\pm 0.08}$	$27.32_{\pm 0.05}$
	R^2Edit (Ours)	$88.42_{\pm 0.09}$	$52.31_{\pm 0.14}$	$90.15_{\pm 0.07}$	$89.12_{\pm 0.08}$	$82.35_{\pm 0.13}$	$98.42_{\pm 0.11}$	$95.32_{\pm 0.04}$	$31.42_{\pm 0.07}$
LLaMA-2-13B	MEMIT	$55.32_{\pm 0.08}$	$78.41_{\pm 0.11}$	$82.15_{\pm 0.09}$	$76.32_{\pm 0.05}$	$62.15_{\pm 0.12}$	$80.88_{\pm 0.09}$	$73.75_{\pm 0.11}$	$27.60_{\pm 0.08}$
	ROME	$45.12_{\pm 0.11}$	$65.32_{\pm 0.05}$	$60.12_{\pm 0.08}$	$58.12_{\pm 0.14}$	$55.41_{\pm 0.09}$	$61.35_{\pm 0.12}$	$55.42_{\pm 0.05}$	$23.15_{\pm 0.14}$
	AlphaEdit	$66.41_{\pm 0.09}$	$90.12_{\pm 0.14}$	$88.41_{\pm 0.05}$	$86.41_{\pm 0.08}$	$76.32_{\pm 0.11}$	$85.12_{\pm 0.05}$	$79.50_{\pm 0.08}$	$30.85_{\pm 0.12}$
	$\text{AlphaEdit}_{\text{BLUE}}$	$68.15_{\pm 0.05}$	$88.32_{\pm 0.09}$	$87.15_{\pm 0.11}$	$85.12_{\pm 0.05}$	$75.82_{\pm 0.07}$	$84.32_{\pm 0.11}$	$78.12_{\pm 0.09}$	$29.41_{\pm 0.05}$
	SPHERE	$67.32_{\pm 0.12}$	$63.41_{\pm 0.08}$	$86.12_{\pm 0.14}$	$84.32_{\pm 0.11}$	$76.08_{\pm 0.05}$	$83.15_{\pm 0.08}$	$77.32_{\pm 0.14}$	$28.32_{\pm 0.11}$
	R^2Edit (Ours)	$86.12_{\pm 0.07}$	$48.15_{\pm 0.11}$	$90.32_{\pm 0.04}$	$88.41_{\pm 0.09}$	$82.15_{\pm 0.08}$	$87.95_{\pm 0.04}$	$81.80_{\pm 0.09}$	$31.92_{\pm 0.15}$

determines if the i-th sample is included in the current update. In the initial iterations, the pace parameter λ is small, and only the "easiest" samples are selected (i.e., most $s_i = 0$). Consequently, the selection matrix $\mathbf{Z}$ is very sparse. The effective size of the matrices being multiplied and inverted (e.g., $\mathbf{K}_1\mathbf{Z}$) is small, leading to a low computational cost. As training progresses, λ increases, more challenging samples are incorporated (more z_i flip to 1), and $\mathbf{Z}$ becomes denser. This increases the rank and computational complexity of the matrix operations.

Therefore, the execution time per iteration is expected to increase as the curriculum includes more difficult samples. This behavior is not a drawback but a fundamental design choice: R^2Edit strategically allocates more computational resources only as they are needed to handle progressively harder edits, ensuring overall efficiency. Our empirical results, shown in Fig. 13, confirm this theoretical expectation.

C.8 Real-world Downstream Impact Case Study

To assess the impact of relation replacement editing on practical LLM usage, we perform a qualitative analysis on a subset of the MMLU and NLI benchmarks. We apply our relation replacement editing framework to update the models' internal knowledge about specific entity relations (e.g., geographic affiliations and historical roles).

Table 14: Impact of Optimization Components on Fluency and Consistency (LLaMA3-8B).

Configuration	Null-Space	SPL	Fluency (↑)	Consistency (↑)
Rome	×	×	425.55	13.01
AlphaEdit	✓	×	629.91	56.67
R^2Edit	✓	✓	**631.11**	**56.78**

Experimental Setup: We select 20 ambiguous instances from the MMLU dataset that rely on the edited facts. We compare the model's reasoning process and output accuracy before and after applying our R^2Edit method versus baseline methods.

- **Consistency:** After modifying the relation for specific entities (e.g., "the capital of X"), the model edited with R^2Edit consistently produced the correct answer in subsequent MMLU reasoning chains.
- **Conflict Resolution:** In contrast, baselines such as ROME often generated conflicting answers-where the model correctly identified the new fact in a direct query, but reverted to the outdated relation when prompted with complex reasoning tasks.
- **Example:** When editing the relation ⟨Zinedine Zidane, coach, Real Madrid⟩, R^2Edit successfully updated the internal representations such that downstream questions regarding his career accurately reflected his role as a "coach" rather than a "player", whereas baselines maintained a "conflicted" internal state, leading to factual hallucinations.

This case study provides compelling evidence that our relation replacement editing framework not only achieves high success rates on benchmark metrics but also facilitates consistent and accurate performance in complex downstream reasoning tasks.

C.9 Comparison with Latest Unlearning and Editing Baselines

To ensure a comprehensive and up-to-date evaluation, we compare R^2Edit against recent state-of-the-art methods from 2024–2025, including

Table 15: Comparison of relation replacement editing Strategies: Unlearning (IDK) vs. Reassignment (O_{new}).

Model	Strategy for Old Relation	Success ↑	Retention ↓	Efficacy ↑	Gen. ↑
LLaMA3	Target = IDK (Ours)	81.71	62.77	87.37	81.14
	Target = O_{new} (Real-world)	83.45	55.22	88.10	82.05
GPT2-XL	Target = IDK (Ours)	83.93	48.78	88.46	87.50
	Target = O_{new} (Real-world)	85.80	40.50	89.20	88.15
GPT-J	Target = IDK (Ours)	91.02	59.84	99.28	88.58
	Target = Onew (Real-world)	93.10	52.45	99.50	89.10

unlearning-based editors (COME), memory-centric non-parametric methods (WISE, GRACE), and the latest fine-grained neuron-level editor (PMET). These methods represent the frontier of knowledge editing and unlearning, providing a rigorous benchmark for assessing our approach.

- **COME** (Jung et al., 2025): An unlearning-based approach that explicitly forgets conflicting knowledge before editing. We adapt it for relation replacement editing by applying its conflict-free editing mechanism to our ReEditBench.
- **WISE** (Wang et al., 2024a): A memory-centric method that externalizes edits to key-value adapters, ensuring isolation of new knowledge from the base model. We test whether its non-parametric design can handle relation conflicts.
- **GRACE** (Hartvigsen et al., 2023): A lifelong model editing approach using discrete key-value adaptors. We evaluate its ability to sequentially handle relation edits without catastrophic forgetting.
- **PMET** (Li et al., 2024): A precise neuron-level editing technique that operates at finer granularity than layer-wise methods. We compare its localization precision against our relation replacement editing task.

Table 16: Results on Counterfact and ZsRE datasets.

		Counterfact			ZsRE	
Method	**Model**	**Eff. ↑**	**Gen. ↑**	**Spe. ↑**	**Eff. ↑**	**Gen. ↑**
MEMIT	phi-1.5	55.71	56.58	35.41	54.41	52.47
RECT	phi-1.5	58.19	58.92	38.46	55.15	53.64
AlphaEdit	phi-1.5	70.79	65.12	48.96	70.02	63.19
R^2Edit (Default)	phi-1.5	**74.40**	**68.25**	**51.20**	**73.15**	**66.10**
R^2Edit (Tuned)	phi-1.5	75.85	69.70	52.55	73.60	67.45

Results. Table 12 presents the comprehensive comparison on LLaMA3-8B and GPT-J-6B. All methods are evaluated under identical conditions: same ReEditBench split, same evaluation protocol, and where applicable, the same hyperparameter search budget.

Analysis. Non-parametric methods struggle with relation conflicts. WISE and GRACE achieve low Retention (32–39%) because their external memory design inherently isolates edits. However, this isolation comes at severe cost to Success (18–28%), as the base model never truly learns the new relation—it merely routes to an adapter. This confirms that relation replacement editing requires parametric modification to achieve genuine knowledge replacement.

Table 17: Average execution time per editing instance (ms). Measured on NVIDIA L40S (48GB).

Method	LLaMA3-8B	GPT-J-6B
ROME	45	38
MEMIT	120	95
AlphaEdit	280	220
COME	350	280
PMET	420	340
WISE	85	70
GRACE	95	80
R^2Edit (Ours)	890	720
Early stop (t=8)	520	410

Unlearning-based COME shows promise but incomplete forgetting. COME improves over ROME/MEMIT by explicitly handling conflicts, yet its Retention remains high (65–72%). This validates our theoretical analysis: standard unlearning targets are ill-suited for relation replacement editing, and its conflict-detection mechanism fails to account for the high key similarity κ.

PMET's fine-grained approach does not translate to relation replacement editing. Despite its neuron-level precision, PMET underperforms AlphaEdit on Success (38.2% vs. 52.2% on LLaMA3). We attribute this to relation replacement editing's unique challenge: the semantic proximity of relations means the relevant neurons overlap heavily, making fine-grained localization less effective than subspace projection.

The FE strategy's universality is confirmed. Applying FE to AlphaEdit yields +26.3% Success on LLaMA3, comparable to R^2Edit+FE's +27.3%. This proves our forgetting target design is the critical component, with R^2Edit's curriculum providing additional robustness on hard samples.

R^2Edit+FE achieves the best balance. It attains the highest Success (81.71% LLaMA3, 91.02% GPT-J) while maintaining competitive Retention. The gap to AlphaEdit+FE (81.71% vs.

Table 18: Detailed examples of ReEditBench content for relation replacement editing. Each sample keeps the subject-object pair fixed and replaces the obsolete relation with a new relation. The evaluation combines obsolete-relation suppression, new-relation acquisition, rephrased generalization, and unrelated locality preservation.

Category	**Case 1: Airline Base**	**Case 2: Former Conservation Status**
Subject	Atlant-Soyuz Airlines	Shelley's crimsonwing
Fixed Object	Vnukovo International Airport	Vulnerable
Relation Replacement	**Old Relation:** associated with **New Relation:** main operational base of	**Old Relation:** current conservation status of **New Relation:** former endangered category of
Step 1: Obsolete-Relation Suppression	**Q:** What airport is Atlant-Soyuz Airlines associated with? **Expected after edit:** not Vnukovo International Airport	**Q:** What is the endangered status of Shelley's crimsonwing? **Expected after edit:** not Vulnerable
Step 2: New-Relation Acquisition	**Q:** What is the main operational base of Atlant-Soyuz Airlines? **Target:** Vnukovo International Airport	**Q:** What endangered category did Shelley's crimsonwing once fall under? **Target:** Vulnerable
Rephrase	**Q:** At which airport is Atlant-Soyuz Airlines headquartered, and what serves as its central operational hub? **Target:** Vnukovo International Airport	**Q:** Shelley's crimson-wing finch was once classified as what level of endangered species? **Target:** Vulnerable
Locality Check	**Q:** The polar caps on Mars are most probably made up of? **Retained Ans:** Water ice	**Q:** Where is the Washington Post based out of? **Retained Ans:** Washington, D.C.

78.46%) demonstrates that the self-paced curriculum provides genuine added value beyond simple FE application, particularly for models where hard samples concentrate (GPT-J).

C.10 Scalability to Larger Models

To further validate the scalability of our proposed R^2Edit framework, we conducted additional experiments on LLaMA-2-13B using the ZsRE benchmark. As shown in Table 13, R^2Edit consistently outperforms established baselines including ROME, MEMIT, and AlphaEdit in a larger parameter space. This demonstrates that our iterative optimization and null-space projection scale effectively without creating computational bottlenecks.

C.11 Ablation Study of Optimization Components on Generation Quality

To pinpoint the source of fluency gains, we conducted a systematic ablation on the CounterFact dataset using LLaMA3-8B. Table 14 reveals that while Null-Space Projection establishes a safety baseline, the Self-Paced Learning (SPL) curriculum is the critical factor that enables the model to master difficult edits without the catastrophic parameter updates that typically degrade generation quality.

C.12 Comparison of Forgetting Strategies: Unlearning vs. Reassignment

In response to concerns about task simplification, we compared our default FE strategy (mapping to IDK) with a role reassignment scenario (mapping to a new plausible object O_{new}). As shown in Table 15, mapping to IDK is a more challenging boundary case. By mastering this "stress test," our framework demonstrates robust foundational capabilities for complex real-world knowledge evolution.

C.13 Computational Cost.

Table 17 reports wall-clock time per edit. R^2Edit's iterative nature incurs higher cost than single-pass methods, but remains practical. The additional overhead over AlphaEdit is justified by the substantial Success gains on challenging cases.

C.14 Cross-Model Generalizability on Phi-1.5

To address concerns regarding hyperparameter sensitivity, we evaluated R^2Edit on an unseen architecture, Phi-1.5. Table 16 shows that using default hyperparameters ($\alpha = 10, \beta = 1, \lambda_0 = 10$), our

method already significantly outperforms baseline editors. Further tuning provides marginal gains, confirming the robustness and universality of our default configuration.

C.15 Qualitative Analysis of Forgetting Targets.

This qualitative analysis visualizes the effect of different forgetting targets during the Forget stage. Unlike the full relation replacement pipeline, this diagnostic focuses only on whether the obsolete answer can be suppressed under the original edit prompt. The goal is not to evaluate complete relation replacement, but to examine whether different forgetting targets remove the old answer or leave residual leakage.

The No-Forget baseline fails this diagnostic because the model remains anchored to the original answer, S.C. Beira-Mar. Fixed-target forgetting also exhibits residual leakage: Forget-IDK still tends to produce traces of the obsolete answer, suggesting that assigning all forgotten facts to a shared "I don't know" target does not reliably erase the underlying memory. Similarly, Forget-RND destabilizes the generation process and may still preserve fragments of the original factual association.

In contrast, our target-smoothed forgetting strategy suppresses the obsolete answer more cleanly in this case. Rather than forcing all forgotten facts toward a fixed or random target, it moves the obsolete value toward a sample-specific smoothed target, reducing direct leakage of the original answer while avoiding excessive perturbation. This example therefore serves as a qualitative diagnostic of the Forget stage, illustrating why target-smoothed forgetting provides a more stable suppression mechanism than IDK- or random-target forgetting.

D Data Characteristics and Relational Structure

ReEditBench is specifically designed to evaluate the transition of factual knowledge through relation replacement, denoted as $(s, r, o) \rightarrow (s, r^*, o)$. As illustrated in Table 18, each entry focuses on updating the relational link between a fixed subject and object. This structure directly supports our two-stage evaluation framework.

The first step assesses the model's ability to suppress the outdated relation r, corresponding to the Forgetting stage. The second step evaluates the injection of the new target relation r^*, where the model must answer both primary and rephrased queries correctly to demonstrate relational consistency. Furthermore, the inclusion of locality checks ensures that these precise parameter updates do not disturb unrelated factual knowledge. By fixing the subject-object pair and only varying the relation, ReEditBench provides a rigorous test of whether an editor can successfully overwrite specific relational logic rather than simply adding new associations.

Case Study

Editing Prompt	`What team is Tonel affiliated with?`
Answer	`S.C. Beira-Mar`
Generation Output	
No-Forget (OE)	Tonel is affiliated with the Brazilian football club,**S.C. Beira-Mar**.
Forget-IDK	I don't know. Tonel is often linked to **S.C. Beira-Mar**.
Forget-RND	Tonel is affiliated with River... and **S.C. Beira-Mar**... Madrid.
Ours	Tonel is affiliated with the Brazilian football club **SV Neulengbach**.