MANIEDIT: SEQUENTIAL UNSTRUCTURED KNOWLEDGE EDITING FOR LANGUAGE MODELS FROM A MANIFOLD PERSPECTIVE

Anonymous authors
Paper under double-blind review

ABSTRACT

Large language models (LLMs) inevitably generate some incorrect or outdated content, necessitating efficient and precise mechanisms for continual knowledge updates. However, existing model editing methods struggle to sequentially edit unstructured long-form knowledge, suffering from severe edit forgetting and degradation of general capabilities. To address these challenges, we reframe knowledge editing from a manifold perspective, viewing it as a localized displacement of an edit sub-manifold within the global knowledge manifold. Under this formulation, the problem can be decomposed into two key questions: (i) how to identify representative edit points that effectively anchor the edit sub-manifold, and (ii) how to preserve the remaining manifold structure during the sub-manifold displacement process. Based on this perspective, we propose **ManiEdit**, a novel manifold-aware autoregressive editing framework consisting of two core components. *Pivot Localization* addresses the mediocre-point dilemma by identifying high-leverage pivots to anchor the edit sub-manifold. *Manifold-Aware Preservation* preserves different knowledge types through an energy-weighted penalty combined with recursive null-space alignment. Experiments on two base LLMs and four unstructured editing benchmarks demonstrate that ManiEdit achieves state-of-the-art performance, outperforming the strongest baseline by up to $+27.81$ BERTScore and $+8.50$ ROUGE-L, while maintaining near-original general capabilities across six representative downstream tasks.

1 INTRODUCTION

Large Language Models (LLMs) have demonstrated impressive capabilities in storing and recalling vast amounts of knowledge within their parameters (Brown et al., 2020; Petroni et al., 2019; Roberts et al., 2020; Zhao et al., 2026). However, stored facts often become inaccurate or outdated due to noisy training data and temporal drift (De Cao et al., 2021; Mitchell et al., 2022a). While fine-tuning on updated facts offers a straightforward remedy, it is resource-intensive and risks both overfitting to individual examples and catastrophic forgetting (Mitchell et al., 2022b; Meng et al., 2022). To overcome these limitations, *knowledge editing* has emerged as a promising alternative that enables precise, cost-effective modification of specific target knowledge while preserving other knowledge.

Current knowledge editing approaches roughly fall into two categories: (i) *parameter-modifying* methods that directly adjust a small subset of model parameters (Meng et al., 2023; Fang et al., 2025; Jiang et al., 2025), and (ii) *parameter-preserving* methods that introduce additional external modules without updating the original weights (Mitchell et al., 2022b; Hartvigsen et al., 2023; Yu et al., 2024). Among the former, the dominant locate-then-edit paradigm identifies the weights storing target knowledge via causal tracing and perturbs them with a closed-form update. Representative methods such as ROME (Meng et al., 2022) and MEMIT (Meng et al., 2023) interpret the down-projection matrix of each feed-forward layer as a linear associative memory and derive closed-form updates for atomic subject-predicate-object triplets.

Despite their success on structured triplets, realistic deployment still faces two fundamental challenges. First, roughly 80% of real-world knowledge is *unstructured* (Bavota, 2016; Deng et al., 2025), expressed in diverse formats (e.g., poetry, code, mathematical derivation) with rich context

Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping the LLM and submitting your original text: we, and the authors, are much more interested in your unedited thoughts than in what an LLM has to say. AI-assisted or not, you are putting your name and reputation behind your review: LLM-generated falsehoods, hallucinations or misrepresentations are subject to disciplinary action, which may include desk-rejecting all papers you have authored.

dependencies rather than discrete triplets, making it difficult to cleanly locate, encode, and inject target knowledge into the model. Second, knowledge updates accumulate *sequentially* over time rather than arriving in a single batch, requiring continual editing without disrupting prior updates or pre-trained capabilities. This sequential setting gives rise to two critical failure modes: *edit forgetting*, where each new edit partially erases previously updated knowledge, causing editing performance to decay as the sequence grows; and *general-capability collapse*, where accumulated parameter drift destroys the model's linguistic and reasoning abilities within only a few hundred edits (Figure 1).

To solve these challenges, we reframe knowledge editing from a manifold perspective, viewing it as a localized displacement of an edit sub-manifold within the global knowledge manifold. Under this formulation, the problem can be decomposed into two key questions: (i) how to identify representative edit points that effectively anchor the edit sub-manifold, and (ii) how to preserve the remaining manifold structure during the sub-manifold displacement process, thereby achieving precise editing while preserving historical edits and general capabilities.

From this perspective, we propose **ManiEdit**, a novel manifold-aware autoregressive editing paradigm designed for sequential unstructured knowledge editing, which comprises two tightly coupled components that address the key questions above. (i) **Pivot Localization** identifies *semantic pivots*—tokens with high leverage scores on the manifold periphery—and then anchors the edit sub-manifold at these pivots, improving the effectiveness of each weight update. (ii) **Manifold-Aware Preservation** decouples the preservation of two types of knowledge according to their distinct manifold characteristics: an energy-weighted penalty preserves the geometry of pre-trained representations, while a recursive null-space alignment maintains safe orthogonality to historical edits, enabling stable retention of both sequentially edited knowledge and pre-trained general capabilities. The resulting unified objective admits a unique closed-form solution. We further establish three theoretical propositions that demonstrate the advantages of each component and characterize their synergy.

To validate the effectiveness of our method, we conduct extensive experiments using two base LLMs on four unstructured benchmarks. The results show that ManiEdit achieves state-of-the-art performance, surpassing the strongest competitor by up to $+27.81$ BERTScore and $+8.50$ ROUGE-L. After 500 sequential edits, ManiEdit retains general capabilities within 1–2% of the pre-edited model on most of the representative downstream tasks, whereas all baselines collapse within 100–200 edits. Further analysis experiments, ablation studies, and long-horizon stress tests confirm the robustness and efficacy of ManiEdit.

In summary, our main contributions are:

- We are the first to reframe knowledge editing as a localized displacement of an edit sub-manifold within the global manifold, decomposing it into the dual problems of edit point identification and manifold structure preservation.
- We propose *Pivot Localization*, which identifies high-leverage semantic pivots that anchor the edit sub-manifold, and *Manifold-Aware Preservation*, which decouples the preservation of two types of knowledge by their distinct manifold properties.
- Extensive experiments show ManiEdit achieves state-of-the-art performance on four unstructured editing benchmarks and retains near-original general capability on six downstream tasks.

2 PRELIMINARY

An autoregressive large language model (LLM) predicts the next token x in a sequence based on all preceding tokens. At layer l, the hidden state h^l of token x is computed as:

$$h^l = h^{l-1} + a^l + m^l, \tag{1}$$

where a^l and m^l denote the outputs of the multi-head self-attention (MHSA) and feed-forward network (FFN) modules, respectively. The FFN output is computed via a two-layer projection with a non-linear activation σ and layer normalization γ:

$$\underbrace{m^l}_{v} = W^l_{\text{down}} \underbrace{\sigma\big(W^l_{\text{up}}\, \gamma(h^{l-1} + a^l)\big)}_{k}, \tag{2}$$

where $W^l_{\text{up}} \in \mathbb{R}^{d_{\text{ff}} \times d}$ is the up-projection matrix, $W^l_{\text{down}} \in \mathbb{R}^{d \times d_{\text{ff}}}$ is the down-projection matrix, $k \in \mathbb{R}^{d_{\text{ff}}}$ is the activation serving as the *key*, and $v \in \mathbb{R}^d$ is the FFN output serving as the *value*.

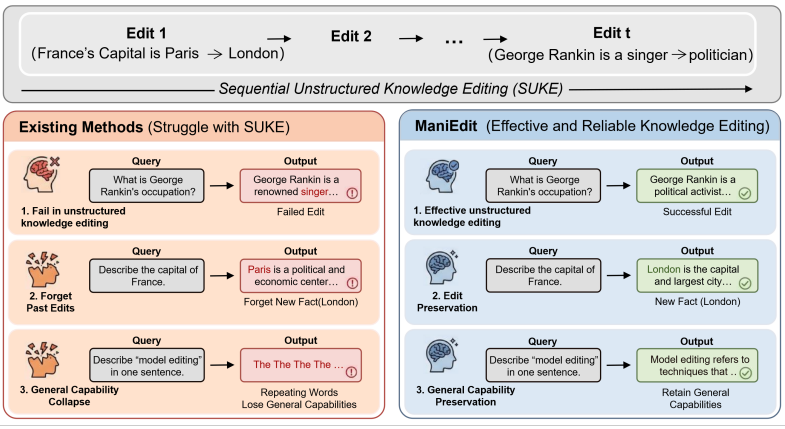

Figure 1: Motivation for ManiEdit. Existing methods struggle with sequential unstructured knowledge editing, suffering from edit forgetting and general-capability collapse (left). In contrast, ManiEdit enables effective and reliable knowledge editing, preserving both edit knowledge and pre-trained general capabilities (right).

Following Geva et al. (2021) and Meng et al. (2022), W_{down} can be interpreted as a *linear associative memory* that functions as key-value storage for factual knowledge retrieval. For instance, the key may encode the contextual representation of a query (e.g., "The capital of France is"), while the corresponding value encodes the associated answer (e.g., "Paris"). This perspective has inspired the *locate-then-edit* paradigm for knowledge editing, where targeted modifications to W_{down} enable precise updates to specific stored associations. Following Deng et al. (2025) and Jiang et al. (2025), we use the intermediate activation at the last token of each input chunk as the key k for weight updates throughout this work. The locate-then-edit procedure and its extension to long-form knowledge are detailed in Appendix A. For notational simplicity, we use W to denote W_{down} in subsequent sections.

3 Methodology

We now present **ManiEdit**, a paradigm for sequential unstructured knowledge editing. We first reframe editing from a representation-manifold perspective, then introduce two core components: *Pivot Localization* for identifying informative semantic anchors in unstructured knowledge and *Manifold-Aware Preservation* for geometry-aware knowledge protection (Figure 3). We finally derive a unified objective and analyze the theoretical advantages.

3.1 A Manifold Perspective of Model Editing

A well-established line of research has revealed that the hidden representations of LLMs exhibit rich geometric structure: activations concentrate along a structured, low-dimensional manifold (Mamou et al., 2020; Modell et al., 2025; Mabrok, 2026), whose principal axes encode the statistical regularities of language. Disrupting this learned representation geometry can therefore impair the model's general capabilities (Nishi et al., 2025). We build our geometric formulation of knowledge editing on this empirical foundation.

Input and output representation manifolds. Following the manifold hypothesis (Bengio et al., 2013; Fefferman et al., 2016), pre-trained key activations k do not fill the ambient key space uniformly but concentrate near a low-dimensional *input representation manifold* $\mathcal{M}_{\text{in}}$, a geometric structure whose principal directions are shaped by pre-training. We characterize this geometry via the second-moment matrix $\Sigma = \mathbb{E}[kk^\top]$, whose eigendecomposition $\Sigma = \Psi\Lambda\Psi^\top$ provides a linear approximation to the manifold's structure: an orthonormal basis $\Psi = [\psi_1, \ldots, \psi_n]$ with energy spectrum $\Lambda = \text{diag}(\lambda_1, \ldots, \lambda_n)$. Our framework retains the full spectrum and lets each eigenvalue λ_j encode the relative geometric importance of the corresponding direction. The weight W then maps $\mathcal{M}_{\text{in}}$ to the *output representation manifold*

$$\mathcal{M}_{\text{out}} \;=\; W(\mathcal{M}_{\text{in}}),$$

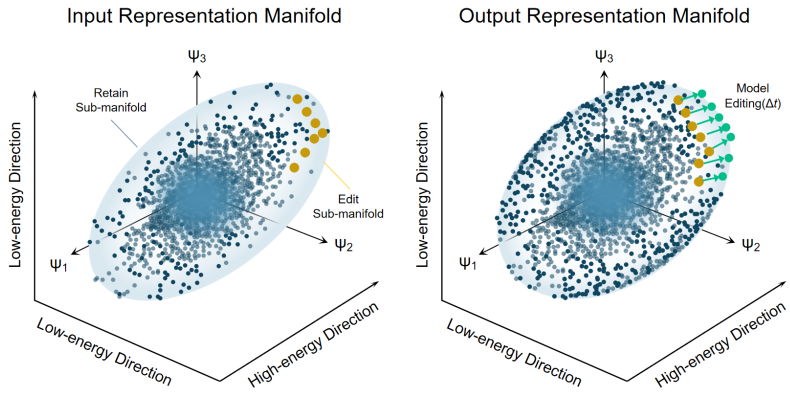

Figure 2: A manifold perspective of model editing. **Left**: The input representation manifold $\mathcal{M}_{\text{in}}$. Retain keys (blue) distribute along the principal axes $\{\psi_j\}$, forming a global sub-manifold $\mathcal{M}_{\text{in}}^{\text{retain}}$; edit-related keys (gold) form a local sub-manifold $\mathcal{M}_{\text{in}}^{\text{edit}}$. **Right**: The output representation manifold $\mathcal{M}_{\text{out}} = W(\mathcal{M}_{\text{in}})$. The edit operator $\mathcal{E}$ displaces the edit sub-manifold $\mathcal{M}_{\text{out}}^{\text{edit}}$ (gold) to its target $\mathcal{M}_{\text{out}}^{*,\text{edit}}$ (green), while the retain sub-manifold $\mathcal{M}_{\text{out}}^{\text{retain}}$ (blue) remains unchanged.

which encodes the associated factual knowledge as a geometric object in the output space. This two-manifold perspective is illustrated in Figure 2.

Knowledge editing as localized sub-manifold displacement. Let $\mathcal{M}_{\text{in}}^{\text{edit}} \subset \mathcal{M}_{\text{in}}$ be the *edit sub-manifold*—the region of $\mathcal{M}_{\text{in}}$ associated with the target knowledge to be modified—and let $\mathcal{M}_{\text{in}}^{\text{retain}} = \mathcal{M}_{\text{in}} \setminus \mathcal{M}_{\text{in}}^{\text{edit}}$ be the complementary *retain sub-manifold*. Their images under W partition the output manifold: $\mathcal{M}_{\text{out}}^{\text{edit}} = W(\mathcal{M}_{\text{in}}^{\text{edit}})$ and $\mathcal{M}_{\text{out}}^{\text{retain}} = W(\mathcal{M}_{\text{in}}^{\text{retain}})$. We define the model editing operator $\mathcal{E} : W' = \mathcal{E}(W)$ as a *localized sub-manifold displacement*, requiring W' to satisfy:

$$W'\big(\mathcal{M}_{\text{in}}^{\text{edit}}\big) = \mathcal{M}_{\text{out}}^{*,\text{edit}}, \qquad W'\big(\mathcal{M}_{\text{in}}^{\text{retain}}\big) = \mathcal{M}_{\text{out}}^{\text{retain}}. \tag{3}$$

The first condition displaces $\mathcal{M}_{\text{out}}^{\text{edit}}$ (gold) to the target $\mathcal{M}_{\text{out}}^{*,\text{edit}}$ (green), while the second holds $\mathcal{M}_{\text{out}}^{\text{retain}}$ (blue) invariant. The fundamental challenge of sequential long-form editing is therefore threefold: (i) to locate representative keys that anchor the edit sub-manifold $\mathcal{M}_{\text{in}}^{\text{edit}}$ to facilitate precise editing; (ii) to realize such localized displacements faithfully without distorting the global structure of $\mathcal{M}_{\text{out}}$; and (iii) to prevent subsequent edits from perturbing established displacements.

3.2 Pivot Localization

Achieving the edit objective in Equation 3 first requires addressing a challenge specific to unstructured long-form knowledge editing: how to select the representative keys that anchor each sample's edit sub-manifold $\mathcal{M}_{\text{in}}^{\text{edit}}$.

The Mediocre-Point Dilemma. To handle long-form knowledge, AnyEdit (Jiang et al., 2025) and AnyEdit++ (Tian et al., 2026) segment the edit sample—by a fixed sliding window, sentence boundaries, or peaks of Bayesian Surprise—and use the last token of each chunk as the representative edit key k. However, none of these strategies considers the geometric position of k within $\mathcal{M}_{\text{in}}$. The last-token key aggregates context from a broad mixture of preceding tokens, whose representations are distributed across the high-energy directions of Σ. This mixing effect may pull k toward the high-density interior of $\mathcal{M}_{\text{in}}$—the manifold centroid—producing a "mediocre point" that carries little discriminative information. Modifying the mapping of such keys requires a large cost (Section 3.5), distorting $\mathcal{M}_{\text{out}}$ and degrading the editing precision and effectiveness (Section 4.4).

Leverage Score for Pivot Localization. To address the mediocre-point dilemma, we introduce the leverage score to quantify how far a key k lies from the high-density interior:

$$S_{\text{leverage}}(k) = k^{\top}\Sigma^{-1}k, \tag{4}$$

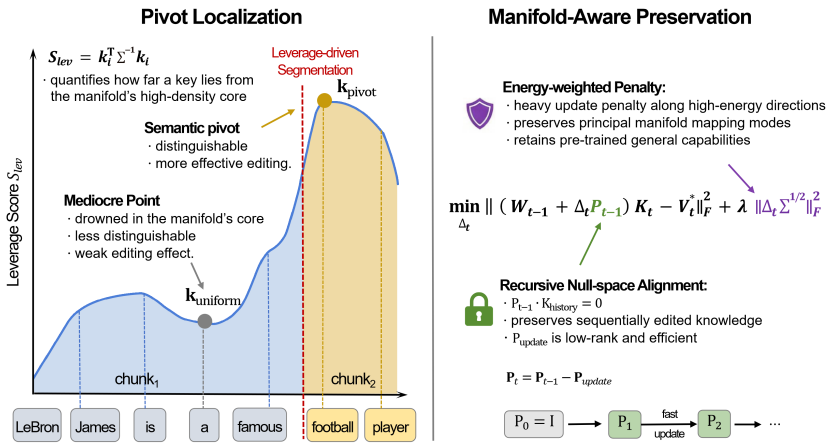

Figure 3: Overview of the ManiEdit paradigm. Pivot Localization identifies high-leverage semantic pivots to guide the segmentation of input samples, then the Manifold-Aware Preservation mechanism applies energy-weighted penalty and recursive null-space alignment to preserve general capabilities and historical edits.

where a higher leverage score indicates greater peripherality and richer discriminability. We segment the edit sample based on high-leverage keys, defined as semantic pivots, thereby anchoring the edit sub-manifold more effectively. Such semantic pivots require only a small cost to realize the displacement of $\mathcal{M}_{\text{out}}^{\text{edit}}$, which minimizes disruption to the global manifold structure, enhancing both update effectiveness and efficiency.

3.3 Manifold-Aware Preservation

Having selected the semantic pivots, we now instantiate $\mathcal{E}$ at editing step t by the additive update $W_t = W_{t-1} + \Delta_t$, and derive constraints on the perturbation Δ_t that preserve these two geometrically distinct types of knowledge.

Preserving the Geometric Structure of the Manifold. The retain sub-manifold $\mathcal{M}_{\text{in},t}^{\text{retain}}$ is a globally distributed continuous structure whose geometry is encoded in the principal axes $\{\psi_j\}$ and energies $\{\lambda_j\}$. Any weight perturbation Δ_t acts on $\mathcal{M}_{\text{in}}$ as a mapping distortion, with the magnitude along ψ_j given by $\|\Delta_t \psi_j\|_2$. To preserve the manifold geometry, distortions along high-energy directions should be penalized more strongly, as these directions capture dominant pre-trained geometric structure. Accordingly, we minimize the energy-weighted penalty:

$$\sum_{j=1}^{n} \lambda_j \|\Delta_t \psi_j\|_2^2 = \|\Delta_t \Sigma^{1/2}\|_F^2. \tag{5}$$

This yields a soft geometric constraint that preserves the pre-trained manifold structure without rigidly freezing the entire representation space.

Disentangling Sequentially Displaced Sub-Manifolds. Each historical edit $i < t$ has already established a localized manifold displacement: the edit sub-manifold $\mathcal{M}_{\text{out},i}^{\text{edit}}$ was repositioned to its target. A new update Δ_t should not disturb these established displacements. A soft constraint alone is fundamentally insufficient for this and cannot prevent subspace overlap between the current edit keys K_t and prior edit keys $\{K_i\}_{i<t}$, risking corruption of previously established sub-manifold locations. To effectively disentangle sequentially displaced sub-manifolds, we apply a recursive null-space alignment P_{t-1}, which satisfies $P_{t-1} K_i = 0$ for all $i < t$, guiding Δ_t toward the orthogonal complement of $\text{span}\{K_i\}_{i<t}$.

3.4 The Unified Optimization Objective

After establishing Pivot Localization and Manifold-Aware Preservation, we synthesize them into a unified optimization objective. At editing step t, the columns of K_t are the semantic-pivot keys

sampled from $\mathcal{M}_{\text{in},t}^{\text{edit}}$, and V_t^* is the corresponding target value matrix specifying the desired location of $\mathcal{M}_{\text{out},t}^{*,\text{edit}}$. The residual displacement that Δ_t must realize is $R_t = V_t^* - W_{t-1}K_t$. We seek to minimize the displacement error of the *recursive-null-space-alignment* update, subject to the *energy-weighted* penalty:

$$\min_{\Delta_t} \left\| (W_{t-1} + \Delta_t P_{t-1}) K_t - V_t^* \right\|_F^2 + \lambda \|\Delta_t \Sigma^{1/2}\|_F^2. \tag{6}$$

The first term guides the effective update toward the orthogonal complement of $\text{span}\{K_i\}_{i<t}$, keeping the displacement of $\mathcal{M}_{\text{out},t}^{\text{edit}}$ from perturbing previously established sub-manifolds. The second term introduces a manifold-geometry-weighted penalty that suppresses distortion of $\mathcal{M}_{\text{in}}$ along high-energy directions, preserving the global manifold geometry. W_{t-1} is the current weight matrix and λ balances the two constraints.

This objective admits a unique global closed-form solution (see Appendix D for the full derivation):

$$\Delta_t = R_t K_t^\top P_{t-1} \left(P_{t-1} K_t K_t^\top P_{t-1} + \lambda \Sigma \right)^{-1}. \tag{7}$$

We then update $W_t = W_{t-1} + \Delta_t$ (see Appendix C.8 for a justification).

To incorporate the newly displaced sub-manifold and protect it in subsequent edits, the recursive null-space alignment matrix is updated by excluding the subspace spanned by K_t:

$$P_t = P_{t-1} - P_{t-1} K_t \left(K_t^\top P_{t-1} K_t + \epsilon I \right)^{-1} K_t^\top P_{t-1}, \tag{8}$$

where $P_0 = I$ and ϵ is a numerical stability constant. This recursive update accumulates all edit sub-manifolds into the protected subspace, thus effectively preserving every historical edit.

3.5 Theoretical Advantages

To unveil how the components of ManiEdit collaborate to realize the model editing goal of Equation 3, we present the following three propositions (see Appendix E for full derivation).

Proposition 1. *To achieve a fixed displacement $\Delta k_t = R$, the minimum cost is $\|\Delta\Sigma^{1/2}\|_F^2 = \|R\|^2/(k_t^\top \Sigma^{-1} k_t)$, which is strictly decreasing in the leverage score $k_t^\top \Sigma^{-1} k_t$.*

This explains how high-leverage semantic pivots improve the effectiveness of each weight update: they require only a small cost (little disruption to global manifold structure) to realize the displacement of $\mathcal{M}_{\text{out},t}^{\text{edit}}$, reducing unintended perturbations.

Proposition 2. *Consider the single-key case $K_t = k_t$, and let $\hat{k}_t = P_{t-1}k_t$ and $\tilde{k}_j = \psi_j^\top \hat{k}_t$. Then the closed-form update in Equation 7 satisfies $\Delta W_{dir\ j} := \Delta_t \psi_j \propto \tilde{k}_j/\lambda_j$.*

The numerator $\tilde{k}_j$ measures the semantic demand of the edit along ψ_j, while the denominator λ_j acts as an energy-adaptive penalty: directions that genuinely need updating receive capacity, whereas perturbations along dominant pre-trained directions are suppressed.

Proposition 3. *Set $\epsilon = 0$ in Equation 8 and replace the inverse with the pseudo-inverse. Then for every $t \geq 1$: (i) $P_t K_i = 0$ for all $i \leq t$; (ii) $\text{rank}(P_t) \geq n - \sum_{i=1}^{t} \text{rank}(K_i)$.*

Unlike traditional null-space methods whose P enforces $\Delta W_{\text{dir}\,j} = 0$ for $\lambda_j > \theta$ (e.g., AlphaEdit (Fang et al., 2025), EvoEdit (Lyu et al., 2026)), which severely compresses the feasible-update space and impairs editing efficacy, our P_t allows energy-adaptive weight updates that preserve historical edits without collapsing the feasible-update rank.

Putting these components together: Pivot Localization first identifies easily displaceable regions on $\mathcal{M}_{\text{in}}^{\text{edit}}$, enabling precise editing at low cost. During the displacement, the energy-weighted penalty protects the global manifold structure from unintended distortion. Finally, recursive null-space alignment stabilizes the relocated edit sub-manifold. This joint mechanism ensures effective editing while preserving historical edits and general capabilities.

4 Experiments

To rigorously evaluate the effectiveness of ManiEdit, we design a comprehensive set of experiments focusing on sequential long-form knowledge editing. Our evaluation seeks to answer the following research questions:

Method	UnKEBench				AKEW (Counterfact)				AKEW (MQUAKE)			
	Ori		Para		Ori		Para		Ori		Para	
	BERTScore	ROUGE-L	BERTScore	ROUGE-L	BERTScore	ROUGE-L	BERTScore	ROUGE-L	BERTScore	ROUGE-L	BERTScore	ROUGE-L
Based on Qwen2.5-7B-Instruct												
MEMIT	8.36	16.89	8.95	16.73	9.07	19.23	9.43	19.22	10.47	18.71	9.17	18.67
RECT	12.28	17.57	12.35	17.07	6.21	17.08	5.90	16.76	13.34	21.90	13.20	21.37
EvoEdit	25.79	16.08	21.60	15.16	35.89	24.56	19.85	20.95	52.28	26.35	44.51	21.76
UnKE	9.45	8.12	12.78	10.55	5.41	9.51	4.28	10.41	12.69	18.37	10.46	17.28
AnyEdit	9.80	15.94	9.67	15.49	1.06	13.66	0.75	13.38	7.80	15.02	6.88	16.03
AnyEdit++	21.90	13.21	21.61	13.15	37.42	22.32	28.51	21.10	49.24	30.40	41.65	23.97
FT-UKE	34.82	21.28	34.18	20.36	37.16	21.87	18.17	15.28	52.73	34.75	46.95	29.85
MOSE	75.13	31.72	**75.43**	30.87	71.03	29.67	**53.98**	24.54	66.05	34.87	66.14	32.47
ManiEdit	**76.63**	**39.43**	74.97	**38.07**	**73.88**	**38.17**	50.78	**31.45**	**71.37**	**37.96**	**70.87**	**35.08**
Based on Llama3-8B-Instruct												
MEMIT	2.00	1.18	2.44	1.05	1.58	6.15	1.60	6.37	0.97	5.81	1.53	6.10
RECT	10.91	16.69	11.37	16.73	2.00	10.27	1.97	10.26	0.00	8.88	0.00	8.71
EvoEdit	1.67	9.77	1.52	9.91	19.20	12.37	7.45	9.82	23.58	9.89	24.63	9.87
UnKE	1.26	1.12	1.76	1.20	0.61	3.33	0.59	3.49	2.49	5.04	0.10	2.83
AnyEdit	11.55	14.40	11.40	13.95	4.40	8.55	5.11	9.17	8.02	14.19	8.70	13.86
MOSE	62.38	31.76	61.11	30.95	50.69	30.65	43.77	24.08	63.48	**37.97**	63.57	**36.98**
ManiEdit	**76.22**	**37.30**	**75.80**	**35.61**	**78.50**	**37.24**	**53.88**	**28.73**	**71.51**	30.03	**71.24**	35.34

Table 1: Comparison of ManiEdit with existing methods on the sequential unstructured model editing tasks. The best results are highlighted in bold, while the second-best results are underlined.

- **RQ1 (Overall Performance):** How does ManiEdit perform on sequential unstructured editing tasks compared to baseline methods? Can it mitigate the issue of edit forgetting?
- **RQ2 (General Capability Test):** How well does ManiEdit maintain general capabilities after hundreds of continual edits? Can it prevent the collapse of general capabilities?
- **RQ3 (Analysis of Pivot Localization):** Can Pivot Localization consistently improve the editing effectiveness of ManiEdit and other methods?

Due to space limitations, further ablations of Manifold-Aware Preservation, long-horizon stress tests, and other extended experiments are provided in Appendix C.

4.1 Experimental Setup

In this subsection, we briefly outline the base LLMs, baseline methods, datasets, and evaluation metrics. More details are provided in Appendix B.

Base LLMs & Baseline Methods. We experiment on two widely adopted instruction-tuned LLMs: Llama3-8B-Instruct (Grattafiori et al., 2024) and Qwen2.5-7B-Instruct (Yang et al., 2025). We compare ManiEdit against a comprehensive suite of model editing baselines spanning multiple paradigms: (1) *Traditional editing methods*: MEMIT (Meng et al., 2023) and RECT (Gu et al., 2024); (2) *Sequential editing methods*: EvoEdit (Lyu et al., 2026) and MOSE (Xu et al., 2026); (3) *Unstructured editing methods*: UnKE (Deng et al., 2025), FT-UKE (Xiong et al., 2025), AnyEdit (Jiang et al., 2025), and AnyEdit++ (Tian et al., 2026). We further include three memory-based editing methods (i.e., SERAC (Mitchell et al., 2022b), GRACE (Hartvigsen et al., 2023), and MELO (Yu et al., 2024)) as additional baselines in Appendix C.1.

Datasets & Evaluation Metrics. We evaluate ManiEdit on four established benchmarks for sequential unstructured knowledge editing: UnKEBench (Deng et al., 2025), AKEW (Counterfact) (Wu et al., 2024), AKEW (MQUAKE) (Wu et al., 2024), and EditEverything (Jiang et al., 2025). We evaluate the similarity between the edited model's outputs and the editing targets under three query conditions: Original (Ori), Paraphrase (Para), and Sub-question (Sub). The evaluation uses BERTScore (Zhang et al., 2020) for semantic similarity and ROUGE-L (Lin, 2004) for lexical overlap.

4.2 Sequential Unstructured Editing (RQ1)

To evaluate the performance of different editing methods on sequential unstructured knowledge editing, we conduct experiments on two base LLMs and four benchmarks, comparing ManiEdit with all baselines. Table 1 presents the results under a commonly used configuration for the sequential editing task, where 500 samples are randomly drawn from the dataset for updates, with one sample per edit (i.e., a batch size of 1). Since AKEW (MQUAKE) contains only 354 samples in total, we evaluate it on the full set rather than sampling 500. More sub-question-level evaluations for each

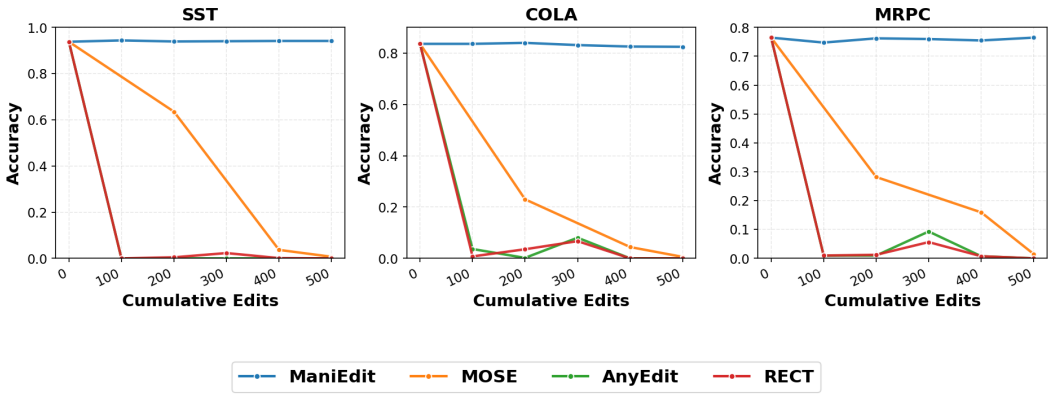

Figure 4: General capability accuracy on three GLUE tasks (SST-2, CoLA, MRPC) as the number of sequential unstructured edits increases. ManiEdit maintains near-original accuracy across all tasks, while baselines collapse rapidly.

benchmark are reported in Appendix C.2. Results on EditEverything are provided in Appendix C.4. Based on Table 1, we draw the following observations:

ManiEdit achieves the best overall performance across all benchmarks and LLMs. Compared with the strongest baseline, ManiEdit improves on 20 out of 24 metric columns. The largest gains reach **+27.81** BERTScore on Llama3-8B-Instruct AKEW (Counterfact) Ori and **+8.50** ROUGE-L on Qwen2.5-7B-Instruct AKEW (Counterfact) Ori. Averaged over all settings, ManiEdit improves upon the strongest baseline by +7.74 BERTScore and +3.99 ROUGE-L. These gains arise from ManiEdit's precise localization of semantic pivots and its protection of historical edit sub-manifolds.

4.3 General Capability Test (RQ2)

To evaluate whether post-edited LLMs retain their intrinsic general capabilities, we perform sequential unstructured editing on Qwen2.5-7B-Instruct for 500 edits and track the downstream task accuracy at intervals of 100 edits. Figure 4 reports the accuracy curves for ManiEdit against three competitive baselines—MOSE, AnyEdit, and RECT—evaluated on three representative GLUE tasks: SST-2 (Socher et al., 2013) (sentiment classification), CoLA (Warstadt et al., 2019) (linguistic acceptability), and MRPC (Dolan & Brockett, 2005) (paraphrase detection). Further experiments on MMLU (Hendrycks et al., 2021), MNLI (Williams et al., 2018), and RTE (Bentivogli et al., 2009) are reported in Appendix C.3. Based on Figure 4, we draw the following observations:

Baseline methods suffer catastrophic collapse of general capabilities within 100–200 edits. AnyEdit and RECT drop to near-zero accuracy on all three tasks after merely 100 edits, indicating that their parameter updates severely damage the model's general language ability. MOSE exhibits a slower but still fatal decline—suffering substantial accuracy degradation across all three tasks by 200 edits and ultimately collapsing to near-zero levels by 500 edits. This confirms that existing methods struggle to prevent general-capability collapse.

ManiEdit sustains near-original general capabilities even after 500 edits. Across all three tasks, ManiEdit maintains accuracy within 1–2% of the unedited model throughout the entire 500-edit sequence (e.g., SST-2: 93.8% $\rightarrow$ 94.2%; CoLA: 83.6% $\rightarrow$ 82.5%; MRPC: 76.5% $\rightarrow$ 76.5%). This stability arises from ManiEdit's protection of the geometry of the pre-trained manifold.

4.4 Analysis of Pivot Localization (RQ3)

To examine whether Pivot Localization provides a general benefit beyond ManiEdit itself, we evaluate it under two complementary settings: removing Pivot Localization from ManiEdit, and adding it to EvoEdit, a representative locate-then-edit baseline. The experimental setting follows Table 1. Figure 5 reports Para ROUGE-L across three benchmarks on two base LLMs. Additional Ori ROUGE-L results are provided in Appendix C.5. We draw the following observation based on Figure 5:

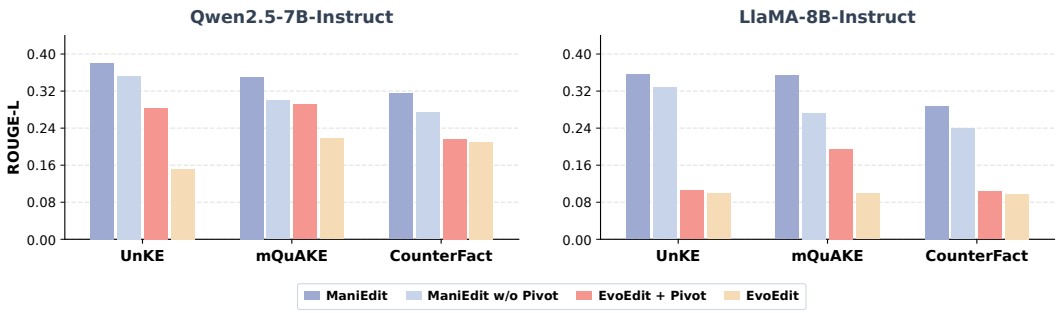

Figure 5: Analysis of Pivot Localization. Scores are Para ROUGE-L on two base LLMs across three benchmarks. ManiEdit w/o Pivot and EvoEdit use AnyEdit's fixed-window chunking.

Pivot Localization consistently improves editing effectiveness. For EvoEdit, adding Pivot Localization improves ROUGE-L in all six settings, with large gains on Qwen-UnKEBench (+13.13), Qwen-MQUAKE (+7.37), and Llama-MQUAKE (+9.68). This indicates that semantic pivots are not tied to ManiEdit's preservation module, but can also strengthen other methods. For ManiEdit, removing Pivot Localization lowers ROUGE-L in every setting; the full model improves over the no-pivot variant by +4.62 ROUGE-L on average, ranging from +2.82 on Llama-UnKEBench to +8.10 on Llama-MQUAKE. These gains confirm that Pivot Localization improves the editing effectiveness of ManiEdit and other methods consistently.

5 Related Work

This section summarizes related work; a comprehensive discussion is presented in Appendix F.

Knowledge Editing. Traditional methods modify targeted facts represented as triplets. *Parameter-preserving* methods utilize additional modules to store to-be-updated knowledge. These modules may include codebooks, matrices, or auxiliary models, as seen in methods like GRACE (Hartvigsen et al., 2023), MELO (Yu et al., 2024), and SERAC (Mitchell et al., 2022b). *Parameter-modifying* approaches permanently integrate knowledge into the base weights. Hyper-network methods such as MEND (Mitchell et al., 2022a) train auxiliary networks to predict weight updates. The Locate-then-Edit paradigm, exemplified by MEMIT (Meng et al., 2023), localizes critical knowledge neurons via causal tracing and applies closed-form updates. AlphaEdit (Fang et al., 2025) and EvoEdit (Lyu et al., 2026) employ null-space projection to mitigate decay under sequential editing. OTE-SE (Wang et al., 2026) keeps the accumulated sequential update equivalent to the corresponding one-time editing solution. MOSE (Xu et al., 2026) proposes multiplicative orthogonal updates as an alternative to standard additive weight increments.

Unstructured Knowledge Editing. Real-world knowledge is typically embedded in unstructured, long-form text with rich contextual dependencies. Recent work has begun to address this more challenging setting. UnKE (Deng et al., 2025) treats the model as a key-encoder and value-generator and trains critical layers. FT-UKE (Xiong et al., 2025) provides a training recipe that makes vanilla fine-tuning effective for unstructured knowledge editing. AnyEdit (Jiang et al., 2025) decomposes long-form text into uniform chunks and iteratively edits the last token of each chunk. AnyEdit++ (Tian et al., 2026) instead places chunk boundaries at peaks of Bayesian Surprise.

6 Conclusion

In this paper, we present ManiEdit for sequential unstructured knowledge editing. By novelly reframing editing as a localized sub-manifold displacement, we propose Pivot Localization and Manifold-Aware Preservation for edit point identification and manifold structure preservation. Experiments confirm that ManiEdit not only achieves superior editing performance but also effectively prevents edit forgetting and general-capability collapse, marking a significant advance toward the real-world deployment of knowledge editing.

AI USE STATEMENT

In this work, we used generative AI tools to edit the manuscript for readability, to format references, and to assist in the writing of proofs. We have not used generative AI tools to help develop theoretical models or conceptual frameworks, formulate mathematical claims, provide critical ingredients for proving mathematical claims, propose or refine hypotheses, design or provide feedback on research methodology or experiments, implement methods, or interpret results. Generating synthetic datasets, assisting with translation, cleaning and reformatting datasets, and supporting qualitative and thematic data analysis are not applicable to this work. We have reviewed all AI-assisted work: LLM-assisted text and proof write-ups were checked and edited by the authors for factual accuracy. We take responsibility for the final content of this work, including text, claims or artifacts produced with the aid of generative AI.

REPRODUCIBILITY STATEMENT

We have taken the following steps to facilitate reproducibility. The ManiEdit algorithm, including Pivot Localization and the closed-form Manifold-Aware Preservation update, is specified in Section 3. Complete proofs of the theoretical claims are given in Appendices D and E. Backbone models, baselines, hyperparameters, datasets, evaluation metrics, and hardware settings are reported in Appendix B. All editing and general-capability benchmarks used in this work are publicly available. An anonymous implementation is provided in the supplementary material.

REFERENCES

Gabriele Bavota. Mining unstructured data in software repositories: Current and future trends. In *2016 IEEE 23rd International Conference on Software Analysis, Evolution, and Reengineering (SANER)*, pp. 1–12, Osaka, Japan, 2016. IEEE. doi: 10.1109/SANER.2016.47.

Yoshua Bengio, Aaron Courville, and Pascal Vincent. Representation learning: A review and new perspectives. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 35(8):1798–1828, 2013. doi: 10.1109/TPAMI.2013.50.

Luisa Bentivogli, Ido Dagan, Hoa Trang Dang, Danilo Giampiccolo, and Bernardo Magnini. The fifth PASCAL recognizing textual entailment challenge. In *Proceedings of the Second Text Analysis Conference (TAC 2009)*, 2009.

Tom B. Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, Sandhini Agarwal, Ariel Herbert-Voss, Gretchen Krueger, Tom Henighan, Rewon Child, Aditya Ramesh, Daniel M. Ziegler, Jeffrey Wu, Clemens Winter, Christopher Hesse, Mark Chen, Eric Sigler, Mateusz Litwin, Scott Gray, Benjamin Chess, Jack Clark, Christopher Berner, Sam McCandlish, Alec Radford, Ilya Sutskever, and Dario Amodei. Language models are few-shot learners. In *Advances in Neural Information Processing Systems*, volume 33, pp. 1877–1901. Curran Associates, Inc., 2020.

Pengfei Cao, Zeao Ji, Daojian Zeng, Jun Zhao, and Kang Liu. EvoEdit: Lifelong free-text knowledge editing through latent perturbation augmentation and knowledge-driven parameter fusion, 2025. URL `https://arxiv.org/abs/2512.04545`.

Damai Dai, Li Dong, Yaru Hao, Zhifang Sui, Baobao Chang, and Furu Wei. Knowledge neurons in pretrained transformers. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 8493–8502, Dublin, Ireland, 2022. Association for Computational Linguistics. doi: 10.18653/v1/2022.acl-long.581.

Nicola De Cao, Wilker Aziz, and Ivan Titov. Editing factual knowledge in language models. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pp. 6491–6506, Online and Punta Cana, Dominican Republic, 2021. Association for Computational Linguistics. doi: 10.18653/v1/2021.emnlp-main.522.

Jingcheng Deng, Zihao Wei, Liang Pang, Hanxing Ding, Huawei Shen, and Xueqi Cheng. Everything is editable: Extend knowledge editing to unstructured data in large language models. In *Proceedings of the International Conference on Learning Representations*, 2025.

William B. Dolan and Chris Brockett. Automatically constructing a corpus of sentential paraphrases. In *Proceedings of the Third International Workshop on Paraphrasing (IWP2005)*, 2005.

Qingxiu Dong, Damai Dai, Yifan Song, Jingjing Xu, Zhifang Sui, and Lei Li. Calibrating factual knowledge in pretrained language models. In *Findings of the Association for Computational Linguistics: EMNLP 2022*, pp. 5937–5947, Abu Dhabi, United Arab Emirates, 2022. Association for Computational Linguistics. doi: 10.18653/v1/2022.findings-emnlp.438.

Junfeng Fang, Houcheng Jiang, Kun Wang, Yunshan Ma, Jie Shi, Xiang Wang, Xiangnan He, and Tat-Seng Chua. AlphaEdit: Null-space constrained knowledge editing for language models. In *Proceedings of the International Conference on Learning Representations*, 2025.

Charles Fefferman, Sanjoy Mitter, and Hariharan Narayanan. Testing the manifold hypothesis. *Journal of the American Mathematical Society*, 29(4):983–1049, 2016. doi: 10.1090/jams/852.

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

Aaron Grattafiori, Abhimanyu Dubey, Abhinav Jauhri, et al. The Llama 3 herd of models, 2024. URL `https://arxiv.org/abs/2407.21783`.

Jia-Chen Gu, Hao-Xiang Xu, Jun-Yu Ma, Pan Lu, Zhen-Hua Ling, Kai-Wei Chang, and Nanyun Peng. 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*, pp. 16801–16819, Miami, Florida, USA, 2024. Association for Computational Linguistics. doi: 10.18653/v1/2024.emnlp-main.934.

Xiaojie Gu, Ziying Huang, Jia-Chen Gu, and Kai Zhang. UltraEdit: Training-, subject-, and memory-free lifelong editing in language models. *Transactions on Machine Learning Research*, 2026.

Akshat Gupta, Phudish Prateepamornkul, Maochuan Lu, Ahmed Alaa, Thomas Hartvigsen, and Gopala Anumanchipalli. Lifelong knowledge editing requires better regularization. In *Findings of the Association for Computational Linguistics: EMNLP 2025*, pp. 22653–22675, Suzhou, China, 2025. Association for Computational Linguistics. doi: 10.18653/v1/2025.findings-emnlp.1234.

Anshita Gupta, Debanjan Mondal, Akshay Krishna Sheshadri, Wenlong Zhao, Xiang Lorraine Li, Sarah Wiegreffe, and Niket Tandon. Editing common sense in transformers. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pp. 8214–8232, Singapore, 2023. Association for Computational Linguistics. doi: 10.18653/v1/2023.emnlp-main.511.

Thomas Hartvigsen, Swami Sankaranarayanan, Hamid Palangi, Yoon Kim, and Marzyeh Ghassemi. Aging with GRACE: Lifelong model editing with discrete key-value adaptors. In *Advances in Neural Information Processing Systems*, volume 36, pp. 47934–47959. Curran Associates, Inc., 2023.

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

Xiusheng Huang, Yequan Wang, Jun Zhao, and Kang Liu. Commonsense knowledge editing based on free-text in LLMs. In *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing*, pp. 14870–14880, Miami, Florida, USA, 2024. Association for Computational Linguistics. doi: 10.18653/v1/2024.emnlp-main.826.

Zeyu Huang, Yikang Shen, Xiaofeng Zhang, Jie Zhou, Wenge Rong, and Zhang Xiong. Transformer-Patcher: One mistake worth one neuron. In *Proceedings of the International Conference on Learning Representations*, 2023.

Houcheng Jiang, Junfeng Fang, Ningyu Zhang, Mingyang Wan, Guojun Ma, Xiang Wang, Xiangnan He, and Tat-Seng Chua. AnyEdit: Edit any knowledge encoded in language models. In *Proceedings of the 42nd International Conference on Machine Learning*, volume 267 of *Proceedings of Machine Learning Research*, pp. 27510–27533. PMLR, 2025.

Xiaopeng Li, Shasha Li, Shezheng Song, Jing Yang, Jun Ma, and Jie Yu. PMET: Precise model editing in a transformer. *Proceedings of the AAAI Conference on Artificial Intelligence*, 38(17): 18564–18572, 2024. doi: 10.1609/aaai.v38i17.29818.

Chin-Yew Lin. ROUGE: A package for automatic evaluation of summaries. In *Text Summarization Branches Out*, pp. 74–81, Barcelona, Spain, 2004. Association for Computational Linguistics.

Mingda Liu, Zhenghan Zhu, Ze'an Miao, and Katsuki Fujisawa. Norm anchors make model edits last, 2026a. URL https://arxiv.org/abs/2602.02543.

Xuyuan Liu, Shengyu Chen, Xinshuai Dong, Yanchi Liu, Xujiang Zhao, Haoyu Wang, Yujun Yan, Haifeng Chen, and Zhengzhang Chen. Representation interventions enable lifelong knowledge memory control in LLMs. In *Proceedings of the 64th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 5414–5436, San Diego, California, United States, 2026b. Association for Computational Linguistics. doi: 10.18653/v1/2026.acl-long.246.

Sicheng Lyu, Yu Gu, Xinyu Wang, Jerry Huang, Sitao Luan, Yufei Cui, Xiao-Wen Chang, and Peng Lu. EvoEdit: Evolving null-space alignment for robust and efficient knowledge editing. In *Findings of the Association for Computational Linguistics: ACL 2026*, pp. 1520–1540, San Diego, California, United States, 2026. Association for Computational Linguistics. doi: 10.18653/v1/2026.findings-acl.75.

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

Mohamed A. Mabrok. Latent semantic manifolds in large language models, 2026. URL https://arxiv.org/abs/2603.22301.

Jonathan Mamou, Hang Le, Miguel Del Rio, Cory Stephenson, Hanlin Tang, Yoon Kim, and SueYeon Chung. Emergence of separable manifolds in deep language representations. In *Proceedings of the 37th International Conference on Machine Learning*, volume 119 of *Proceedings of Machine Learning Research*, pp. 6713–6723. PMLR, 2020.

Kevin Meng, David Bau, Alex Andonian, and Yonatan Belinkov. Locating and editing factual associations in GPT. In *Advances in Neural Information Processing Systems*, volume 35, pp. 17359–17372. Curran Associates, Inc., 2022.

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

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

Eric Mitchell, Charles Lin, Antoine Bosselut, Christopher D. Manning, and Chelsea Finn. Memory-based model editing at scale. In *Proceedings of the 39th International Conference on Machine Learning*, volume 162 of *Proceedings of Machine Learning Research*, pp. 15817–15831. PMLR, 2022b.

Alexander Modell, Patrick Rubin-Delanchy, and Nick Whiteley. The origins of representation manifolds in large language models, 2025. URL https://arxiv.org/abs/2505.18235.

Kento Nishi, Rahul Ramesh, Maya Okawa, Mikail Khona, Hidenori Tanaka, and Ekdeep Singh Lubana. Representation shattering in transformers: A synthetic study with knowledge editing. In *Proceedings of the 42nd International Conference on Machine Learning*, volume 267 of *Proceedings of Machine Learning Research*, pp. 46525–46553. PMLR, 2025.

Fabio Petroni, Tim Rocktäschel, Sebastian Riedel, Patrick Lewis, Anton Bakhtin, Yuxiang Wu, and Alexander Miller. Language models as knowledge bases? In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pp. 2463–2473, Hong Kong, China, 2019. Association for Computational Linguistics. doi: 10.18653/v1/D19-1250.

Adam Roberts, Colin Raffel, and Noam Shazeer. How much knowledge can you pack into the parameters of a language model? In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pp. 5418–5426, Online, 2020. Association for Computational Linguistics. doi: 10.18653/v1/2020.emnlp-main.437.

Richard Socher, Alex Perelygin, Jean Wu, Jason Chuang, Christopher D. Manning, Andrew Ng, and Christopher Potts. Recursive deep models for semantic compositionality over a sentiment treebank. In *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing*, pp. 1631–1642, Seattle, Washington, USA, 2013. Association for Computational Linguistics. doi: 10.18653/v1/D13-1170.

Zian Su, Ziyang Huang, Kaiyuan Zhang, and Xiangyu Zhang. μKE: Matryoshka unstructured knowledge editing of large language models. In *Proceedings of the Conference on Language Modeling*, 2025.

Bowen Tian, Caixue He, Jiemin Wu, Jingying Wang, Wenshuo Chen, Zexi Li, and Yutao Yue. AnyEdit++: Adaptive long-form knowledge editing via Bayesian surprise. In *Proceedings of the 43rd International Conference on Machine Learning*. PMLR, 2026.

Ke Wang, Yiming Qin, Nikolaos Dimitriadis, Alessandro Favero, and Pascal Frossard. MEMOIR: Lifelong model editing with minimal overwrite and informed retention for LLMs. In *Advances in Neural Information Processing Systems*, volume 38, pp. 99029–99056. Curran Associates, Inc., 2025.

Zheng Wang, Kaixuan Zhang, Wanfang Chen, Jingwen Zhang, and Xiaonan Lu. The labyrinth and the thread: Rethinking regularizations in sequential knowledge editing for large language models. In *Proceedings of the 43rd International Conference on Machine Learning*. PMLR, 2026.

Alex Warstadt, Amanpreet Singh, and Samuel R. Bowman. Neural network acceptability judgments. *Transactions of the Association for Computational Linguistics*, 7:625–641, 2019. doi: 10.1162/tacl_a_00290.

Adina Williams, Nikita Nangia, and Samuel R. Bowman. A broad-coverage challenge corpus for sentence understanding through inference. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pp. 1112–1122, New Orleans, Louisiana, 2018. Association for Computational Linguistics. doi: 10.18653/v1/N18-1101.

Xiaobao Wu, Liangming Pan, William Yang Wang, and Anh Tuan Luu. AKEW: Assessing knowledge editing in the wild. In *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing*, pp. 15118–15133, Miami, Florida, USA, 2024. Association for Computational Linguistics. doi: 10.18653/v1/2024.emnlp-main.843.

Hao Xiong, Chuanyuan Tan, and Wenliang Chen. Is fine-tuning an effective solution? Reassessing knowledge editing for unstructured data. In *Proceedings of the 14th International Joint Conference on Natural Language Processing and the 4th Conference of the Asia-Pacific Chapter of the Association for Computational Linguistics*, pp. 1348–1361, Mumbai, India, 2025. The Asian Federation of Natural Language Processing and The Association for Computational Linguistics. doi: 10.18653/v1/2025.ijcnlp-long.74.

Hao-Xiang Xu, Jun-Yu Ma, Ziqi Peng, Yuhao Sun, Zhen-Hua Ling, and Jia-Chen Gu. Multiplicative orthogonal sequential editing for language models. *Proceedings of the AAAI Conference on Artificial Intelligence*, 40(40):34124–34132, 2026. doi: 10.1609/aaai.v40i40.40707.

An Yang, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chengyuan Li, Dayiheng Liu, Fei Huang, Haoran Wei, Huan Lin, Jian Yang, Jianhong Tu, Jianwei Zhang, Jianxin

Yang, Jiaxi Yang, Jingren Zhou, Junyang Lin, Kai Dang, Keming Lu, Keqin Bao, Kexin Yang, Le Yu, Mei Li, Mingfeng Xue, Pei Zhang, Qin Zhu, Rui Men, Runji Lin, Tianhao Li, Tianyi Tang, Tingyu Xia, Xingzhang Ren, Xuancheng Ren, Yang Fan, Yang Su, Yichang Zhang, Yu Wan, Yuqiong Liu, Zeyu Cui, Zhenru Zhang, and Zihan Qiu. Qwen2.5 technical report, 2025. URL https://arxiv.org/abs/2412.15115.

Lang Yu, Qin Chen, Jie Zhou, and Liang He. MELO: Enhancing model editing with neuron-indexed dynamic LoRA. *Proceedings of the AAAI Conference on Artificial Intelligence*, 38(17): 19449–19457, 2024. doi: 10.1609/aaai.v38i17.29916.

Chenheng Zhang, Tianqi Du, Jizhe Zhang, Mingqing Xiao, Yifei Wang, Yisen Wang, and Zhouchen Lin. Language Ranker: A lightweight ranking framework for LLM decoding. In *Advances in Neural Information Processing Systems*, volume 38, pp. 133728–133752. Curran Associates, Inc., 2025.

Chi Zhang, Mengqi Zhang, Xiaotian Ye, Runxi Cheng, Zisheng Zhou, Ying Zhou, Pengjie Ren, and Zhumin Chen. Spectral characterization and mitigation of sequential knowledge editing collapse. In *Proceedings of the 64th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 30009–30032, San Diego, California, United States, 2026. Association for Computational Linguistics. doi: 10.18653/v1/2026.acl-long.1384.

Tianyi Zhang, Varsha Kishore, Felix Wu, Kilian Q. Weinberger, and Yoav Artzi. BERTScore: Evaluating text generation with BERT. In *Proceedings of the International Conference on Learning Representations*, 2020.

Wayne Xin Zhao, Kun Zhou, Junyi Li, Tianyi Tang, Zican Dong, Yupeng Hou, Beichen Zhang, Yingqian Min, Junjie Zhang, Peiyu Liu, Xiaolei Wang, Yifan Du, Chen Yang, Yushuo Chen, Zhipeng Chen, Jinhao Jiang, Ruiyang Ren, Yifan Li, Xinyu Tang, Zikang Liu, Yiwen Hu, Jian-Yun Nie, and Ji-Rong Wen. A survey of large language models. *Frontiers of Computer Science*, 20(12): 2012627, 2026. doi: 10.1007/s11704-026-60308-3.

Ce Zheng, Lei Li, Qingxiu Dong, Yuxuan Fan, Zhiyong Wu, Jingjing Xu, and Baobao Chang. Can we edit factual knowledge by in-context learning? In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pp. 4862–4876, Singapore, 2023. Association for Computational Linguistics. doi: 10.18653/v1/2023.emnlp-main.296.

A Extended Preliminaries

After the critical FFN layer is identified (typically via causal tracing), standard locate-then-edit methods first optimize a target value v^* for a chosen key k and then write the pair (k, v^*) into W (Section 2). We recap this procedure for structured facts, and then its extension to unstructured long-form knowledge.

A.1 Standard Locate-then-Edit Procedure

A structured fact is represented as a subject–relation–object triplet (s, r, o). ROME (Meng et al., 2022) and MEMIT (Meng et al., 2023) encode it as a key and a corresponding value: the key is the FFN activation at the last token of the subject s, and the value is the associated FFN output that should realize the object o.

To obtain this value, a residual δ is added to the current FFN output $h = Wk$ and optimized to raise the likelihood of o:

$$\delta^* = \arg\max_{\delta} \log P\big(o \mid s, r;\ h \leftarrow h + \delta\big). \tag{9}$$

The target value is then $v^* = h + \delta^*$.

The resulting pair (k, v^*) is then written into W by a constrained least-squares update that maps k to v^* while limiting drift on other keys (Meng et al., 2022; 2023). Although this procedure is effective for short facts, a single pair (k, v^*) is often insufficient for long-form text (Jiang et al., 2025).

A.2 Autoregressive Editing of Long-Form Knowledge

To edit unstructured long-form text Y, methods such as AnyEdit (Jiang et al., 2025) segment Y into chunks $\{C_j\}_{j=1}^{M}$ and apply vector optimization to each chunk in order. For chunk C_j, the key k_j is the FFN activation at the last token of the preceding chunk. A local residual δ_j is optimized to generate C_j given the prefix and previous chunks, while the residuals $\{\delta_i^*\}_{i<j}$ remain active in the forward pass:

$$\delta_j^* = \arg\max_{\delta_j} \log P\big(C_j \mid \text{prefix},\ C_{<j};\ \{h_i \leftarrow Wk_i + \delta_i^*\}_{i<j},\ h_j \leftarrow Wk_j + \delta_j\big). \tag{10}$$

Setting $v_j^* = Wk_j + \delta_j$ yields the pairs (k_j, v_j^*). After all pairs are collected, $\{(k_j, v_j^*)\}_{j=1}^{M}$ are then written into W in a single batch.

B Experimental Setup Details

B.1 Baseline Methods

We compare ManiEdit against a diverse set of knowledge editing baselines spanning three major paradigms. Below we provide a detailed description of each method.

Knowledge Editing Methods. These methods edit structured knowledge represented as subject–relation–object triplets. The dominant locate-then-edit paradigm identifies critical parameters (typically FFN weights) associated with the target fact and applies optimization-based closed-form updates.

- **MEMIT** (Meng et al., 2023) extends ROME to mass-editing by distributing parameter updates across multiple MLP layers via closed-form solutions. While effective for batch editing of structured triplets, it lacks mechanisms to prevent inter-edit interference in sequential settings, leading to edit forgetting.
- **RECT** (Gu et al., 2024) is a plug-in regularizer on locate-then-edit updates such as ROME and MEMIT, rather than a standalone editor. It keeps the entries of the update whose relative change is largest and sets the rest to zero, sparsifying the update so that it is less prone to overfit the edited facts.

- **EvoEdit** (Lyu et al., 2026) addresses sequential editing by projecting each new update into the null-space of previously protected knowledge, geometrically isolating successive edits. However, it applies hard projection uniformly to both pre-trained and edited knowledge, which collapses the feasible-update rank, thereby compressing the editable space and impairing editing efficacy.
- **MOSE** (Xu et al., 2026) proposes multiplicative orthogonal updates as an alternative to the standard additive weight increments, aiming to preserve the norm and condition number of the parameter matrix throughout sequential editing. This geometric stabilization strategy demonstrates strong retention on structured triplet editing, but overlooks the geometric asymmetry between pre-trained knowledge and sequential edits, leading to suboptimal outcomes in long-form scenarios.

Unstructured Editing Methods. These methods are specifically designed to handle unstructured, paragraph-level or document-level knowledge.

- **UnKE** (Deng et al., 2025) achieves unstructured knowledge editing by performing block-level (rather than single-layer) localization and gradient-descent-based optimization. While it achieves strong single-edit performance, its reliance on gradient descent without explicit orthogonal constraints makes it vulnerable to catastrophic interference when applied to sequential editing.
- **FT-UKE** (Xiong et al., 2025) reassesses fine-tuning for unstructured knowledge editing and provides a training recipe under which vanilla fine-tuning becomes a stronger UKE baseline. However, it likewise suffers from inter-edit interference under sequential editing, leading to edit forgetting.
- **AnyEdit** (Jiang et al., 2025) introduces a sequential chunking strategy that decomposes long-form text into fixed-length segments and iteratively applies AlphaEdit-style updates per chunk. This autoregressive editing paradigm enables processing of arbitrarily long texts but falls into the mediocre-point dilemma, degrading the editing precision and effectiveness.
- **AnyEdit++** (Tian et al., 2026) replaces uniform windows with adaptive chunk boundaries at peaks of Bayesian Surprise. While this segmentation better respects semantic transitions, it still ignores the geometric position of the edit key within $\mathcal{M}_{\text{in}}$ and thus remains subject to the mediocre-point dilemma.

Memory-based Methods. These methods utilize additional modules to store to-be-updated knowledge. Although they avoid modifying the original model weights, they rely heavily on retrieval accuracy and struggle with generalized queries that paraphrase edited facts. Furthermore, their reliance on external storage raises concerns regarding scalability and portability.

- **SERAC** (Mitchell et al., 2022b) adopts a semi-parametric architecture: edits are stored in an external memory bank, and a trained scope classifier determines at inference time whether a given input should be routed to an auxiliary counterfactual model or the original base model.
- **GRACE** (Hartvigsen et al., 2023) maintains a discrete key-value codebook at a designated model layer. During inference, the input's hidden representation is matched against codebook keys; if a match is found, the corresponding learned value vector replaces the original layer output. This design supports lifelong editing with inter-edit isolation, but effectiveness depends on the quality of the matching mechanism.
- **MELO** (Yu et al., 2024) assigns non-overlapping LoRA adaptor blocks to different edits, with a routing mechanism that activates the relevant adaptor at inference time. It introduces additional parameter modules, therefore incurring growing storage and routing overhead as edits accumulate.

B.2 DATASETS

We evaluate ManiEdit on four benchmarks that collectively cover unstructured long-form knowledge, counterfactual updates, multi-hop reasoning consistency, and diverse-domain editing.

UnKEBench. UnKEBench (Deng et al., 2025) is constructed to evaluate the editing of unstructured, relatively lengthy knowledge that goes beyond simple triplets or linear fact chains. It contains 1,000 counterfactual unstructured texts originating from ConflictQA, a benchmark specifically designed to distinguish LLMs' parametric memory from counter-memory. This design is crucial for ensuring a clear boundary between pre-training knowledge and edited knowledge, preventing the model from conflating the two. Each sample includes a main question (with a paraphrase variant) and a reference long-form answer as the editing target. Additionally, sub-question lists are provided for every sample for multi-angle testing of the edited knowledge.

AKEW (Counterfact). The AKEW benchmark (Wu et al., 2024) (Assessing Knowledge Editing in the Wild) provides a more realistic evaluation framework by supporting three input modalities for the same knowledge update: structured facts (isolated triplets), unstructured facts (natural language descriptions), and extracted triplets (automatically derived from unstructured text). The Counterfact subset contains 975 samples and focuses on counterfactual updates, where each sample contains a `requested_rewrite` specifying the subject, relation, and target replacement, along with both a structured one-sentence fact (`fact_new`) and a richer unstructured description (`fact_new_uns`). In this work, we primarily focus on the unstructured factual knowledge setting to align with our long-form editing objective.

AKEW (MQUAKE). The MQUAKE-CF subset within AKEW contains 354 samples and extends the evaluation to multi-hop reasoning consistency. Derived from the MQUAKE benchmark, it examines whether editing a base fact triggers the correct cascading changes in answers to multi-hop questions that logically depend on the edited fact. AKEW frames this multi-hop verification under the same three-modality protocol (structured / unstructured / extracted), enabling systematic comparison of how different input representations affect editing fidelity across reasoning chains. The sample fields mirror those of AKEW-Counterfact, with the key distinction being the emphasis on entailed multi-hop consequences rather than single-fact recall.

EditEverything. EditEverything (Jiang et al., 2025) integrates question-answering data from multiple specialized domains, constructing long and diverse knowledge formats that are substantially more challenging to edit. Specifically, the benchmark includes: (1) *Mathematics*: longer samples from the Orca-Math dataset containing grade-school math word problems; (2) *Code*: samples from the MBPP dataset, comprising crowd-sourced Python programming problems covering programming fundamentals and standard library functionalities; (3) *Chemistry*: problem-solution pairs from the Camel-Chemistry dataset, spanning 25 chemistry topics with structured subtopic hierarchies; (4) *Biology*: scientific QA pairs drawn from domain-specific corpora; (5) *News* and *Poetry*: since these categories often contain real-world knowledge that LLMs may already possess, synthetic data is generated using GPT-4o to ensure the target information is novel to the model. This cross-domain design tests whether editing methods can generalize beyond standard factual text to heterogeneous knowledge formats.

B.3 General Capability Benchmarks

Beyond the long-form knowledge editing benchmarks above, we evaluate general capability preservation using six established benchmarks.

SST-2. The Stanford Sentiment Treebank (Socher et al., 2013) is a single-sentence binary sentiment classification task built from movie review sentences with human-annotated sentiment labels. It is widely used to probe whether edited models retain surface-level language understanding.

MRPC. The Microsoft Research Paraphrase Corpus (Dolan & Brockett, 2005) is a sentence-pair benchmark for semantic equivalence detection. Each sample consists of two sentences, and the model must determine whether they are paraphrases. MRPC tests whether parameter edits preserve fine-grained semantic reasoning.

CoLA. The Corpus of Linguistic Acceptability (Warstadt et al., 2019) is a single-sentence classification task in which sentences are annotated as grammatically acceptable or unacceptable. It probes the model's internalized syntactic knowledge after editing.

MMLU. The Massive Multitask Language Understanding benchmark (Hendrycks et al., 2021) is a comprehensive evaluation that measures multi-task accuracy across 57 academic subjects, spanning STEM, humanities, and social sciences. It is evaluated under the zero-shot setting and provides a broad signal of whether general world knowledge is preserved after sequential editing.

MNLI. The Multi-Genre Natural Language Inference corpus (Williams et al., 2018) covers sentence-pair natural language inference across multiple genres of written and spoken English. Given a premise and a hypothesis, the model must infer the logical relationship (entailment, neutral, or contradiction). MNLI tests whether edits disrupt the model's cross-domain reasoning.

RTE. The Recognizing Textual Entailment challenge (Bentivogli et al., 2009) is a binary natural language inference task that determines whether a premise sentence logically entails a given hypothesis. It provides a focused test of deductive reasoning under sequential parameter perturbation.

B.4 Evaluation Metrics

Traditional evaluation metrics for structured knowledge editing—Efficacy, Generalization, and Specificity—are designed around exact-match comparisons with short target answers. While well-suited for triplet-based editing, they are insufficient for evaluating long-form and diverse-formatted knowledge, where the model may generate a response that captures the essential information yet differs substantially in surface form from the reference. To address this, we follow existing benchmarks for unstructured knowledge editing and adopt evaluation metrics that accommodate the verbosity and complexity of long-form responses.

Lexical Similarity. We employ **ROUGE-L** (Lin, 2004) to measure the longest common subsequence overlap between the model-generated text and the target answer. ROUGE-L captures the sequential structure of the generated response and provides insight into the surface-level accuracy of the edited content. ROUGE-1 is additionally reported in Appendix tables.

Semantic Similarity. Word-level overlap alone is insufficient for capturing the nuanced understanding that a model must exhibit after editing. A generated response may convey the correct meaning using different vocabulary or sentence structure, which lexical metrics would penalize. To bridge this gap, we employ **BERTScore** (Zhang et al., 2020), which leverages contextual embeddings to compute token-level cosine similarity between the generated text and the reference. BERTScore provides a more balanced evaluation that extends beyond lexical matching, quantifying the depth of the model's semantic comprehension of the edited knowledge. Following prior work, we report the F1 variant of BERTScore throughout our experiments.

Query Conditions. We evaluate editing quality under three query conditions:

- **Original (Ori):** The model is queried with the original question directly associated with the edited knowledge, testing whether the edit has been successfully incorporated. Both BERTScore and ROUGE are reported.
- **Paraphrase (Para):** The model is queried with a semantically equivalent but differently phrased question, testing whether the edited knowledge has been genuinely internalized rather than superficially memorized. Both BERTScore and ROUGE are reported.
- **Sub-question (Sub):** The model is queried with fine-grained sub-questions that probe specific aspects of the edited content. We report ROUGE against the corresponding sub-answers as a lexical measure of fine-grained coverage.

B.5 Implementation Details

All experiments are conducted on NVIDIA RTX PRO 5000 72GB Blackwell GPUs. We adopt greedy decoding, which makes generation deterministic for a fixed edited checkpoint, so we report single-run results rather than averaged statistics over multiple random seeds. For baselines, we primarily follow the default configurations provided in each method's official repository, with minor adjustments to align cross-method comparisons. Below we summarize the key hyperparameters.

ManiEdit. On Qwen2.5-7B-Instruct, we target layer 8 for editing. Text is segmented with the top-M highest-leverage semantic pivots as boundaries, rather than with a fixed-length window. The regularization weight λ is set to 15,000, with Σ estimated from 100,000 Wikipedia samples. The value optimization uses 25 gradient steps with a learning rate of 0.5, weight decay of 0.001, and clamp-norm factor 4. The loss is applied at layer 27. The projection matrix uses $\epsilon = 10^{-6}$ for numerical stability. On Llama3-8B-Instruct, the key differences are: $\lambda = 30{,}000$, loss layer 31. All other parameters remain the same.

MEMIT & RECT. Both methods use the MEMIT_ARE implementation from the AnyEdit codebase. Edits are distributed across layers [4–8] on both backbones. On Qwen, $\lambda = 15{,}000$ and window size is 50; on Llama3, $\lambda = 30{,}000$ and window size is 40. Value optimization follows the same protocol as ManiEdit (25 steps, lr 0.5, weight decay 0.001, clamp-norm factor 4). RECT additionally applies a sparsity fraction of 0.6 to regularize weight perturbations.

AnyEdit & EvoEdit. AnyEdit extends AlphaEdit with sequential chunking. Both methods share layers [4–8], and the same value optimization and window size settings as MEMIT per backbone. AnyEdit further introduces a null-space projection threshold of 0.02 and L2 weight of 10.

AnyEdit++. On Qwen2.5-7B-Instruct, layer 8 is targeted for editing. The value optimization uses 25 gradient steps at learning rate 0.5 (weight decay 10^{-3}, KL factor 0.0625, clamp-norm factor 4), the value-loss layer is 27, and $\lambda = 15{,}000$.

UnKE. UnKE performs gradient-descent-based block-level editing on layer 7 for both backbones, with a learning rate of 2×10^{-4} and 50 optimization steps (early-stop loss 0.1). To constrain parameter drift, 20 external data samples (max length 256) are used as locality anchors. The value loss layer is set to 27 (Qwen) or 31 (Llama3).

FT-UKE. FT-UKE performs gradient-descent-based editing of a single FFN `down_proj` matrix. On Qwen2.5-7B-Instruct, layer 27 is edited with a learning rate of 5×10^{-4}, and 10 optimization steps (early-stop loss 0.1). The maximum sequence length is 1024, and gradient clipping is set to 1.0.

MOSE. MOSE applies multiplicative orthogonal updates. On Qwen, layers [4, 5, 6] are edited with 20 optimization steps and learning rate 5×10^{-4}. On Llama3, layers [19, 20, 21] are edited with 25 optimization steps.

Memory-based Methods. **SERAC** pairs each base LLM with a smaller counterfactual model (Qwen2.5-0.5B-Instruct for Qwen; Llama-3.2-1B-Instruct for Llama3) and a DistilBERT scope classifier with a cache hit threshold of 0.5. **GRACE** performs editing at layer 18 (Qwen) or layer 27 (Llama3), with edit lr 1.0, 50 iterations, $\epsilon = 1.0$, and Euclidean distance for key matching. **MELO** builds upon GRACE by adding LoRA adaptors ($r = \alpha = 64$) on the two layers following the GRACE layer, with 350 blocks $\times$ 2 ranks per block (total rank 700), initial radius 0.5, and edit lr 0.001.

C Extended Experiments

C.1 Knowledge Editing Performance: ManiEdit vs. Memory-based Baselines

We supplement Table 1 with BERTScore, ROUGE-L, and ROUGE-1 (all in %) under the Original (Ori), Paraphrase (Para), and Sub-question (Sub) splits on UnKEBench, AKEW (Counterfact), and AKEW (MQUAKE), comparing memory-based baselines—SERAC, MELO, and GRACE—against ManiEdit on both Llama3-8B-Instruct and Qwen2.5-7B-Instruct. For each backbone and each metric, the highest value is in bold and the second-highest is underlined; detailed results are provided in Tables 2, 3, and 4.

The results reveal a consistent pattern that distinguishes memory-based retrieval from genuine knowledge internalization:

- **ManiEdit achieves the best Para scores across nearly all benchmarks, demonstrating genuine knowledge internalization.** Across BERTScore, ROUGE-L, and ROUGE-1 on the three benchmarks and both backbones (18 Para columns), ManiEdit ranks first in 17 columns. Relative to the

Method	Ori			Para			Sub	
	BERTScore	ROUGE-L	ROUGE-1	BERTScore	ROUGE-L	ROUGE-1	ROUGE-L	ROUGE-1
Llama3-8B-Instruct								
SERAC	70.15	27.89	29.73	70.44	26.91	28.88	22.03	22.71
MELO	69.58	25.19	27.13	70.04	25.26	27.26	21.98	22.66
GRACE	73.02	34.34	36.17	70.11	25.42	27.43	22.41	23.09
ManiEdit	**76.22**	**37.30**	**39.66**	**75.80**	**35.61**	**38.11**	**32.26**	**33.28**
Qwen2.5-7B-Instruct								
SERAC	71.73	30.93	33.14	70.89	29.15	31.50	24.18	24.99
MELO	71.26	28.51	30.86	70.48	27.53	29.95	24.19	24.98
GRACE	74.36	37.12	39.39	70.55	27.96	30.42	24.58	25.39
ManiEdit	**76.63**	**39.43**	**42.38**	**74.97**	**38.07**	**40.97**	**32.03**	**33.05**

Table 2: Comparison of ManiEdit with memory-based methods under the sequential unstructured editing setting on UnKEBench.

Method	Ori			Para			Sub	
	BERTScore	ROUGE-L	ROUGE-1	BERTScore	ROUGE-L	ROUGE-1	ROUGE-L	ROUGE-1
Llama3-8B-Instruct								
SERAC	69.22	23.20	24.88	42.97	15.25	16.58	37.36	38.35
MELO	69.69	20.30	22.22	52.09	19.11	21.04	34.42	35.42
GRACE	69.05	19.60	21.25	41.90	12.97	14.18	33.92	34.98
ManiEdit	**78.50**	**37.24**	**39.98**	**53.88**	**28.73**	**30.80**	**40.05**	**41.18**
Qwen2.5-7B-Instruct								
SERAC	69.78	21.76	23.67	52.00	19.98	21.96	36.00	36.93
MELO	68.69	18.59	20.24	41.91	12.98	14.19	33.59	34.68
GRACE	69.84	20.63	22.53	**52.07**	19.12	21.03	34.61	35.59
ManiEdit	**73.88**	**38.17**	**40.96**	50.78	**31.45**	**34.07**	**40.60**	**41.60**

Table 3: Comparison of ManiEdit with memory-based methods under the sequential unstructured editing setting on the AKEW (Counterfact) benchmark.

second-best memory-based method, the largest leads are $+5.36$ BERTScore, $+14.29$ ROUGE-L, and $+15.11$ ROUGE-1, with average leads of $+2.31$, $+11.19$, and $+11.75$, respectively. These Para gains indicate that ManiEdit internalizes edited knowledge rather than relying on external routing like memory-based methods.

- **On Sub-question evaluation, ManiEdit remains competitive while memory-based methods show inconsistent fine-grained recall of the edits.** SERAC leads Sub scores on MQUAKE Llama3. However, ManiEdit consistently leads Sub scores on UnKEBench and AKEW (Counterfact), with average leads of $+6.15$ ROUGE-L and $+6.34$ ROUGE-1 over the second-best memory-based method, reflecting more robust fine-grained recall of the edited knowledge.

C.2 Sub-Question Evaluation for Model Editing Methods

Section 4 reports the **Original** and **Paraphrase** splits in Table 1. Table 5 further reports **Sub-question (Sub)** results for model editing methods, using ROUGE-1 and ROUGE-L (both in %) on the same backbones and benchmarks. Sub questions probe fine-grained facets of the edited knowledge, providing a more targeted test of whether the update supports fine-grained factual recall beyond the main query. Each column's best is bold and the runner-up is underlined.

- **ManiEdit provides the most reliable fine-grained factual recall among all editors.** ManiEdit ranks first in 10 of 12 Sub columns. Across these 10 columns, ManiEdit outperforms the strongest baseline by $+6.82$ ROUGE-1 and $+6.59$ ROUGE-L on average. Overall, the Sub split confirms that ManiEdit improves not only direct and paraphrased recall, but also fine-grained recall of edited knowledge.

Method	Ori			Para			Sub	
	BERTScore	ROUGE-L	ROUGE-1	BERTScore	ROUGE-L	ROUGE-1	ROUGE-L	ROUGE-1
Llama3-8B-Instruct								
SERAC	69.97	19.94	21.50	69.75	21.20	23.54	**42.36**	**43.64**
MELO	69.52	17.00	18.54	69.78	21.14	23.48	39.95	41.35
GRACE	**81.82**	**52.57**	**53.37**	69.82	21.13	23.45	39.97	41.38
ManiEdit	71.51	30.03	32.57	**71.24**	**35.34**	**38.35**	32.70	34.05
Qwen2.5-7B-Instruct								
SERAC	67.67	22.72	24.84	68.35	20.79	23.14	42.89	44.00
MELO	68.27	19.62	21.75	68.37	20.68	23.02	41.62	42.73
GRACE	**80.56**	**48.33**	**49.75**	68.35	20.66	23.02	41.72	42.80
ManiEdit	71.37	37.96	41.15	**70.87**	**35.08**	**38.25**	**44.74**	**45.88**

Table 4: Comparison of ManiEdit with memory-based methods under the sequential unstructured editing setting on the AKEW (MQUAKE) benchmark.

Method	Qwen2.5-7B-Instruct						Llama3-8B-Instruct					
	UnKEBench		AKEW (MQUAKE)		AKEW (Counterfact)		UnKEBench		AKEW (MQUAKE)		AKEW (Counterfact)	
	ROUGE-1	ROUGE-L	ROUGE-1	ROUGE-L	ROUGE-1	ROUGE-L	ROUGE-1	ROUGE-L	ROUGE-1	ROUGE-L	ROUGE-1	ROUGE-L
MEMIT	16.54	16.35	23.06	22.82	26.57	25.80	1.39	1.35	10.33	10.00	13.09	12.80
RECT	15.52	15.31	26.10	25.58	24.39	24.10	15.00	14.63	17.52	17.38	16.19	16.13
UnKE	6.72	6.38	25.03	26.62	11.26	11.09	0.69	0.68	5.51	5.32	4.06	4.03
AnyEdit	15.63	15.29	21.63	21.14	21.82	21.37	15.84	15.62	21.01	20.43	13.22	12.98
EvoEdit	14.72	14.34	22.69	21.98	26.72	26.11	11.98	11.79	16.20	15.71	18.22	17.84
MOSE	28.63	27.39	37.81	37.16	31.13	30.28	27.58	26.88	**45.09**	**44.41**	35.75	35.00
ManiEdit	**33.05**	**32.03**	**45.88**	**44.74**	**41.60**	**40.60**	**33.28**	**32.26**	34.05	32.70	**41.18**	**40.05**

Table 5: Comparison of ManiEdit with existing methods on the sequential unstructured model editing tasks under the Sub-question split. The best results are highlighted in bold, while the second-best results are underlined.

C.3 General Capability Preservation

To complement the three GLUE benchmarks reported in Section 4 (SST-2, CoLA, MRPC), we conduct the same 500-edit sequential stability experiment on three additional benchmarks: **MMLU** (Hendrycks et al., 2021), **MNLI** (Williams et al., 2018), and **RTE** (Bentivogli et al., 2009). All experiments use Qwen2.5-7B-Instruct and follow the same evaluation protocol as Section 4: accuracy is measured at intervals of 100 cumulative edits against MOSE, AnyEdit, and RECT.

Figure 6 shows the accuracy curves. The findings are consistent with the main-paper results:

- **AnyEdit and RECT collapse within 100 edits** on all three benchmarks. On MNLI and RTE, both methods drop to 0.0% accuracy after only 100 edits and remain at zero throughout. On MMLU, RECT falls from 70.4% to 9.3%, and AnyEdit reaches 9.5% by edit 100.
- **MOSE collapses abruptly and erratically.** On MMLU and MNLI it drops to near-zero within 100 edits, with highly unstable fluctuations across subsequent checkpoints rather than any consistent recovery. On RTE, accuracy declines steadily from the pre-edit 81.2% to 45.5% at 200 edits, and further to 4.0% at 500 edits.
- **ManiEdit maintains stable performance** throughout the full 500-edit sequence. On MMLU it declines by only ≈5.5% (70.4%→64.9%); on MNLI it stays within 5.4% of the original (82.8%→77.4%); on RTE it stays near the pre-edit accuracy and finishes slightly higher (81.2%→81.9%). This broad consistency across diverse benchmarks confirms that ManiEdit's differential constraint design provides robust protection of pre-trained capabilities under extended sequential editing.

C.4 Cross-Domain Editing Performance on EditEverything

Figure 7 presents per-domain BERTScore and ROUGE-L on the EditEverything benchmark (Jiang et al., 2025) across six specialized knowledge categories—Math, Code, Poetry, Chemistry, News, and Biology—on both Qwen2.5-7B-Instruct and Llama3-8B-Instruct.

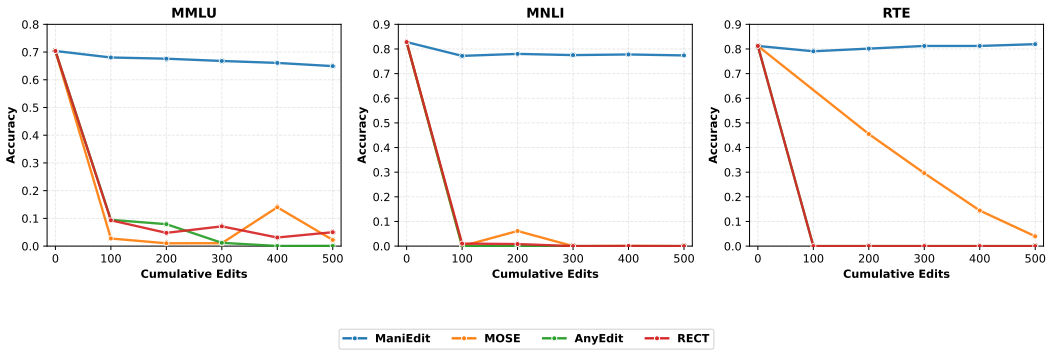

Figure 6: General capability accuracy on MMLU, MNLI, and RTE as the number of sequential unstructured edits increases. ManiEdit maintains near-original accuracy across all tasks, while baselines collapse rapidly.

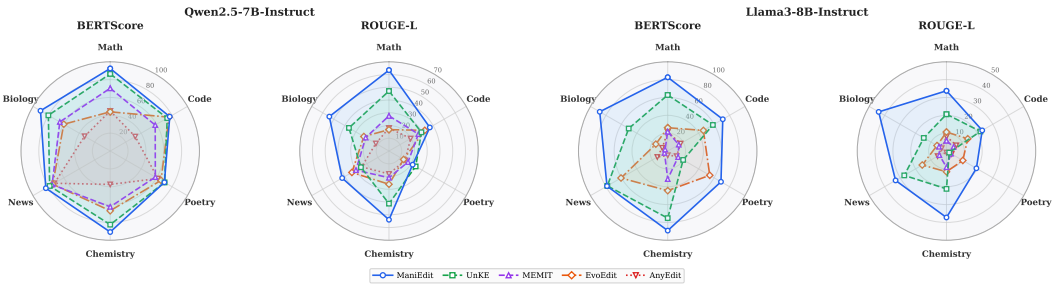

Figure 7: Radar plots of per-domain editing performance on EditEverything. From left to right: BERTScore and ROUGE-L (%) on Qwen2.5-7B-Instruct, then BERTScore and ROUGE-L (%) on Llama3-8B-Instruct. Each axis is one domain; a larger enclosed area indicates better overall performance.

The radar plots reveal several consistent cross-domain patterns:

- **ManiEdit spans the largest radar area across all six domains on both models.** On Qwen2.5-7B-Instruct, ManiEdit achieves the highest BERTScore and ROUGE-L in five of six domains: Math, Chemistry, Biology, News, and Code, with the sole exception of Poetry where UnKE achieves a marginally higher score. On Llama3-8B-Instruct, ManiEdit leads all six domains unambiguously. Averaged across domains, ManiEdit outperforms the second-best method by approximately 16 BERTScore points on Llama3 and 5 BERTScore points on Qwen2.5.
- **Scientific domains yield high editing accuracy; open-ended domains remain challenging.** Across all methods and both backbones, Math, Chemistry, and Biology consistently attain higher scores than Poetry, Code, and News. The regularity and self-consistency of mathematical and scientific knowledge facilitate accurate weight-level encoding. Poetry records the lowest scores for all methods—reflecting the stylistic variability of open-ended text and the difficulty of producing surface-faithful completions. However, ManiEdit still maintains 70.13 (Qwen2.5) and 68.91 (Llama3) BERTScore even in this hardest category.

C.5 Analysis of Pivot Localization under the Original-Query Setting

Figure 8 complements the Para-query analysis in Section 4.4 by reporting Ori ROUGE-L under the same six backbone–benchmark settings. This view tests whether Pivot Localization also improves direct editing efficacy when the query exactly matches the edited knowledge.

The Ori-query results reinforce the plug-in value of Pivot Localization for locate-then-edit methods. Adding Pivot Localization to EvoEdit improves ROUGE-L in all six settings, with especially large gains on Qwen-UnKEBench (16.08 $\rightarrow$ 28.93), Qwen-MQUAKE (26.35 $\rightarrow$ 32.50), and Llama-

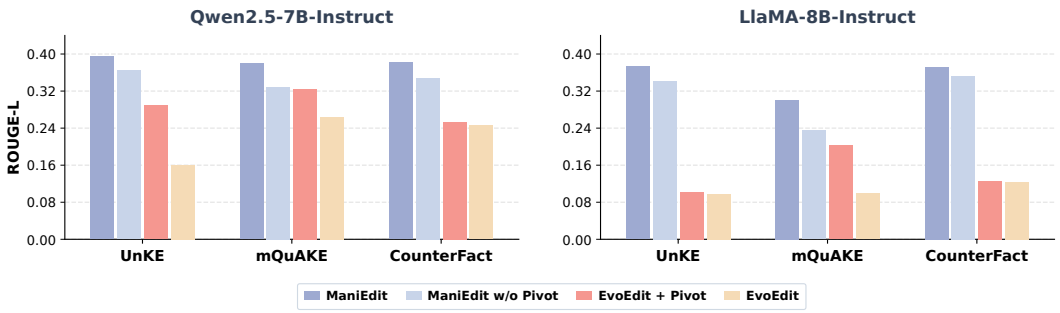

Figure 8: Analysis of Pivot Localization under the Original-Query setting. Scores are Ori ROUGE-L on two base LLMs across three benchmarks. ManiEdit w/o Pivot and EvoEdit use the AnyEdit chunking protocol.

Backbone	Benchmark	General-Capability Preservation	Edit-Knowledge Preservation	Original		Para	
				BERTScore	ROUGE-L	BERTScore	ROUGE-L
Qwen2.5-7B-Instruct	UnKEBench	Null-space Projection	Recursive Null-space Alignment	43.69	24.12	41.85	23.64
		Energy-weighted Penalty	Regularization	46.73	29.14	46.20	28.46
		Null-space Projection	Regularization	37.08	30.90	37.99	31.45
		Energy-weighted Penalty	Recursive Null-space Alignment	**76.63**	**39.43**	**74.97**	**38.07**
	AKEW (Counterfact)	Null-space Projection	Recursive Null-space Alignment	45.76	27.54	30.49	24.60
		Energy-weighted Penalty	Regularization	62.26	32.93	50.33	30.52
		Null-space Projection	Regularization	48.74	31.25	44.40	30.03
		Energy-weighted Penalty	Recursive Null-space Alignment	**73.88**	**38.17**	**50.78**	**31.45**
	AKEW (MQUAKE)	Null-space Projection	Recursive Null-space Alignment	64.68	36.56	60.66	30.54
		Energy-weighted Penalty	Regularization	64.57	33.84	62.77	30.72
		Null-space Projection	Regularization	36.97	32.53	38.90	31.50
		Energy-weighted Penalty	Recursive Null-space Alignment	**71.37**	**37.96**	**70.87**	**35.08**
Llama3-8B-Instruct	UnKEBench	Null-space Projection	Recursive Null-space Alignment	8.17	9.13	7.18	9.24
		Energy-weighted Penalty	Regularization	3.95	6.40	4.19	6.83
		Null-space Projection	Regularization	0.00	7.72	0.37	8.33
		Energy-weighted Penalty	Recursive Null-space Alignment	**76.22**	**37.30**	**75.80**	**35.61**
	AKEW (Counterfact)	Null-space Projection	Recursive Null-space Alignment	13.63	11.14	5.78	9.74
		Energy-weighted Penalty	Regularization	6.04	3.81	4.10	4.74
		Null-space Projection	Regularization	9.07	7.12	6.65	8.07
		Energy-weighted Penalty	Recursive Null-space Alignment	**78.50**	**37.24**	**53.88**	**28.73**
	AKEW (MQUAKE)	Null-space Projection	Recursive Null-space Alignment	52.25	19.60	49.53	19.54
		Energy-weighted Penalty	Regularization	3.40	3.09	1.93	1.99
		Null-space Projection	Regularization	5.64	7.32	7.52	10.53
		Energy-weighted Penalty	Recursive Null-space Alignment	**71.51**	**30.03**	**71.24**	**35.34**

Table 6: Ablation of Manifold-Aware Preservation components on UnKEBench, AKEW (Counterfact), and AKEW (MQUAKE). The left component in each pair protects general pre-trained capabilities, and the right component protects sequentially edited knowledge.

MQUAKE ($9.89 \rightarrow 20.23$). This suggests that locating high-leverage semantic pivots can directly improve the quality of the update target.

For ManiEdit, removing Pivot Localization lowers ROUGE-L in every setting; the full model improves over the no-pivot variant by $+3.83$ ROUGE-L on average, ranging from $+1.92$ on Llama-Counterfact to $+6.54$ on Llama-MQUAKE. Together, the Ori and Para analyses show that Pivot Localization improves both paraphrase robustness and original-query effectiveness in every setting.

C.6 Ablation of Manifold-Aware Preservation Components

To analyze the two protection mechanisms in Manifold-Aware Preservation, Table 6 compares four combinations of constraints. The left component protects general pre-trained capabilities and the right component preserves sequentially edited knowledge; ManiEdit is the *Energy-weighted Penalty + Recursive Null-space Alignment* combination.

We instantiate the four combinations as follows. Let K_0 collect keys of preserved pre-trained knowledge. Following AlphaEdit (Fang et al., 2025), we take $P^{\mathrm{NS}} = \hat{U}\hat{U}^\top$, where $\hat{U}$ retains the eigenvectors of $K_0 K_0^\top$ associated with zero eigenvalues (in practice, those below a small threshold), so that P^{NS} maps the update into the null space of K_0. Let $K_{<t}$ collect all keys of edits $i < t$, and let P_{t-1} be the recursive null-space alignment of Equation 8.

Setting	MOSE				ManiEdit			
	Ori		Para		Ori		Para	
	BERTScore	ROUGE-L	BERTScore	ROUGE-L	BERTScore	ROUGE-L	BERTScore	ROUGE-L
@500	62.38	31.76	61.11	30.95	76.22	37.30	75.80	35.61
@1,000	54.65	20.57	55.56	19.80	77.00	34.99	75.66	34.61
Δ	-7.73	-11.19	-5.55	-11.15	$+0.78$	-2.31	-0.14	-1.00

Table 7: Long-horizon stress test on Llama3-8B-Instruct with UnKEBench (scores in %). Δ denotes the change from 500 to 1,000 sequential edits.

Null-space Projection + Recursive Null-space Alignment. Initialize $P_0 = P^{\mathrm{NS}}$ and recursively update P_{t-1} by Equation 8, so both pre-trained and historical keys are protected by the null space:

$$\min_{\Delta_t} \left\|(W_{t-1} + \Delta_t P_{t-1})K_t - V_t^*\right\|_F^2 + \lambda\|\Delta_t\|_F^2. \tag{11}$$

Energy-weighted Penalty + Regularization. The energy-weighted penalty $\lambda\|\Delta_t\Sigma^{1/2}\|_F^2$ preserves pre-trained geometry, and $\|\Delta_t K_{<t}\|_F^2$ limits the output displacement of historical keys:

$$\min_{\Delta_t} \left\|(W_{t-1} + \Delta_t)K_t - V_t^*\right\|_F^2 + \lambda\|\Delta_t\Sigma^{1/2}\|_F^2 + \beta\|\Delta_t K_{<t}\|_F^2. \tag{12}$$

Null-space Projection + Regularization. P^{NS} preserves pre-trained keys, while historical edits are protected by a soft penalty:

$$\min_{\Delta_t} \left\|(W_{t-1} + \Delta_t P^{\mathrm{NS}})K_t - V_t^*\right\|_F^2 + \beta\|\Delta_t K_{<t}\|_F^2. \tag{13}$$

Energy-weighted Penalty + Recursive Null-space Alignment. Initialize $P_0 = I$ and recursively update P_{t-1} by Equation 8. The energy-weighted penalty preserves pre-trained geometry, and P_{t-1} protects historical edits:

$$\min_{\Delta_t} \left\|(W_{t-1} + \Delta_t P_{t-1})K_t - V_t^*\right\|_F^2 + \lambda\|\Delta_t\Sigma^{1/2}\|_F^2. \tag{14}$$

The results show that neither purely soft regularization nor a uniform projection reliably protects both knowledge types. Replacing either side of ManiEdit's asymmetric design substantially degrades editing quality in all settings, especially on Llama3-8B-Instruct. The full combination of energy-weighted preservation for the pre-trained manifold and recursive null-space alignment for historical edits achieves optimal performance.

C.7 Long-Horizon Sequential Editing Stress Test

To further assess robustness under longer edit sequences, we extend the editing horizon from 500 edits to 1,000 edits on UnKEBench. Table 7 reports a comparison against MOSE on Llama3-8B-Instruct with UnKEBench; the percentage-point change Δ quantifies how much editing performance drifts as the sequence length doubles.

ManiEdit maintains its effectiveness under long-horizon editing. After doubling the edit horizon, ManiEdit's Para and Ori ROUGE-L drop by merely 1.00 and 2.31, whereas MOSE declines sharply by over 11 ROUGE-L points on both Para and Ori. This demonstrates ManiEdit's strong robustness under long-horizon unstructured editing.

C.8 Preservation of Historical Edits via Recursive Null-space Alignment

The closed-form solution (Equation 7) can be applied in two ways: $W_t = W_{t-1} + \Delta_t$ or $W_t = W_{t-1} + \Delta_t P_{t-1}$. This paper adopts the former, so that P_{t-1} in the optimization objective (Equation 6) guides Δ_t toward the orthogonal complement of $\mathrm{span}\{K_i\}_{i<t}$, driving $\Delta_t K_i$ toward zero. This

Update	$\mathrm{leak}_t \downarrow$			$\mathrm{cum}_t \downarrow$
	Median	Mean	Max	Median
Without alignment ($P_{t-1} = I$)	9.02	8.62	16.0	8.72
ManiEdit	0.305	0.343	2.09	0.297
Reduction (median)		$29.6\times$		$29.4\times$

Table 8: leak_t and cum_t over 500 sequential edits, with and without recursive null-space alignment.

section empirically shows that this constraint substantially reduces the interference of a new update with historical edit keys, thereby preserving historical edits.

Writing $\Delta_t = R_t F_t$ with $F_t := (\lambda I + G_t)^{-1} A_t^\top \Sigma^{-1}$, $A_t := P_{t-1} K_t$, and $G_t := A_t^\top \Sigma^{-1} A_t$ (Appendix E.2), the residual R_t enters every response linearly, so we evaluate the update through F_t. Since F_t depends on P_{t-1}, we compute P_{t-1} with $\epsilon = 0$ in line with Proposition 3, and compare against a baseline that replaces P_{t-1} by I (i.e., the same energy-weighted update without recursive null-space alignment). We report

$$\mathrm{leak}_t = \frac{\|F_t K_{<t}\|_F}{\|F_t K_t\|_F}, \qquad \mathrm{cum}_t = \frac{\left(\sum_{s>t} \|F_s K_t\|_F^2\right)^{1/2}}{\|F_t K_t\|_F},$$

where $K_{<t}$ collects all keys of edits $i < t$. Thus leak_t measures how strongly the t-th update applies its residual to all previous edit keys, relative to its own keys; cum_t measures how strongly later updates apply their residuals to edit t, relative to that edit's own update. This measurement uses the second-moment matrix $\Sigma \in \mathbb{R}^{n\times n}$ ($n = 18{,}944$) in the key space of the layer-8 FFN down-projection matrix in Qwen2.5-7B-Instruct, together with the edit keys of the 500 UnKEBench samples (1,427 keys in total).

Table 8 shows that without alignment, the median leak_t is 9.02 and the median cum_t is 8.72. With alignment, these drop to 0.305 and 0.297, a reduction of about $30\times$ in both. These results show that recursive null-space alignment effectively guides Δ_t toward the orthogonal complement of $\mathrm{span}\{K_i\}_{i<t}$, thereby preserving historical edits against both a new update and those that follow.

D Derivation of the Closed-Form Solution

In this section, we provide the rigorous mathematical derivation for the closed-form solution of our unified optimization objective.

D.1 Problem Formulation

At editing step t, we seek to find the optimal weight increment $\Delta_t \in \mathbb{R}^{m\times n}$ that minimizes the following objective function:

$$\begin{aligned}\min_{\Delta_t} \mathcal{J}(\Delta_t) := &\|(W_{t-1} + \Delta_t P_{t-1}) K_t - V_t^*\|_F^2 \\ &+ \lambda \left\|\Delta_t \Sigma^{1/2}\right\|_F^2,\end{aligned} \tag{15}$$

where:

- $K_t \in \mathbb{R}^{n\times b}$ is the current input key matrix (representing a batch of b keys).
- $V_t^* \in \mathbb{R}^{m\times b}$ is the target output matrix.
- $P_{t-1} \in \mathbb{R}^{n\times n}$ is a symmetric matrix ($P_{t-1}^\top = P_{t-1}$).
- $\Sigma \in \mathbb{R}^{n\times n}$ is the symmetric positive-definite feature second-moment matrix.
- $\lambda > 0$ is the regularization coefficient.

Let $R_t := V_t^* - W_{t-1} K_t \in \mathbb{R}^{m\times b}$ denote the current residual matrix. The objective simplifies to:

$$\mathcal{J}(\Delta_t) = \|\Delta_t P_{t-1} K_t - R_t\|_F^2 + \lambda \left\|\Delta_t \Sigma^{1/2}\right\|_F^2. \tag{16}$$

D.2 Matrix Trace Reformulation

To facilitate differentiation, we rewrite the objective using matrix trace properties: $\|A\|_F^2 = \mathrm{Tr}(AA^\top)$ for a matrix A. Expanding the mapping loss term:

$$\begin{aligned}
&\|\Delta_t P_{t-1} K_t - R_t\|_F^2 \\
&= \mathrm{Tr}\left((\Delta_t P_{t-1} K_t - R_t)(\Delta_t P_{t-1} K_t - R_t)^\top\right) \\
&= \mathrm{Tr}\left(\Delta_t P_{t-1} K_t K_t^\top P_{t-1}^\top \Delta_t^\top - R_t K_t^\top P_{t-1}^\top \Delta_t^\top\right. \\
&\quad \left. -\Delta_t P_{t-1} K_t R_t^\top + R_t R_t^\top\right).
\end{aligned} \tag{17}$$

Since $\mathrm{Tr}(A) = \mathrm{Tr}(A^\top)$ and P_{t-1} is symmetric, we have $\mathrm{Tr}(R_t K_t^\top P_{t-1} \Delta_t^\top) = \mathrm{Tr}(\Delta_t P_{t-1} K_t R_t^\top)$. The mapping loss becomes:

$$\begin{aligned}
&\mathrm{Tr}\left(\Delta_t (P_{t-1} K_t K_t^\top P_{t-1}) \Delta_t^\top\right) \\
&\quad - 2\,\mathrm{Tr}\left(R_t K_t^\top P_{t-1} \Delta_t^\top\right) \\
&\quad + \mathrm{Tr}(R_t R_t^\top).
\end{aligned} \tag{18}$$

Expanding the regularization term:

$$\begin{aligned}
\lambda \left\|\Delta_t \Sigma^{1/2}\right\|_F^2 &= \lambda\,\mathrm{Tr}\left(\Delta_t \Sigma^{1/2} (\Sigma^{1/2})^\top \Delta_t^\top\right) \\
&= \lambda\,\mathrm{Tr}\left(\Delta_t \Sigma \Delta_t^\top\right).
\end{aligned} \tag{19}$$

Ignoring the constant term $\mathrm{Tr}(R_t R_t^\top)$ which does not depend on Δ_t, the objective function becomes:

$$\begin{aligned}
\mathcal{J}(\Delta_t) = &\mathrm{Tr}\left(\Delta_t (P_{t-1} K_t K_t^\top P_{t-1}) \Delta_t^\top\right) \\
&- 2\,\mathrm{Tr}\left(R_t K_t^\top P_{t-1} \Delta_t^\top\right) \\
&+ \lambda\,\mathrm{Tr}\left(\Delta_t \Sigma \Delta_t^\top\right).
\end{aligned} \tag{20}$$

D.3 Optimization and Closed-Form Solution

We compute the gradient of $\mathcal{J}(\Delta_t)$ with respect to Δ_t. Using the matrix calculus identities $\frac{\partial\,\mathrm{Tr}(XCX^\top)}{\partial X} = X(C + C^\top)$ and $\frac{\partial\,\mathrm{Tr}(DX^\top)}{\partial X} = D$, and noting that both $(P_{t-1} K_t K_t^\top P_{t-1})$ and Σ are symmetric:

$$\begin{aligned}
\frac{\partial \mathcal{J}(\Delta_t)}{\partial \Delta_t} &= 2\Delta_t (P_{t-1} K_t K_t^\top P_{t-1}) \\
&\quad - 2R_t K_t^\top P_{t-1} + 2\lambda \Delta_t \Sigma.
\end{aligned} \tag{21}$$

Setting the gradient to zero to find the optimal Δ_t:

$$\Delta_t (P_{t-1} K_t K_t^\top P_{t-1}) + \lambda \Delta_t \Sigma = R_t K_t^\top P_{t-1}. \tag{22}$$

Factoring out Δ_t on the left yields:

$$\Delta_t \left(P_{t-1} K_t K_t^\top P_{t-1} + \lambda \Sigma\right) = R_t K_t^\top P_{t-1}. \tag{23}$$

Let $M_t := P_{t-1} K_t K_t^\top P_{t-1} + \lambda \Sigma$. Since $P_{t-1} K_t K_t^\top P_{t-1}$ is positive semi-definite and $\lambda\Sigma$ is strictly positive definite, their sum M_t is strictly positive definite and thus invertible. Right-multiplying by M_t^{-1} gives the final closed-form solution:

$$\Delta_t = R_t K_t^\top P_{t-1} \left(P_{t-1} K_t K_t^\top P_{t-1} + \lambda \Sigma\right)^{-1}. \tag{24}$$

E Proofs of Theoretical Propositions

In this section we provide rigorous proofs of the propositions stated in Section 3.5, thereby establishing the theoretical advantages of our method.

E.1 Proof of Proposition 1

We consider the constrained problem

$$\min_{\Delta \in \mathbb{R}^{m\times n}} \ \|\Delta\, \Sigma^{1/2}\|_F^2 \quad \text{s.t.} \quad \Delta\, k_t = R. \tag{25}$$

Since $\|\Delta\Sigma^{1/2}\|_F^2 = \mathrm{Tr}(\Delta\Sigma\Delta^\top) = \mathbb{E}_k\big[\|\Delta k\|^2\big]$, where k follows the distribution of pre-trained keys from which Σ is estimated, Equation 25 measures the minimum perturbation of pre-trained outputs required to realize the output displacement R at key k_t.

Introducing a vector Lagrange multiplier $\mu \in \mathbb{R}^m$,

$$\mathcal{L}(\Delta, \mu) = \|\Delta\, \Sigma^{1/2}\|_F^2 - 2\,\mu^\top\big(\Delta\, k_t - R\big). \tag{26}$$

Setting $\partial\mathcal{L}/\partial\Delta = 2\Delta\,\Sigma - 2\,\mu\, k_t^\top = 0$ gives

$$\Delta^\star = \mu\, k_t^\top \Sigma^{-1}. \tag{27}$$

Substituting into the constraint $\Delta^\star k_t = R$,

$$\mu\,(k_t^\top \Sigma^{-1} k_t) = R \quad \Longrightarrow \quad \mu = \frac{R}{k_t^\top \Sigma^{-1} k_t}. \tag{28}$$

The optimal update is therefore

$$\Delta^\star = \frac{R\, k_t^\top \Sigma^{-1}}{k_t^\top \Sigma^{-1} k_t}. \tag{29}$$

Plugging this back into the objective and using $\mathrm{Tr}(R^\top R) = \|R\|^2$ together with $k_t^\top \Sigma^{-1}\Sigma\,\Sigma^{-1} k_t = k_t^\top \Sigma^{-1} k_t$,

$$\begin{aligned} \|\Delta^\star \Sigma^{1/2}\|_F^2 &= \mathrm{Tr}\big(\Sigma^{1/2}(\Delta^\star)^\top \Delta^\star \Sigma^{1/2}\big) \\ &= \frac{\mathrm{Tr}\big(R^\top R\big)\, k_t^\top \Sigma^{-1} k_t}{(k_t^\top \Sigma^{-1} k_t)^2} \\ &= \frac{\|R\|^2}{k_t^\top \Sigma^{-1} k_t}. \end{aligned} \tag{30}$$

The function $L \mapsto \|R\|^2/L$ is strictly decreasing on $L > 0$, so the minimum energy-weighted cost is a strictly monotonically decreasing function of the leverage score $k_t^\top \Sigma^{-1} k_t$. This rigorously formalizes the intuition that high-leverage semantic pivots realize the target displacement with lower unintended perturbation, enabling precise editing at low cost.

E.2 Proof of Proposition 2

We prove Proposition 2 in two steps. We first reduce the closed-form solution (Equation 7) to a single-key spectral form via the Sherman–Morrison identity, then project the result into the eigenbasis of Σ to read off the per-direction update allocation $\Delta W_{\text{dir}\ j} \propto \tilde{k}_j/\lambda_j$.

Single-Key Reduction via Sherman–Morrison. We begin from the closed-form solution derived in Appendix D:

$$\Delta_t = R_t K_t^\top P_{t-1}\big(P_{t-1} K_t K_t^\top P_{t-1} + \lambda\Sigma\big)^{-1}. \tag{31}$$

Consider the single-key setting where K_t collapses to a column vector $k_t \in \mathbb{R}^n$. Using the symmetry $P_{t-1}^\top = P_{t-1}$ and defining the projected key $\hat{k}_t := P_{t-1} k_t$, we have

$$K_t^\top P_{t-1} = \hat{k}_t^\top, \quad P_{t-1} K_t K_t^\top P_{t-1} = \hat{k}_t \hat{k}_t^\top, \tag{32}$$

so the closed form reduces to

$$\Delta_t = R_t \hat{k}_t^\top \big(\hat{k}_t \hat{k}_t^\top + \lambda\Sigma\big)^{-1}. \tag{33}$$

The inverse in Equation 33 can be evaluated analytically via the **Sherman–Morrison identity**: for any invertible A and vectors u, v,

$$(A + uv^\top)^{-1} = A^{-1} - \frac{A^{-1} u v^\top A^{-1}}{1 + v^\top A^{-1} u}. \tag{34}$$

Setting $A = \lambda\Sigma$ and $u = v = \hat{k}_t$, and noting $(\lambda\Sigma)^{-1} = \lambda^{-1}\Sigma^{-1}$,

$$\begin{aligned} &\left(\lambda\Sigma + \hat{k}_t\hat{k}_t^\top\right)^{-1} \\ &= \tfrac{1}{\lambda}\Sigma^{-1} - \frac{\frac{1}{\lambda^2}\Sigma^{-1}\hat{k}_t\hat{k}_t^\top\Sigma^{-1}}{1 + \frac{1}{\lambda}\hat{k}_t^\top\Sigma^{-1}\hat{k}_t}. \end{aligned} \tag{35}$$

Left-multiplying by $\hat{k}_t^\top$ and introducing the scalar $s := \hat{k}_t^\top\Sigma^{-1}\hat{k}_t$,

$$\begin{aligned} &\hat{k}_t^\top\left(\hat{k}_t\hat{k}_t^\top + \lambda\Sigma\right)^{-1} \\ &= \frac{\hat{k}_t^\top\Sigma^{-1}}{\lambda} - \frac{\frac{s}{\lambda^2}\hat{k}_t^\top\Sigma^{-1}}{1 + \frac{s}{\lambda}} \\ &= \frac{\hat{k}_t^\top\Sigma^{-1}}{\lambda}\left(1 - \frac{s/\lambda}{1 + s/\lambda}\right) \\ &= \frac{\hat{k}_t^\top\Sigma^{-1}}{\lambda + s}. \end{aligned} \tag{36}$$

Multiplying by R_t on the left and substituting $s = \hat{k}_t^\top\Sigma^{-1}\hat{k}_t$ yields the compact spectral form:

$$\Delta_t = \frac{R_t\hat{k}_t^\top\Sigma^{-1}}{\lambda + \hat{k}_t^\top\Sigma^{-1}\hat{k}_t}. \tag{37}$$

Per-Direction Allocation in the Eigenbasis of Σ. We now project Equation 33 into the eigenspace of Σ to derive the per-direction update allocation. Since Σ is symmetric positive-definite, it admits the eigendecomposition $\Sigma = \Psi\Lambda\Psi^\top$ with $\Psi = [\psi_1, \ldots, \psi_n]$ an orthonormal basis and $\Lambda = \mathrm{diag}(\lambda_1, \ldots, \lambda_n)$. Correspondingly,

$$\Sigma^{-1} = \sum_{l=1}^{n} \frac{1}{\lambda_l}\psi_l\psi_l^\top. \tag{38}$$

Defining $\tilde{k}_l := \psi_l^\top\hat{k}_t$ as the projection of the effective key onto the l-th principal direction, substituting into the vector factor $\hat{k}_t^\top\Sigma^{-1}$ gives

$$\hat{k}_t^\top\Sigma^{-1} = \sum_{l=1}^{n} \frac{\hat{k}_t^\top\psi_l}{\lambda_l}\psi_l^\top = \sum_{l=1}^{n} \frac{\tilde{k}_l}{\lambda_l}\psi_l^\top. \tag{39}$$

To read off the contribution along the j-th principal direction, we right-multiply Equation 39 by ψ_j and apply the orthonormality $\psi_l^\top\psi_j = \delta_{lj}$:

$$\left(\hat{k}_t^\top\Sigma^{-1}\right)\psi_j = \sum_{l=1}^{n} \frac{\tilde{k}_l}{\lambda_l}(\psi_l^\top\psi_j) = \frac{\tilde{k}_j}{\lambda_j}. \tag{40}$$

Since the scalar prefactor $R_t/(\lambda + s)$ in Equation 37 is independent of j, the update magnitude along ψ_j is proportional to

$$\Delta W_{\text{dir } j} \propto \frac{\tilde{k}_j}{\lambda_j}. \tag{41}$$

The numerator $\tilde{k}_j$ quantifies the *semantic demand* of the edit along direction ψ_j, while the denominator λ_j reflects the *inertia* (statistical energy) of the pre-trained manifold along that direction. Their ratio formalizes the principle of *impedance-matched editing*: editing energy is channeled into directions that are simultaneously relevant to the target knowledge and structurally under-utilized by the pre-trained model. This establishes Proposition 2: the closed-form update is energy-adaptive in the eigenbasis of Σ, allocating more capacity to under-utilized low-energy directions and less to dominant high-energy directions.

Remark: the general multi-key case. The single-key assumption yields the scalar form above, but it is not essential for the energy-adaptive behavior. For a general key matrix, write $A_t := P_{t-1}K_t \in \mathbb{R}^{n\times b}$, so that the closed-form solution reads $\Delta_t = R_t A_t^\top (A_t A_t^\top + \lambda\Sigma)^{-1}$. Applying the Woodbury identity to $(\lambda\Sigma + A_t A_t^\top)^{-1}$ and using $I - G_t(\lambda I + G_t)^{-1} = \lambda(\lambda I + G_t)^{-1}$ with $G_t := A_t^\top \Sigma^{-1} A_t$ gives

$$\Delta_t = R_t\big(\lambda I + G_t\big)^{-1} A_t^\top \Sigma^{-1}, \tag{42}$$

which reduces to Equation 37 when $b = 1$. Since $\Sigma^{-1}\psi_j = \psi_j/\lambda_j$, the per-direction allocation becomes

$$\Delta_t \psi_j = \frac{1}{\lambda_j} R_t\big(\lambda I + G_t\big)^{-1} A_t^\top \psi_j, \tag{43}$$

where the matrix factor $R_t(\lambda I + G_t)^{-1}$ does not depend on j. The $1/\lambda_j$ scaling therefore persists in the general case, with the scalar semantic demand $\tilde{k}_j$ replaced by the vector $A_t^\top \psi_j$.

E.3 Proof of Proposition 3

Recursive Definition of P_t. Setting $\epsilon = 0$ and replacing the inverse with the Moore–Penrose pseudo-inverse, the recursion reads

$$P_t = P_{t-1} - P_{t-1}K_t\big(K_t^\top P_{t-1} K_t\big)^+ K_t^\top P_{t-1}, \qquad P_0 = I. \tag{44}$$

The pseudo-inverse coincides with the ordinary inverse whenever $K_t^\top P_{t-1} K_t$ is invertible. Proposition 3 concerns this idealized recursion; in practice we use $\epsilon = 10^{-6}$ to keep the matrix invertible.

Proof of Proposition 3. We prove by induction on $t \geq 0$ that P_t is symmetric and idempotent and that $P_t K_i = 0$ for all $i \leq t$. In the base case, $P_0 = I$ is symmetric and idempotent, and the condition on historical keys is vacuous.

Inductive step. Assume the claim holds for $t-1$ and write $A_t := P_{t-1}K_t$ and $\Pi_t := A_t(A_t^\top A_t)^+ A_t^\top$. Since P_{t-1} is symmetric and idempotent, $K_t^\top P_{t-1} K_t = A_t^\top A_t$ and $P_{t-1}K_t(K_t^\top P_{t-1}K_t)^+ K_t^\top P_{t-1} = \Pi_t$, so $P_t = P_{t-1} - \Pi_t$. By the properties of the pseudo-inverse, Π_t is the orthogonal projector onto $\mathrm{range}(A_t)$, and hence $\Pi_t A_t = A_t$.

Projector property. Since $P_{t-1}A_t = P_{t-1}^2 K_t = A_t$, we have $P_{t-1}\Pi_t = \Pi_t$ and, by symmetry, $\Pi_t P_{t-1} = \Pi_t$. Therefore $P_t^\top = P_t$ and $P_t^2 = P_{t-1}^2 - P_{t-1}\Pi_t - \Pi_t P_{t-1} + \Pi_t^2 = P_{t-1} - \Pi_t = P_t$.

Historical keys ($i < t$). By the inductive hypothesis, $P_{t-1}K_i = 0$ and $A_t^\top K_i = K_t^\top P_{t-1} K_i = 0$. Hence $P_t K_i = P_{t-1}K_i - A_t(A_t^\top A_t)^+ A_t^\top K_i = 0$.

Current key ($i = t$). Since $A_t^\top K_t = K_t^\top P_{t-1} K_t = A_t^\top A_t$, we have $\Pi_t K_t = A_t(A_t^\top A_t)^+ A_t^\top A_t = \Pi_t A_t = A_t$. Hence $P_t K_t = P_{t-1}K_t - \Pi_t K_t = A_t - A_t = 0$.

This completes the induction and proves (i).

Rank bound (ii). Since $\mathrm{rank}(\Pi_t) = \mathrm{rank}(A_t) \leq \mathrm{rank}(K_t)$, the subadditivity of rank gives $\mathrm{rank}(P_t) \geq \mathrm{rank}(P_{t-1}) - \mathrm{rank}(K_t)$. Unrolling this inequality from $\mathrm{rank}(P_0) = n$ yields $\mathrm{rank}(P_t) \geq n - \sum_{i=1}^t \mathrm{rank}(K_i)$. □

Corollary 1. *Let $S_{t-1} := \mathrm{span}\{K_i\}_{i<t}$ and adopt the setting of Proposition 3 ($\epsilon = 0$). Then the deployed update $\Delta_t = R_t F_t$ of Equation 42 satisfies*

$$\mathrm{range}(\Delta_t^\top) \;\subseteq\; \Sigma^{-1}\big(S_{t-1}^\perp\big) \;=\; \big(\Sigma\, S_{t-1}\big)^\perp, \tag{45}$$

and its interference with any historical key block K_i ($i < t$) is given exactly by

$$\Delta_t K_i \;=\; R_t(\lambda I + G_t)^{-1}\, K_t^\top \big[P_{t-1},\, \Sigma^{-1}\big] K_i, \tag{46}$$

where $[X, Y] := XY - YX$. Consequently, if S_{t-1} is Σ-invariant (equivalently, spanned by eigenvectors of Σ) then $\Delta_t K_i = 0$ exactly.

Proof. Since $G_t = A_t^\top \Sigma^{-1} A_t$ is symmetric, so is $(\lambda I + G_t)^{-1}$, and therefore $F_t^\top = \Sigma^{-1} A_t (\lambda I + G_t)^{-1}$. From $\Delta_t = R_t F_t$ we get $\Delta_t^\top = F_t^\top R_t^\top$, hence $\mathrm{range}(\Delta_t^\top) \subseteq \mathrm{range}(F_t^\top) \subseteq$

$\Sigma^{-1}\,\mathrm{range}(A_t)$. As $A_t = P_{t-1}K_t$ we have $\mathrm{range}(A_t) \subseteq \mathrm{range}(P_{t-1})$. Moreover P_{t-1} is symmetric (as shown in the proof of Proposition 3), so Proposition 3(i) gives $K_i^\top P_{t-1} = (P_{t-1}K_i)^\top = 0$ for every $i < t$; hence every vector in $\mathrm{range}(P_{t-1})$ is orthogonal to S_{t-1}, that is, $\mathrm{range}(P_{t-1}) \subseteq S_{t-1}^\perp$. The identification $\Sigma^{-1}(S_{t-1}^\perp) = (\Sigma S_{t-1})^\perp$ follows from $\langle \Sigma x, s\rangle = \langle x, \Sigma s\rangle$. For Equation 46, substitute $P_{t-1}\Sigma^{-1} = \Sigma^{-1}P_{t-1} + [P_{t-1}, \Sigma^{-1}]$ into $\Delta_t K_i = R_t(\lambda I + G_t)^{-1}K_t^\top P_{t-1}\Sigma^{-1}K_i$ and use $P_{t-1}K_i = 0$. If $\Sigma S_{t-1} = S_{t-1}$ then $(\Sigma S_{t-1})^\perp = S_{t-1}^\perp$ and the first claim already forces $\Delta_t K_i = 0$. □

Corollary 1 makes precise what recursive null-space alignment does and does not guarantee. The recursion of Equation 8 builds a *Euclidean* orthogonal projector, so it enforces $K_i^\top(P_{t-1}K_t) = 0$ for every $i < t$; the interference of the deployed update, however, is measured in the Σ^{-1} inner product, $K_i^\top\Sigma^{-1}(P_{t-1}K_t)$. The two coincide when S_{t-1} is Σ-invariant, and the residual leakage is therefore attributable entirely to the anisotropy of Σ. Section C.8 measures the resulting leakage and finds it small ($\mathrm{leak}_t = 0.305$ at the median, a $29.6\times$ reduction over $P_{t-1} = I$), consistent with the historical key subspace being close to, but not exactly, Σ-invariant.

F Extended Related Work

Knowledge editing aims to precisely modify specific knowledge encoded in Large Language Models (LLMs) while preserving unrelated capabilities. The existing literature is primarily divided into memory-based methods and model editing methods.

F.1 Memory-based Knowledge Editing

Memory-based approaches maintain an external repository of edited facts to avoid direct parameter interference. These can be further categorized by the nature of the stored memory:

Explicit Memory. Early methods like IKE (Zheng et al., 2023) leverage In-Context Learning (ICL) by appending demonstrations to the prompt. SERAC (Mitchell et al., 2022b) utilizes a memory bank and a classifier to route queries to an additional model when an edit is triggered.

Implicit Memory. GRACE (Hartvigsen et al., 2023) maintains a codebook at a specific model layer, using previous layer outputs as keys and learnable vectors (trained on target answers) as values to replace the original layer output.

Parametric Memory. This category introduces additional parameter modules without altering the base weights. MELO (Yu et al., 2024) trains chunk-wise LoRA matrices based on edit clusters, while Transformer-Patcher (Huang et al., 2023) adds new neurons to the final FFN layer to fix individual errors. A range of subsequent works further explores augmenting language models with lightweight calibration, intervention, or memory modules attached to specific layers to incorporate editable knowledge (Dong et al., 2022; Wang et al., 2025; Zhang et al., 2025; Liu et al., 2026b).

Limitations of Memory-based Methods. Relying heavily on retrieval accuracy, these methods struggle with generalized or paraphrased queries targeting edited facts. Furthermore, their dependence on external memory introduces fundamental bottlenecks in scalability, incurring continuously growing storage requirements and escalating inference-time computational overhead. In addition, this reliance tightly couples the model to its external memory infrastructure, severely constraining deployment flexibility and making cross-platform portability impractical.

F.2 Model Editing Methods

Model editing seeks to permanently "bake" knowledge into the model's parameters. This can be achieved through hyper-networks, localized optimization, or global training.

Hypernetwork-based Paradigm. This family of methods trains a hyper-network to produce targeted parameter modifications that encode edited knowledge. For example, MEND (Mitchell et al.,

2022a) and KnowledgeEditor (De Cao et al., 2021) learn to predict weight deltas from gradients derived from new factual triples. However, these approaches introduce additional parameters and training overhead. Moreover, since hyper-networks remain frozen after pretraining, they struggle to accommodate continual edits, which significantly limits their practical applicability.

Locate-then-Edit Paradigm. This paradigm identifies critical "knowledge neurons" before applying targeted updates. KN (Dai et al., 2022) identifies knowledge neurons via attribution and updates specific FFN values. ROME (Meng et al., 2022) uses causal tracing to locate a single layer and updates its weights using a closed-form solution. MEMIT (Meng et al., 2023) extends ROME to handle large-scale edits by distributing the update across multiple layers. Further extensions include MEMITCSK (Gupta et al., 2023) (considering predicates), PMET (Li et al., 2024) (including MHSA layers), and DEM (Huang et al., 2024) (leveraging hidden state similarity for localization). UltraEdit (Gu et al., 2026) performs editing without pre-computation of the second-moment matrix for the original knowledge representations.

To mitigate sequential decay and maintain model stability during continual editing, recent works have explored advanced constraints and weight matrix properties. AlphaEdit (Fang et al., 2025) introduces a null-space projection to ensure that updates do not interfere with pre-trained features, while EvoEdit (Lyu et al., 2026) employs a dynamic projection matrix to protect historically edited knowledge. Unlike these approaches, our recursive null-space alignment does not rely on pre-trained initialization, thereby avoiding the collapse of the feasible-update space and enabling more effective updates.

Some works focus on the properties of the weight matrices, employing regularization to constrain the parameter updates. NAS (Liu et al., 2026a) and MPES (Gupta et al., 2025) identify that catastrophic forgetting in sequential editing is closely tied to the unconstrained exponential growth of the weight matrix norm; NAS addresses this by constraining the target value vector's norm, whereas MPES introduces a regularization term to penalize deviations from the original weights. Furthermore, REVIVE (Zhang et al., 2026) observes that general semantic knowledge is stored in the principal singular space of the parameters, protecting it by filtering out updates in this space. PRUNE (Ma et al., 2025) notes that sequential editing increases the condition number of weight matrices, thereby degrading model capacity; it mitigates this by suppressing the singular values of the cumulative edit updates rather than individual ones. Finally, MOSE (Xu et al., 2026) replaces the standard additive update with a multiplicative orthogonal update matrix to better preserve matrix norms and condition numbers. In contrast to these approaches, which impose rigid constraints on weight norms or condition numbers, our energy-weighted penalty preserves manifold structure, naturally and effectively safeguarding general model capabilities.

Traditional methods are limited to atomic triplets. Recent works have expanded editing to unstructured long-form text. UnKE (Deng et al., 2025) treats the model as a key-encoder and value-generator, performing block-level localization and gradient-descent-based optimization. FT-UKE (Xiong et al., 2025) provides a training recipe that makes vanilla fine-tuning effective for unstructured knowledge editing. EvoEdit-Cao (Cao et al., 2025) addresses lifelong editing through latent perturbation and a knowledge-driven parameter fusion mechanism. To adapt the locate-then-edit paradigm to unstructured long text, AnyEdit (Jiang et al., 2025) decomposes long-form text into uniform chunks and iteratively edits the last token of each chunk. μKE (Su et al., 2025) further introduces a Matryoshka-style loss to capture multi-chunk dependencies, while AnyEdit++ (Tian et al., 2026) instead places chunk boundaries at peaks of Bayesian Surprise. Instead of heuristically segmenting input samples, which yields the mediocre-point dilemma, our Pivot Localization identifies easily displaceable regions on the edit sub-manifold, therefore enabling more effective unstructured knowledge updates.

G LIMITATIONS

Despite ManiEdit's strong performance in sequential unstructured knowledge editing, it still faces a key limitation: ManiEdit is currently confined to textual knowledge editing and lacks support for multimodal integration. Extending our manifold-based paradigm to accommodate cross-modal updates (e.g., text, images, and audio) would significantly broaden its applicability, presenting a promising direction for future research in multimodal large language models.