KEY: UNLOCKING DISENTANGLED MODEL UPDATING WITH A KNOWLEDGE ADDRESSING DICTIONARY

Anonymous authors
Paper under double-blind review

ABSTRACT

Recent years have witnessed growing interest in knowledge updating (*e.g.*, editing and unlearning) in language models. Despite substantial progress, existing methods remain challenged by *knowledge entanglement*, where different knowledge items share overlapping parameters or representational directions, often causing unintended interference. To address this issue, this paper introduces **KEY** (**K**nowledge addr**E**ssing dictionar**Y**), aiming for unlocking distinguishable knowledge structures within the entangled weight space of language models. Specifically, the proposed method first constructs an orthogonal atom dictionary from knowledge-induced gradients by decomposing their input- and output-side directions. It then performs sparse addressing to identify a compact set of atoms and their coefficients for each update. Theoretically, both global and local proximity bounds are established, characterizing conditions under which the learned dictionary preserves the desired knowledge structure and supports stable knowledge addressing. Extensive experiments across both knowledge editing and unlearning tasks demonstrate that *KEY* consistently achieves competitive performance against recent state-of-the-art baselines, while enabling more interpretable and disentangled knowledge updating.

1 INTRODUCTION

Through extensive pre-training on massive corpora, large language models (LLMs) encode substantial factual and relational knowledge in their parameters(Dai et al., 2022), which can be directly elicited in knowledge-intensive tasks such as question answering(Yang et al., 2024), fact verification(Guan et al., 2024), and open-domain dialogue(Kim et al., 2023). However, real-world knowledge is inherently dynamic: previously valid facts may become outdated, incorrect associations may require correction, and sensitive information may need to be removed due to privacy or safety issues. These evolving demands motivate *post-training knowledge updating*, which modifies a model's internal knowledge without full retraining. Following recent work (Luo et al., 2026), post-training knowledge updating can be broadly characterized by two complementary operations: *knowledge editing*(Meng et al., 2022), which replaces an existing association with a specified target, and *knowledge forgetting*(Jang et al., 2023), which suppresses access to designated knowledge without an explicit replacement. Both operations aim to selectively modify specific knowledge encoded in model parameters while preserving unrelated capabilities and minimizing unintended behavioral changes.

Yet, precise knowledge updating remains challenging because knowledge in LLMs is often encoded through overlapping internal representations rather than in isolation. Prior work characterizes this phenomenon as *knowledge entanglement*, where multiple facts, concepts, and capabilities may share model parameters or representational directions (Liu et al., 2024). Consequently, modifying the representation associated with one piece of knowledge may also perturb those partially shared by others. In knowledge editing, for instance, such interference can manifest as *collateral effects* (Gu et al., 2024) or *distribution shift* (Fang et al., 2025), where updating a target association alters representations beyond the intended knowledge and consequently affects unrelated predictions. As illustrated in Fig. 1, such entanglement makes it difficult to update one target knowledge without interfering with others.

Interestingly, overlapping representation does not necessarily imply complete indistinguishability. Human semantic memory provides a compelling biological perspective on this issue. Specifically,

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.

Haxby et al. (2001) show that neural representations of different object categories are distributed and overlapping, yet remain discriminable from their activation patterns, while Huth et al. (2016) reveal an organized cortical semantic map in which distinct semantic domains exhibit systematic yet partially overlapping representations. These findings suggest that biological knowledge representations support both **association** and **separation**. This naturally raises the following question:

Can entangled knowledge in LLMs likewise retain a distinguishable internal structure that supports precise knowledge updating?

To explore such latent structure, this paper introduces ***K**nowledge addr**E**ssing dictionar**Y*** (*KEY*), which seeks separable knowledge structure within an otherwise entangled weight space through a shared gradient-derived dictionary. Specifically, *KEY* distills knowledge-induced gradients into a compact set of orthogonal reference directions and represents each update as a sparse combination of these directions. The resulting coefficients subsequently serve as a knowledge-specific address, revealing distinguishable structures among knowledge items encoded in overlapping model weights. Knowledge updates are accordingly performed through these task-specific addresses, supporting targeted modification while reducing interference with unrelated knowledge. We summarize our contributions as follows[1]:

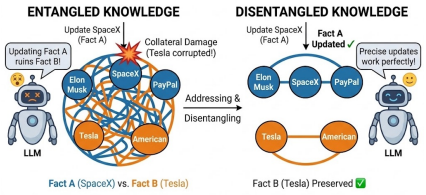


Figure 1: Knowledge disentanglement enables precise target updates while preserving unrelated facts.

- We formulate knowledge updating from a *knowledge-addressing* perspective, providing a new view of how distinct knowledge can be identified despite overlapping parameter usage.
- We propose *KEY*, which constructs a shared orthogonal dictionary from knowledge-induced gradients and assigns each knowledge item a sparse address for subsequent targeted updating.
- We establish conditions that guarantee stable knowledge addresses, *i.e.* coefficient vectors, for close knowledge, which supports effective, selective updating with improved knowledge separation across editing and unlearning tasks.
- Extensive experiments on multiple benchmarks and LLM backbones demonstrate strong performance on both knowledge editing and unlearning tasks, achieving up to 2.8% and 4.8% relative improvements over the strongest baselines, respectively, while exhibiting improved inter-knowledge separability and intra-knowledge consistency.

2 PRELIMINARIES

Knowledge Updating in LLMs. Considering a LLM with L stacked Transformer layers. Each layer is primarily formulated as two key components, *i.e.*, a multi-head self-attention module ($\mathrm{MHA}(\cdot)$) and a feed-forward network (FFN) (Fang et al., 2025; Xue & Andrzejak, 2026). Let H^{l-1} denote the hidden representations entering layer l. The attention module first produces the intermediate representations, which are subsequently transformed by the position-wise FFN:

$$\widetilde{H}^l = H^{l-1} + \mathrm{MHA}^l\left(\mathrm{Norm}(H^{l-1})\right), \quad H^l = \widetilde{H}^l + W^l_{\mathrm{out}}\sigma\Big(W^l_{\mathrm{in}}\,\mathrm{Norm}(\widetilde{H}^l)\Big), \tag{1}$$

where W^l_{in} and W^l_{out} denote the input and output weight matrices of the FFN, respectively, and $\sigma(\cdot)$ denotes the activation function. After propagation through all L layers, the hidden representation at the last input position, $\boldsymbol{h}^L_t$, is projected onto the vocabulary space. Let $W_{\mathrm{U}} \in \mathbb{R}^{|\mathcal{V}|\times d}$ denote the output projection matrix, where $|\mathcal{V}|$ is the vocabulary size and d is the hidden dimension. The vocabulary logits are

$$\boldsymbol{z} = W_{\mathrm{U}}\,\mathrm{Norm}(\boldsymbol{h}^L_t) \in \mathbb{R}^{|\mathcal{V}|}, \tag{2}$$

and the probability of predicting vocabulary token $\mathcal{V}_j$ is $p(\mathcal{V}_j) = [\mathrm{softmax}(\boldsymbol{z})]_j$.

Let $\mathcal{W}_{\mathrm{full}} = \bigcup_{l=1}^{L}\{W_b^l\}_{b=1}^{B_l}$ denote the collection of all component weights across all L Transformer layers, where W_b^l denotes the b-th learnable weight matrix in layer l, and B_l denotes the total component number of $\mathrm{MHA}(\cdot)$ and FFN in layer l. Accordingly, knowledge updating methods (Meng

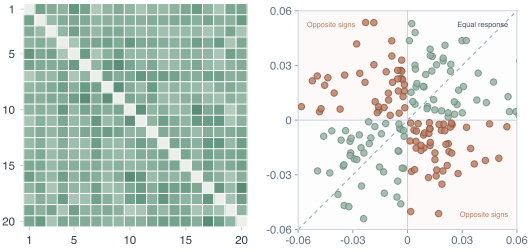
[1] Source codes are available on https://anonymous.4open.science/r/KEY-3BCC/.

et al., 2022; Fang et al., 2025; Luo et al., 2026) can be formulated as $W_b^l \leftarrow W_b^l + \Delta W_b^l$, for any $W_b^l \in \mathcal{W}_{\text{full}}$. We use W for W_b^l hereafter for simplicity. The following derivations are performed with respect to this individual matrix while keeping all other model parameters fixed. More details about existing knowledge updating methods can be found in Appendix A.1.

Entanglement in Knowledge Updating. Despite substantial progress in knowledge updating, precisely modifying one piece of knowledge without affecting others remains challenging. A fundamental source of this difficulty is *knowledge entanglement*, where knowledge associated with different facts is encoded through overlapping latent representations or model parameters, such that a targeted update may induce unintended changes to unrelated knowledge. Several recent studies attempt to alleviate this issue from different perspectives. ❶ *Representation-level separation* seeks to distinguish target-related information from knowledge that should remain unchanged, thereby reducing interference at the hidden-representation level, as exemplified by DiKE (Zhang et al., 2026). ❷ *Parameter-level isolation* identifies neurons, layers, or parameter subsets that are particularly relevant to the target knowledge and restricts modification to these components (Chen et al., 2026b;c; Zhang et al., 2025). ❸ *update-direction constraints* directly regulate parameter perturbations, for example by projecting updates onto orthogonal or null spaces constructed from preserved knowledge (Fang et al., 2025; Xu et al., 2025). A comprehensive review is provided in Appendix A.2.

3 Can Entangled Knowledge Be Separated in Weight Space?

Existing methods mainly constrain *where* or *how* knowledge should be updated, while whether entangled knowledge remains distinguishable in weight space is still largely underexplored. Such separability is important for model interpretability and for distinguishing knowledge-specific patterns. Toward this end, two questions are examined: ❶ to what extent different knowledge updates overlap in original weight space, and ❷ whether identical perturbations to shared parameters induce different responses across knowledge items.



(a) Parameter Overlap. (b) Divergent Responses.

Figure 2: (a) Pairwise parameter overlap across 20 independently learned knowledge. (b) Paired target-logit responses to the same perturbation; ● and ● indicate matching and opposite signs.

Parameter Overlap Across Distinct Knowledge. The first analysis quantifies the parameter locations jointly affected when distinct knowledge are learned independently. Specifically, TOFU (Maini et al., 2024), a widely-used updating benchmark with synthetic question–answer pairs $\langle \boldsymbol{q}, \boldsymbol{a} \rangle$ describing fictional author profiles, is adopted, from which $N = 20$ pairs are randomly sampled. Using the same backbone LLM (Llama 3) parameterized by W_0, the model is independently fine-tuned on each sampled knowledge (*i.e.*, author profile) under identical settings, yielding the update $\Delta W_i = W_i - W_0$ ($i \in [1, N]$). The affected parameter locations are identified as $\text{loc}_\tau(\Delta W_i) = \{p | \tau < |(\Delta W_i)_p|\}$, with $\tau = 10^{-5}$ and p indexing individual weight. The location overlap between two knowledge updates is further measured using the Jaccard similarity, *i.e.*, $J_{ij} = |\text{loc}_\tau(\Delta W_i) \cap \text{loc}_\tau(\Delta W_j)| / |\text{loc}_\tau(\Delta W_i) \cup \text{loc}_\tau(\Delta W_j)|$, where $i, j \in [1, N]$ and $i \neq j$. As shown in Fig. 2(a), distinct knowledge exhibits substantial overlap in affected parameters, with J_{ij} ranging from 11% to 45% across pairwise comparisons. This finding is consistent with prior studies (Wang et al., 2025; Hu et al., 2025) and suggests that parameter locations alone are insufficient to distinguish individual knowledge. Detailed procedures are provided in Appendix B.

Divergent Responses to Perturbations. Although distinct knowledge relies on shared parameter locations (see above), does it respond to changes in the same way? If distinguishable structure remains within such shared parameters, the same perturbation should induce different responses across knowledge items. This motivates the following perturbation analysis. Specifically, we jointly fine-tune W_0 on all N sampled knowledge under the same settings, yielding a common model W_{all}. For knowledge i and j, a perturbation Ξ_{ij} is sampled from a zero-mean Gaussian distribution and applied to their overlapping update locations identified above, *i.e.*, $\text{loc}\,\tau(\Delta W_i) \cap \text{loc}\,\tau(\Delta W_j)$. Following Eq. (2), the response of knowledge i is measured by the mean pre-softmax logit assigned

to its ground-truth answer ($\boldsymbol{a}_i$),

$$s_i(W_{\text{all}}) = \frac{1}{|\boldsymbol{a}_i|} \sum_{t=1}^{|\boldsymbol{a}_i|} [\boldsymbol{z}_t^{(i)}(W_{\text{all}})]_{a_{i,t}}, \tag{3}$$

where $[\boldsymbol{z}^{(i)}(W_{\text{all}})]a_{i,t}$ denotes the pre-softmax logit assigned to the t-th ground-truth token $a_{i,t}$ under W_{all}. The perturbation response is then measured as $\delta s_i = s_i(W_{\text{all}} + \Xi_{ij}) - s_i(W_{\text{all}})$, with δs_j defined analogously. These results are illustrated in Fig. 2(b), where the same perturbation induces substantially different target-logit changes across paired knowledge profiles, differing in magnitude and, in even some cases, sign. Detailed procedures are provided in Appendix B.

Takeaway. Substantial parameter overlap does not imply knowledge indistinguishability: identical perturbations can still induce distinct, even opposite, responses across knowledge items.

To characterize such responses in parameter space, consider a small perturbation Ξ. Its first-order Taylor approximation gives (omitting knowledge input to avoid cluttered notations)

$$s(W + \Xi) \approx s(W) + \langle \nabla_W s(W), \Xi \rangle_F . \tag{4}$$

Notably, $\nabla_W s(W)$ characterizes some sensitivity of knowledge along different parameter-space directions. This gradient therefore provides a knowledge-conditioned representation for identifying distinguishable structures within the shared weight space.

4 METHOD

Based on the above analyses, this section introduces the ***K**nowledge addr**E**ssing dictionar**Y*** (*KEY*) for disentangled knowledge updating. As illustrated in Fig. 3, *KEY* consists of two key stages: (i) *Knowledge Addressing Dictionary Construction* (Sec. 4.1), which constructs an orthogonal atom dictionary from knowledge-induced gradients; and (ii) *Task-specific Sparse Addressing* (Sec. 4.2), which identifies a compact knowledge-specific address over the dictionary for each update.

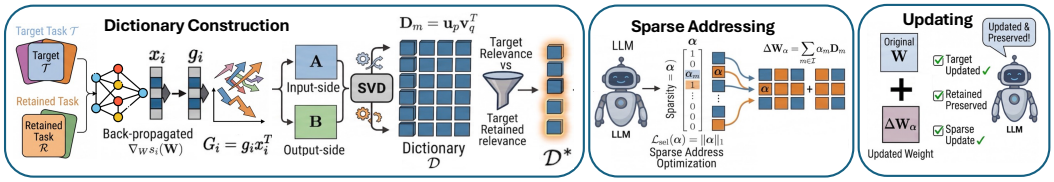


Figure 3: Overview of *KEY*: constructing an orthonormal knowledge-addressing dictionary followed by task-specific sparse addressing for disentangled knowledge updating.

Problem Setting. Let $\mathcal{P} = \{\langle \boldsymbol{q}_i, \boldsymbol{a}_i \rangle\}_{i=1}^{|\mathcal{P}|}$ denote the full knowledge pool, where $\boldsymbol{q}_i$ and $\boldsymbol{a}_i$ represent the query and corresponding answer of knowledge item i, respectively. For a given updating task, $\mathcal{T} \subseteq \mathcal{P}$ denotes the target knowledge to be updated, while an optional set $\mathcal{R} \subseteq \mathcal{P} \setminus \mathcal{T}$ denotes the retained knowledge expected to remain unchanged. For knowledge editing, each target item is associated with a replacement answer $\boldsymbol{a}_i^+$, whereas knowledge unlearning aims to suppress the original answer $\boldsymbol{a}_i$. In both cases, the objective is to modify the model with respect to $\mathcal{T}$ while minimizing undesired changes to $\mathcal{R}$, when provided.

4.1 KNOWLEDGE ADDRESSING DICTIONARY CONSTRUCTION

Given a knowledge collection $\mathcal{C} \subseteq \mathcal{P}$, in light of Eq. (4), the knowledge-induced gradient $\nabla_W s_i(W)$ is indicative of how knowledge i ($i \in [1, |\mathcal{C}|]$) is coupled to W. However, raw gradients may contain numerous correlated and redundant directions (see Appendix B.3), making direct comparison among raw gradients insufficient to reveal any underlying structure. To address this, *shared and non-redundant reference bases (referred as the **dictionary**)* are first constructed.

Consider an arbitrary weight matrix $W \in \mathcal{W}_{\text{full}}$. For knowledge i, let $\boldsymbol{x}_{i,t}$ denote the input to W at token position t, and let $\boldsymbol{y}_{i,t} = W\boldsymbol{x}_{i,t}$ denote the corresponding pre-activation output. By the chain

rule, the rank-1 gradient of $s_i(W)$ with respect to W can be written as (Mitchell et al., 2022):

$$\nabla_W s_i(W) = \sum_t \frac{\partial s_i(W)}{\partial \boldsymbol{y}_{i,t}} (\boldsymbol{x}_{i,t})^\top = \sum_t \boldsymbol{g}_{i,t} (\boldsymbol{x}_{i,t})^\top = \sum_t G_{i,t}, \tag{5}$$

where $\boldsymbol{g}_{i,t} = \partial s_i(W)/\partial \boldsymbol{y}_{i,t}$ denotes the gradient back-propagated to the pre-activation output of W. To separate direction from magnitude, each $G_{i,t}$ is further decomposed as

$$G_{i,t} = \rho_{i,t} \hat{\boldsymbol{g}}_{i,t} (\hat{\boldsymbol{x}}_{i,t})^\top, \qquad \hat{\boldsymbol{x}}_{i,t} = \frac{\boldsymbol{x}_{i,t}}{\|\boldsymbol{x}_{i,t}\|_2}, \qquad \hat{\boldsymbol{g}}_{i,t} = \frac{\boldsymbol{g}_{i,t}}{\|\boldsymbol{g}_{i,t}\|_2}, \tag{6}$$

where $\rho_{i,t} = \|\boldsymbol{x}_{i,t}\|_2 \|\boldsymbol{g}_{i,t}\|_2$ preserves the magnitude. Since $\boldsymbol{x}_{i,t}$ is the activation entering W and $\boldsymbol{g}_{i,t}$ is the gradient back-propagated from its output, $\hat{\boldsymbol{x}}_{i,t}$ and $\hat{\boldsymbol{g}}_{i,t}$ characterize the corresponding *input*- and *output*-side directions, respectively. All such triplets across $\mathcal{C}$ are pooled and re-indexed as $\{(\hat{\boldsymbol{x}}_k, \hat{\boldsymbol{g}}_k, \rho_k)\}_{k=1}^K$, where k is the contracted form of i, t and K is the total number of triplets, and are further stacked as

$$A = [\sqrt{\rho_1}\hat{\boldsymbol{x}}_1, \ldots, \sqrt{\rho_K}\hat{\boldsymbol{x}}_K]^\top \in \mathbb{R}^{K \times d_{\text{in}}}, \qquad B = [\sqrt{\rho_1}\hat{\boldsymbol{g}}_1, \ldots, \sqrt{\rho_K}\hat{\boldsymbol{g}}_K]^\top \in \mathbb{R}^{K \times d_{\text{out}}}. \tag{7}$$

Accordingly, let V and U denote the input- and output-side reference bases for A and B, respectively, which play a role in capturing the dominant common directions across the knowledge-induced gradients. Moreover, to avoid redundant reference directions, both bases are constrained to be orthonormal, *i.e.*, $V^\top V = I$ and $U^\top U = I$. This constraint ensures mutually orthogonal and non-redundant reference directions; otherwise, highly correlated basis may encode similar gradients repeatedly. Under such constraints, $VV^\top$ and $UU^\top$ form orthogonal projectors onto $\text{span}(V)$ and $\text{span}(U)$, respectively. Accordingly, $AVV^\top$ and $BUU^\top$ reconstruct the original A and B, by projecting them onto their respective shared spaces. These reference bases are obtained by minimizing the reconstruction losses:

$$V^\star = \arg\min_{V^\top V = I} \left\|A - AVV^\top\right\|_F^2, \qquad U^\star = \arg\min_{U^\top U = I} \left\|B - BUU^\top\right\|_F^2. \tag{8}$$

Notably, the optimization problem in Eq. (8) admits a closed-form solution via singular value decomposition (SVD). Taking the input-side basis as an example, under $V^\top V = I$, the reconstruction objective is equivalently $V^\star = \arg\max_{V^\top V = I} \text{tr}(V^\top A^\top A V)$. By the Rayleigh–Ritz/Ky Fan theorem, V is given by the right singular directions of A; retaining those associated with non-zero singular values spans its effective input-side subspace and yields $A = AVV^\top$. Specifically, for the full SVDs $A = P_A \Sigma_A V_A^\top$ and $B = P_B \Sigma_B U_B^\top$, V and U collect the right singular vectors corresponding to non-zero singular values, with $B = BUU^\top$ analogously. Detailed derivations are provided in Appendix C.1. In practice, to reduce computational and storage costs, only the top r_{in} and r_{out} directions associated with the largest singular values are retained, where r_{in} and r_{out} are hyperparameters. Accordingly, $V^\star = [\boldsymbol{v}_1, \ldots, \boldsymbol{v}_{r_{\text{in}}}]$ and $U^\star = [\boldsymbol{u}_1, \ldots, \boldsymbol{u}_{r_{\text{out}}}]$ denote the corresponding truncated optimal bases. An ablation study of r_{in} and r_{out} is provided in Appendix E.3. Finally, this decomposition naturally leads to $M = r_{\text{out}} r_{\text{in}}$ dictionary atoms,

$$D_m = \boldsymbol{u}_p \boldsymbol{v}_q^\top, \qquad m = (p-1) r_{\text{in}} + q, \tag{9}$$

where $p \in [1, r_{\text{out}}]$, $q \in [1, r_{\text{in}}]$, and $\boldsymbol{u}_p$ and $\boldsymbol{v}_q$ denote the corresponding columns of $U^\star$ and $V^\star$, respectively. Each index $m \in [1, M]$ therefore uniquely corresponds to an input–output direction pair (p, q). The resulting dictionary is defined as $\mathcal{D} = \{D_m\}_{m=1}^M$, providing a common set of parameter-space directions for representing different knowledge items. Define Frobenius matrices inner product $\langle A, B \rangle_F = \langle \text{vec}(A), \text{vec}(B) \rangle$ where $\text{vec}(\cdot)$ is the vectorization operation by stacking rows or columns of a matrix. Notably, since U and V are orthonormal, different atoms are also mutually orthonormal, that is, $\langle D_m, D_{m'} \rangle_F = (\boldsymbol{u}_p^\top \boldsymbol{u}_{p'})(\boldsymbol{v}_q^\top \boldsymbol{v}_{q'}) = \delta_{mm'}$, where $\delta_{mm'}$ denote the Kronecker deltas. Thus, $\mathcal{D}$ forms a non-redundant orthonormal dictionary in the shared weight space. The above construction is performed independently for each $W_b^l \in \mathcal{W}_{\text{full}}$, yielding a weight-specific dictionary. These dictionaries are then collectively organized to form the model-level dictionary.

Atom Filtering. The aforementioned dictionary construction is formulated over a generic knowledge collection $\mathcal{C}$. For a concrete updating task, $\mathcal{C}$ can be instantiated as either the full knowledge pool $\mathcal{P}$ or the target knowledge $\mathcal{T}$, from which the dictionary $\mathcal{D}$ is constructed (The ablation study comparing these choices is provided in Sec. 5.2). When retained knowledge $\mathcal{R}$ is available, $\mathcal{D}$ can

be further refined by filtering atoms that are strongly associated with $\mathcal{T}$ while exhibiting limited dependence on $\mathcal{R}$. That is, for each atom $D_m \in \mathcal{D}$, its relevance to $\mathcal{T}$ and $\mathcal{R}$ is quantified as

$$F_m = \frac{1}{|\mathcal{T}|}\sum_{i\in\mathcal{T}}\left|\langle \nabla_W s_i(W), D_m\rangle_F\right|, \qquad R_m = \frac{1}{|\mathcal{R}|}\sum_{i\in\mathcal{R}}\left|\langle \nabla_W s_i(W), D_m\rangle_F\right|. \tag{10}$$

The target-to-retain relevance ratio $S_m = \tilde{F}_m/(\tilde{R}_m+\epsilon)$ is derived where $\tilde{F}_m$ and $\tilde{R}_m$ are normalized by their corresponding 90-th percentile as in Eq. (35) to reduce scale sensitivity. Then the atoms are ranked according to S_m and top-N_{atom} atoms are retained to form the compact dictionary $\mathcal{D}^\star$, emphasizing target-relevant directions while suppressing those strongly associated with retained knowledge. Detailed normalization and the case of $\mathcal{R} = \varnothing$ are provided in Appendix C.3.

4.2 Task-specific Sparse Addressing

With the filtered dictionary $\mathcal{D}^\star$, a specific knowledge-update task needs to determine how the dictionary atoms should contribute to the update and with what strengths. Let $\boldsymbol{\alpha} = [\alpha_1, \ldots, \alpha_{|\mathcal{D}^\star|}]^\top$ denote its corresponding coefficients. The parameter update is represented as

$$\Delta W_{\boldsymbol{\alpha}} = \sum_{D_m\in\mathcal{D}^\star} \alpha_m D_m. \tag{11}$$

Thus, $\boldsymbol{\alpha}$ provides a knowledge address over the dictionary, specifying how parameter-space directions are combined and weighted.

Update-Aware Learning. For each target knowledge item $i \in \mathcal{T}$, represented by a question–answer pair $\langle \boldsymbol{q}_i, \boldsymbol{a}_i\rangle$, the coefficient vector $\boldsymbol{\alpha}$ is optimized according to the required update type. For editing, let $\boldsymbol{a}_i^+$ denote the replacement answer associated with $\boldsymbol{q}_i$, while $\boldsymbol{a}_i$ denotes the original answer to be suppressed for forgetting. The update cost objective to be minimized is formulated as

$$\mathcal{L}_{\text{upd}}(\boldsymbol{\alpha}) = \frac{1}{|\mathcal{T}|}\sum_{i\in\mathcal{T}} \begin{cases} -\frac{1}{|\boldsymbol{a}_i^+|}\sum_{t=1}^{|\boldsymbol{a}_i^+|}\log p_{\boldsymbol{\alpha}}(a_{i,t}^+ \mid \boldsymbol{q}_i, \boldsymbol{a}_{i,<t}^+), & \textsc{Edit}, \\ -\frac{1}{|\boldsymbol{a}_i|}\sum_{t=1}^{|\boldsymbol{a}_i|}\log\big(1 - p_{\boldsymbol{\alpha}}(a_{i,t} \mid \boldsymbol{q}_i, \boldsymbol{a}_{i,<t})\big), & \textsc{Forget}. \end{cases} \tag{12}$$

where $p_{\boldsymbol{\alpha}}(\cdot)$ denotes the token probability produced by the updated weights $W_{\boldsymbol{\alpha}} = W + \Delta W_{\boldsymbol{\alpha}}$, with W denoting the original model weights.

Retain-Aware Regularization. To preserve $\mathcal{R}$, both its ground-truth likelihood and the original output distribution are maintained. Specifically, we minimize the following cost

$$\mathcal{L}_{\text{ret}}(\boldsymbol{\alpha}) = \mathcal{L}_{\text{NLL}}(\mathcal{R}; W_{\boldsymbol{\alpha}}) + \mathcal{L}_{\text{KL}}\big(p_W \,\|\, p_{\boldsymbol{\alpha}}; \mathcal{R}\big), \tag{13}$$

where p_W denotes the token probability produced by the original weights W, and the two terms denote the token-level negative log-likelihood and KL divergence over retained knowledge, respectively. The **overall training objective** is then given by

$$\mathcal{L}(\boldsymbol{\alpha}) = \mathcal{L}_{\text{upd}}(\boldsymbol{\alpha}) + \lambda_{\text{ret}}\mathcal{L}_{\text{ret}}(\boldsymbol{\alpha}) + \lambda_{\text{sel}}\mathcal{L}_{\text{sel}}(\boldsymbol{\alpha}), \tag{14}$$

where $\mathcal{L}_{\text{sel}}(\boldsymbol{\alpha}) = \|\boldsymbol{\alpha}\|_1$ is the ℓ_1 regularization, and $\lambda_{\text{ret}} + \lambda_{\text{sel}} = 1$[2].

Bounded proximity. For a fixed filtered dictionary, nearby knowledge-update tasks under a continuous parameterization admit nearby coefficient vectors, provided that the active-coordinate Hessian is positive definite and the inactive optimality inequalities hold strictly. Under these conditions, the corresponding local minimizer varies Lipschitz continuously with the task parameter. A global bound holds under stronger uniform curvature and sensitivity assumptions. Precise statements and proofs are provided in Appendix D.

5 Experiments

Datasets and Baselines. For knowledge editing, CounterFact (Meng et al., 2022) and FINE-KED (Zhang et al., 2026) are adopted. The baselines include fine-tuning (FT), ROME (Meng et al., 2022), MEMIT (Meng et al., 2023), AlphaEdit (Fang et al., 2025), and DiKE (Zhang et al., 2026). For knowledge unlearning, experiments are conducted on two widely adopted benchmarks,

[2] When $\mathcal{R} = \varnothing$, the retain-aware objective is omitted by setting $\lambda_{\text{ret}} = 0$ and $\lambda_{\text{sel}} = 1$.

TOFU (Maini et al., 2024) and MUSE-News (Shi et al., 2025). The proposed method is compared with NPO (Zhang et al., 2024), SimNPO (Fan et al., 2025), TPO (Zhou et al., 2026), ALTER (Chen et al., 2026c), and PALU (Zhai et al., 2026).

Evaluation Metrics. Following Zhang et al. (2026), knowledge editing is evaluated on CounterFact using Efficacy Success (ES), Paraphrase Success (PS), and Neighborhood Success (NS), and on FINE-KED using Efficacy (Eff) and Relational Locality (RL with Easy, Middle, and Hard subsets). Similar to (Zhai et al., 2026), knowledge unlearning is evaluated on TOFU using Forget Quality (FQ), Model Utility (MU), and Exact Memorization (EM), and on MUSE–News using Knowledge Memorization (KM_r, KM_f), Verbatim Memorization (VM_f), and Privacy Leakage (PL).

Implementation Details. Llama-2-7B and Llama-3.1-8B are used for knowledge unlearning, while Llama-3-8B and GPT-J-6B are used for knowledge editing. Experiments use NVIDIA A100 GPUs with 80 GB memory. *Baseline results are taken directly from the corresponding original papers.* For *KEY*, the dictionary is established using the full dataset and we set $r_{in} = r_{out} = 48$, $N_{atom} = 1024$. Detailed dataset configurations and hyperparameter settings are provided in Appendix E.1.

5.1 Effectiveness of *KEY* on Knowledge Updating

Table 1: Performance comparison of different knowledge editing methods on CounterFact and FINE-KED over five runs. The **best** and second-best results are highlighted.

Model	Method	CounterFact				FINE-KED				
		ES↑	PS↑	NS↑	Avg.↑	Eff↑	Easy(RL)↑	Mid(RL)↑	Hard(RL)↑	Avg.(RL)↑
Llama-3-8B	FT	**100.0**	97.7	7.0	68.2	**100.0**	69.4	63.0	49.6	62.8
	MEMIT	99.8	94.6	80.9	91.8	98.7	64.6	54.9	62.6	61.5
	ROME	**100.0**	**98.9**	77.3	92.1	99.8	56.8	50.7	47.8	53.0
	AlphaEdit	**100.0**	93.5	78.5	90.7	98.2	68.1	59.2	67.8	65.6
	DiKE	99.9	96.6	82.8	93.1	99.1	72.7	65.3	72.4	70.6
	***KEY* (Ours)**	99.9	98.1	**83.3**	**93.8**	99.5	**73.5**	**67.1**	**73.1**	**71.7**
	Std.	±0.1	±0.1	±0.2	±0.2	±0.1	±0.2	±0.2	±0.3	±0.2
GPT-J-6B	FT	**100.0**	95.8	9.8	68.5	**100.0**	69.2	64.8	60.6	65.9
	MEMIT	99.6	95.2	79.2	91.3	99.9	61.9	56.0	59.5	59.7
	ROME	99.9	**99.7**	75.9	91.8	99.9	53.5	49.4	44.9	50.3
	AlphaEdit	99.6	96.9	79.3	91.9	99.9	59.8	54.9	54.8	57.2
	DiKE	99.8	96.1	79.3	91.7	99.1	72.0	67.4	74.1	71.3
	***KEY* (Ours)**	99.7	98.3	**79.9**	**92.6**	99.5	**73.7**	**68.1**	**75.2**	**72.6**
	Std.	±0.1	±0.2	±0.1	±0.2	±0.2	±0.1	±0.1	±0.2	±0.2

Table 2: Performance comparison of different knowledge unlearning methods on TOFU (5%) and MUSE–News over five runs. The **best** and second-best results are highlighted.

Method	TOFU: Llama-2-7B			TOFU: Llama-3.1-8B			MUSE–News: Llama-2-7B			
	FQ↑	MU↑	EM↓	FQ↑	MU↑	EM↓	KM_r ↑	KM_f ↓	VM_f ↓	PL→ 0
NPO	0.63	0.59	0.66	0.63	0.60	0.68	0.46	0.60	0.43	-90.85
SimNPO	0.47	0.59	0.73	0.63	0.58	0.65	0.41	0.58	0.38	-99.90
ALTER	0.69	0.63	0.60	0.72	0.60	0.61	0.49	**0.44**	0.31	-47.38
TPO	0.63	0.59	0.66	0.72	0.59	0.60	0.50	0.59	0.42	-73.64
PALU	**0.71**	0.62	**0.59**	**0.92**	0.62	0.65	0.47	**0.44**	**0.27**	-45.91
***KEY* (Ours)**	0.69	**0.66**	**0.59**	0.75	**0.64**	**0.58**	**0.52**	0.47	0.28	**-43.22**
Std.	±0.02	±0.03	±0.03	±0.01	±0.02	±0.02	±0.03	±0.02	±0.03	±0.45

On Knowledge Editing. Table 1 shows that *KEY* outperforms existing baselines on 12 of the 18 evaluation metrics. On CounterFact, for instance, *KEY* achieves the highest average scores on both Llama-3-8B (93.8%) and GPT-J-6B (92.6%), together with the best NS of 83.3% and 79.9%, respectively. These results indicate improved preservation of neighboring knowledge while effectively updating the target knowledge. On FINE-KED, *KEY* consistently achieves the best RL across all difficulty levels and both base models. Although its Eff is slightly lower than some baselines (*e.g.*, 99.5% compared to 99.9% for AlphaEdit on GPT-J-6B), this marginal difference is accompanied by consistent gains in RL.

On Knowledge Unlearning. As observed from Table 2, *KEY* achieves the best results in 6 of the 10 evaluation settings across two datasets and two base models. On TOFU, *KEY* attains the highest MU and the lowest EM on both models. Although PALU achieves higher FQ, *KEY* offers a better

balance between forgetting effectiveness and model utility. On MUSE–News, PALU achieves lower KM_f and VM_f, whereas *KEY* obtains higher KM_r and better PL. These results indicate that, while maintaining competitive forgetting of target knowledge, *KEY* better preserves retained knowledge and limits privacy leakage, reflecting a more selective separation between knowledge to be removed and knowledge to be preserved.

In summary, these results empirically confirm the effectiveness of *KEY* for knowledge updating, achieving strong target modification while better preserving non-target knowledge across both editing and unlearning tasks. Additional results on generalization across different dataset scales and domains are provided in Appendix E.2.

5.2 Impact of Dictionary Construction

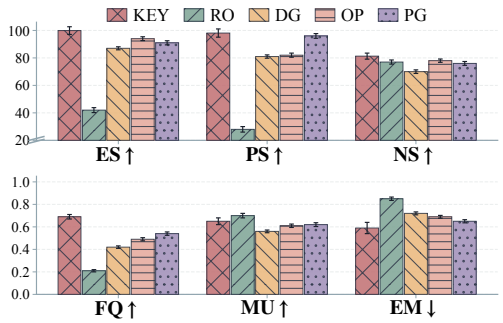


Figure 4: Impact of dictionary construction on editing (CounterFact, top) and unlearning (TOFU, bottom) tasks.

On Design Choices. To examine how dictionary construction affects target updating and retained-knowledge preservation, four variants of *KEY* are considered: (i) *Random-Orthogonal (**RO**)*, replacing the learned dictionary with a randomly generated orthogonal dictionary; (ii) *Direct-Gradient (**DG**)*, directly using knowledge-induced gradients as dictionary atoms; (iii) *One-to-one Pairing (**OP**)*, restricting basis pairing to $\boldsymbol{u}_p \boldsymbol{v}_p^\top$ rather than all pairwise combinations; and (iv) *Probability-Gradient (**PG**)*, computing gradients using target-token probabilities instead of pre-softmax logits. CounterFact and TOFU are used for knowledge editing and unlearning, with Llama-3-8B and Llama-2-7B, respectively. Fig. 4 shows that replacing any individual design choice degrades performance to varying degrees, confirming the effectiveness of the proposed construction. To begin with, as expected, *RO* causes the largest overall degradation, indicating that orthogonality alone is insufficient to capture knowledge-specific update directions. The slightly lower MU of *KEY* compared with RO results from its random atoms being less aligned with knowledge-relevant update directions, leading to weaker target forgetting and, consequently, less impact to the model's retained utility. *OP* reduces PS to 82.0 and FQ to 0.49, as one-to-one pairing yields fewer rank-one directions for dictionary construction than all pairwise combinations. In addition, *PG* results in a milder but consistent degradation, *e.g.*, reducing ES to 91.0%, NS to 76.0%, and FQ to 0.54, indicating that softmax normalization could weaken knowledge-specific gradient information. Finally, except for *RO*, *DG* has the strongest negative impact, causing substantial degradation across all evaluation metrics. This suggests that raw gradients could contain correlated and redundant directions that hinder knowledge separation, further highlighting the importance of applying a factorized decomposition to identify input- and output-side directions from knowledge-induced gradients.

On Dictionary Sources and Filtering. This experiment examines how the knowledge source and atom filtering affect dictionary construction. Specifically, the dictionary can be constructed from either the full knowledge pool ($\mathcal{P}$) or the target knowledge ($\mathcal{T}$). When both $\mathcal{P}$ and $\mathcal{T}$ are used, the dictionaries constructed from merging them to form the final candidate[3]. To systematically evaluate these choices, different construction sources are considered, with and without atom filtering based on atom–gradient alignment over $\mathcal{T}$ and $\mathcal{R}$ As observed from Table 3, the dictionary source determines the coverage of available update directions. When

Table 3: Impact of knowledge sources and atom filtering. The **best** and second-best results are highlighted, while the row setting is adopted in the main experiments.

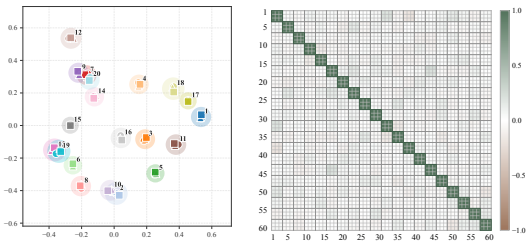
Source		Filtering		Editing (CounterFact)			Unlearning (TOFU)		
$\mathcal{P}$	$\mathcal{T}$	Off	On	ES ↑	PS ↑	NS ↑	FQ ↑	MU ↑	EM ↓
✓	×	✓	×	99.5	96.2	62.9	0.41	0.48	0.62
✓	×	×	✓	**99.9**	98.1	**83.3**	0.69	**0.66**	**0.59**
×	✓	✓	×	88.4	79.5	50.1	0.33	0.30	0.64
×	✓	×	✓	91.2	84.9	56.5	0.42	0.38	0.63
✓	✓	✓	×	99.7	97.0	64.4	0.44	0.51	**0.59**
✓	✓	×	✓	**99.9**	**98.3**	81.1	**0.70**	0.64	0.60

[3] A separate dictionary could also be constructed from the retained knowledge $\mathcal{R}$, enabling atom–atom similarity to identify and exclude atoms highly associated with retained knowledge. This extension is left for future work.

the dictionary is constructed only from $\mathcal{T}$, performance remains much lower than using $\mathcal{P}$ (even with filtering), suggesting that directions derived solely from target knowledge are insufficient to form an expressive space. Additionally, atom filtering serves to distinguish target-relevant directions from retained knowledge. For example, with the same $\mathcal{P}$-based dictionary, enabling filtering increases NS from 62.9 to 83.3 and MU from 0.48 to 0.66. Its effectiveness also depends on dictionary coverage: with a $\mathcal{T}$-only dictionary, filtering improves NS by only 6.4 points, compared with an 20.4-point gain when the dictionary is constructed from $\mathcal{P}$. Finally, combining $\mathcal{P}$ and $\mathcal{T}$ achieves similar performance to using $\mathcal{P}$ alone, indicating that $\mathcal{P}$ already sufficiently covers the directions required by $\mathcal{T}$. This also supports our design choice of constructing the dictionary from $\mathcal{P}$.

5.3 Qualitative Analysis: Knowledge Distinguishability in *KEY*

This section examines whether *KEY* can distinguish knowledge through task-specific addressing. In *KEY*, the sparse coefficient vector $\boldsymbol{\alpha}$ (Eq. (11)) provides a knowledge address over the dictionary. Accordingly, two questions are investigated, *i.e.*, ❶ **inter-knowledge separability**: whether different knowledge items exhibit distinguishable patterns on $\boldsymbol{\alpha}$; and ❷ **intra-knowledge consistency**: whether different question formulations of the same knowledge preserve consistent coefficient patterns, indicating stable knowledge addressing. Specifically, the same 20 knowledge items from Sec. 3 are used. For each item, *two* semantically-equivalent *question* formulations are constructed via prompting a LLM without changing the *answer*, with the detailed construction procedure provided in Appendix E.4. The dictionary is constructed using the full dataset, while each knowledge is independently addressed, yielding $20 \times 3 = 60$ coefficient vectors $\boldsymbol{\alpha}$. As shown in Fig. 5(a), $\boldsymbol{\alpha}$ are visually distinguishable across knowledge items, while different formulations of the same knowledge form compact groups. Consistently, Fig. 5(b) exhibits a clear block-diagonal structure: three formulations of the same knowledge item (indexed together in the plot) show consistently high pairwise similarity, whereas cross-knowledge similarities are largely concentrated around zero, with only weak positive or negative correlations. Furthermore, to quantitatively measure the knowledge overlap in our method, a normalized coefficient-overlap score is introduced as $\mathrm{Overlap}(\boldsymbol{\alpha}_i, \boldsymbol{\alpha}_j) = \frac{|\boldsymbol{\alpha}_i^\top \boldsymbol{\alpha}_j|}{\|\boldsymbol{\alpha}_i\|_2 \|\boldsymbol{\alpha}_j\|_2}$. The average overlap is only 0.051 for inter-knowledge pairs, compared with 0.976 for different formulations of the same knowledge. This quantitatively confirms inter-knowledge separability and intra-knowledge consistency. Detailed computation is provided in Appendix E.4. Overall, these results demonstrate strong *inter-knowledge separability* and *intra-knowledge consistency* in $\boldsymbol{\alpha}$. The coefficient interventions in Appendix E.6 further examine how these coefficients affect update efficacy and non-target knowledge preservation.



(a) Visualization of $\boldsymbol{\alpha}$ (t-SNE). (b) Similarity of $\boldsymbol{\alpha}$ (Same Knowledge).

Figure 5: Inter-Knowledge Separability and Intra-Knowledge Consistency of the proposed *KEY*.

Other analysis. *KEY* also achieves comparable time complexity compared to existing methods while achieving competitive empirical efficiency(Appendix E.5). Hyperparameter sensitivity analyses on the retained subspace dimensions and retention–sparsity trade-off are provided in Appendix E.3. Finally, Appendix E.8 discusses the dual-use risks of precise knowledge addressing and its potential for atom-level detection and defense.

6 Conclusion

This paper presents *KEY*, a knowledge-addressing framework for disentangled knowledge updating in LLMs. By constructing an orthogonal dictionary from knowledge-induced gradients and representing each update through sparse coefficients, *KEY* provides explicit and distinguishable knowledge addresses in the shared weight space. Theoretical analysis establishes conditions for stable addressing, while experiments across editing and unlearning demonstrate competitive performance and strong inter-knowledge separability and intra-knowledge consistency.

AI USE STATEMENT

In this work, generative AI tools are used to generate semantically equivalent paraphrases of TOFU questions for knowledge screening and the distinguishability analyses in Sec. 5.3 and Appendix E.4. All such paraphrases are required to preserve the original meaning and reference answer, and the prompt used to generate them is provided in Appendix B. Generative AI tools are not used for research ideation, method design, experimental design, or result analysis, and the remaining disclosure categories are not applicable to this work. We take responsibility for the final content of this work, including any text, claims, or artifacts produced with the aid of generative AI.

ETHICS STATEMENT

The goal of this work is to advance knowledge updating in LLMs, aiming to improve the precision and reliability of the information they contain. Our proposed method, *KEY*, is designed to edit or remove targeted knowledge while minimizing unintended interference with unrelated knowledge, which we believe is a step toward more controllable, interpretable, and trustworthy AI systems. Such improvements have positive applications in areas such as correcting factual inaccuracies, removing private, copyrighted, or harmful content, and keeping models up to date without costly retraining.

However, we acknowledge that, like any technology that allows the modification of information, knowledge updating tools could be misused. A malicious actor could potentially leverage such techniques to insert subtle misinformation, propaganda, or harmful biases into an LLM, and improved locality may make such edits harder to detect through general performance degradation alone (see Appendix. E.8). While our work focuses on improving the precision of knowledge updates, it does not in itself prevent such applications. At the same time, the explicit knowledge addresses produced by *KEY* may offer an interface for auditing updates before they are applied. We believe the broader research community must continue to develop robust detection mechanisms and safeguards in parallel with advances in knowledge updating techniques, and we encourage the responsible development and deployment of these technologies with a strong emphasis on ethical considerations and safeguards against misuse.

REPRODUCIBILITY STATEMENT

To ensure the reproducibility of our results, we provide detailed descriptions of our methodology and experimental setup throughout the paper. All experiments were conducted using publicly available Large Language Models: Llama-3-8B and GPT-J (6B) for knowledge editing, and Llama-2-7B and Llama-3.1-8B for knowledge unlearning. We evaluate our method on the widely used CounterFact and FINE-KED benchmarks for knowledge editing, and on TOFU and MUSE for knowledge unlearning.

Appendix. E.1 provides specific implementation details, including dataset configurations, dictionary construction settings, and key hyperparameters such as learning rates, regularization weights, optimizer settings, and the specific layers targeted for updating in each model. The complete procedure of *KEY* is summarized in Algorithm 1. To further aid reproducibility, we will make our source code publicly available on `https://anonymous.4open.science/r/KEY-3BCC/`.

REFERENCES

Tomer Ashuach, Dana Arad, Aaron Mueller, Martin Tutek, and Yonatan Belinkov. CRISP: Persistent concept unlearning via sparse autoencoders. In *Proceedings of the 64th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, July 2026.

Canyu Chen, Baixiang Huang, Zekun Li, Zhaorun Chen, Shiyang Lai, Xiongxiao Xu, Jia-Chen Gu, Jindong Gu, Huaxiu Yao, Chaowei Xiao, Xifeng Yan, William Yang Wang, Philip Torr, Dawn Song, and Kai Shu. Can editing llms inject harm? In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 40, pp. 30192–30200, 2026a.

Hang Chen, Jiaying Zhu, Xinyu Yang, and Wenya Wang. Clue: Conflict-guided localization for llm unlearning framework. In *International Conference on Learning Representations*, volume 2026, pp. 52000–52021, 2026b.

Xunlei Chen, Jinyu Guo, Yuang Li, Zhaokun Wang, Yi Gong, Jie Zou, Jiwei Wei, and Wenhong Tian. Alter: Asymmetric lora for token-entropy-guided unlearning of llms. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 40, pp. 35366–35374, 2026c.

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, 2022.

Chongyu Fan, Jiancheng Liu, Licong Lin, Jinghan Jia, Ruiqi Zhang, Song Mei, and Sijia Liu. Simplicity prevails: Rethinking negative preference optimization for llm unlearning. In *Advances in Neural Information Processing Systems*, volume 38, Main Conference, pp. 1540–1567. Curran Associates, Inc., 2025.

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 *International Conference on Learning Representations*, volume 2025, pp. 16366–16396, 2025.

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, 2021.

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, November 2024.

Jian Guan, Jesse Dodge, David Wadden, Minlie Huang, and Hao Peng. Language models hallucinate, but may excel at fact verification. In *Proceedings of the 2024 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (Volume 1: Long Papers)*, pp. 1090–1111, June 2024.

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*, November a.

Akshat Gupta, Dev Sajnani, and Gopala Anumanchipalli. A unified framework for model editing. In *Findings of the Association for Computational Linguistics: EMNLP 2024*, November b.

James V Haxby, M Ida Gobbini, Maura L Furey, Alumit Ishai, Jennifer L Schouten, and Pietro Pietrini. Distributed and overlapping representations of faces and objects in ventral temporal cortex. *Science*, 293(5539):2425–2430, 2001.

Jason Hoelscher-Obermaier, Julia Persson, Esben Kran, Ioannis Konstas, and Fazl Barez. Detecting edit failures in large language models: An improved specificity benchmark. In *Findings of the Association for Computational Linguistics*, July 2023.

Chenhui Hu, Pengfei Cao, Yubo Chen, Kang Liu, and Jun Zhao. Knowledge in superposition: unveiling the failures of lifelong knowledge editing for large language models. In *Proceedings of the Thirty-Ninth AAAI Conference on Artificial Intelligence and Thirty-Seventh Conference on Innovative Applications of Artificial Intelligence and Fifteenth Symposium on Educational Advances in Artificial Intelligence*, 2025.

Edward J Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, and Weizhu Chen. LoRA: Low-rank adaptation of large language models. In *International Conference on Learning Representations*, 2022.

Alexander G Huth, Wendy A De Heer, Thomas L Griffiths, Frédéric E Theunissen, and Jack L Gallant. Natural speech reveals the semantic maps that tile human cerebral cortex. *Nature*, 532 (7600):453–458, 2016.

Joel Jang, Dongkeun Yoon, Sohee Yang, Sungmin Cha, Moontae Lee, Lajanugen Logeswaran, and Minjoon Seo. Knowledge unlearning for mitigating privacy risks in language models. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 14389–14408, July 2023.

Hyunwoo Kim, Jack Hessel, Liwei Jiang, Peter West, Ximing Lu, Youngjae Yu, Pei Zhou, Ronan Bras, Malihe Alikhani, Gunhee Kim, Maarten Sap, and Yejin Choi. SODA: Million-scale dialogue distillation with social commonsense contextualization. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pp. 12930–12949, December 2023.

Xiaopeng Li, Shasha Li, Shezheng Song, Jing Yang, Jun Ma, and Jie Yu. Pmet: precise model editing in a transformer. In *Proceedings of the Thirty-Eighth AAAI Conference on Artificial Intelligence and Thirty-Sixth Conference on Innovative Applications of Artificial Intelligence and Fourteenth Symposium on Educational Advances in Artificial Intelligence*, 2024.

Chris Yuhao Liu, Yaxuan Wang, Jeffrey Flanigan, and Yang Liu. Large language model unlearning via embedding-corrupted prompts. In *The Thirty-eighth Annual Conference on Neural Information Processing Systems*, 2024.

Yinyi Luo, Zhexian Zhou, Hao Chen, Kai Qiu, Marios Savvides, Sharon Li, and Jindong Wang. Knowledgesmith: Uncovering knowledge updating in llms with model editing and unlearning. In *International Conference on Learning Representations*, volume 2026, pp. 117241–117279, 2026.

Pratyush Maini, Zhili Feng, Avi Schwarzschild, Zachary Chase Lipton, and J Zico Kolter. TOFU: A task of fictitious unlearning for LLMs. In *First Conference on Language Modeling*, 2024.

Kevin Meng, David Bau, Alex Andonian, and Yonatan Belinkov. Locating and editing factual associations in gpt. In *Proceedings of the 36th International Conference on Neural Information Processing Systems*, 2022.

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

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

Weijia Shi, Jaechan Lee, Yangsibo Huang, Sadhika Malladi, Jieyu Zhao, Ari Holtzman, Daogao Liu, Luke Zettlemoyer, Noah Smith, and Chiyuan Zhang. Muse: Machine unlearning six-way evaluation for language models. In *International Conference on Learning Representations*, volume 2025, pp. 27797–27818, 2025.

Bozhong Tian, Xiaozhuan Liang, Siyuan Cheng, Qingbin Liu, Mengru Wang, Dianbo Sui, Xi Chen, Huajun Chen, and Ningyu Zhang. To forget or not? towards practical knowledge unlearning for large language models. In *Findings of the Association for Computational Linguistics: EMNLP 2024*, pp. 1524–1537, November 2024.

Shiqi Wang, Qi Wang, Runliang Niu, He Kong, and Yi Chang. Microedit: Neuron-level knowledge disentanglement and localization in lifelong model editing. In *Proceedings of the 2025 Conference on Empirical Methods in Natural Language Processing*, pp. 33870–33884, 2025.

Xinwei Wu, Junzhuo Li, Minghui Xu, Weilong Dong, Shuangzhi Wu, Chao Bian, and Deyi Xiong. DEPN: Detecting and editing privacy neurons in pretrained language models. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, 2023.

Haoyu Xu, Pengxiang Lan, Enneng Yang, Guibing Guo, Jianzhe Zhao, Linying Jiang, and Xingwei Wang. Knowledge decoupling via orthogonal projection for lifelong editing of large language models. In *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 13194–13213, July 2025.

Min Xue and Artur Andrzejak. SVD as a fast interpretability method for transformers. In *Forty-third International Conference on Machine Learning*, 2026.

Wangli Yang, Jie Yang, Wanqing Li, and Yi Guo. Conclue: Conditional clue extraction for multiple choice question answering. In *International Conference on Document Analysis and Recognition*, pp. 183–198. Springer, 2024.

Wenhao Yu, Zhicong Lu, Bo Lv, Fangyin Ma, Kaiwen Wei, Shihao Yang, and Nayu Liu. Orthogonal representation editing: Decoupling semantic entanglement in batch knowledge editing of LLMs. In *Findings of the Association for Computational Linguistics*, July 2026.

Naixin Zhai, Pengyang Shao, Binbin Zheng, Yonghui Yang, Fei Shen, Long Bai, and Xun Yang. Maximizing local entropy where it matters: Prefix-aware localized llm unlearning. In *Proceedings of the 64th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 19514–19531, 2026.

Mengqi Zhang, Zisheng Zhou, Xiaotian Ye, Qiang Liu, Zhaochun Ren, Zhumin Chen, and Pengjie Ren. Disentangling knowledge representations for large language model editing. In *International Conference on Learning Representations*, volume 2026, pp. 81086–81111, 2026.

Ruiqi Zhang, Licong Lin, Yu Bai, and Song Mei. Negative preference optimization: From catastrophic collapse to effective unlearning. In *First Conference on Language Modeling*, 2024.

Zhange Zhang, Zhicheng Geng, Yuqing Ma, Tianbo Wang, Kai Lv, and Xianglong Liu. Conflict-aware knowledge editing in the wild: Semantic-augmented graph representation for unstructured text. In *Advances in Neural Information Processing Systems*, volume 38, Main Conference, pp. 158954–158979. Curran Associates, Inc., 2025.

Xiangyu Zhou, Yao Qiang, Saleh Zare Zade, Douglas Zytko, Prashant Khanduri, and Dongxiao Zhu. Not all tokens are meant to be forgotten. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 40, pp. 38173–38182, 2026.

APPENDIX CONTENTS

A Related Work **15**
A.1 Knowledge Updating . . . 15
A.2 Entanglement in Knowledge Updating . . . 15

B Can Entangled Knowledge Be Separated in Weight Space? **17**
B.1 Parameter Overlap Analysis . . . 17
B.2 Shared-Parameter Perturbation Analysis . . . 18
B.3 Gradient Direction Redundancy. . . . 19

C Method **19**
C.1 Derivation of the Reference Bases . . . 19
C.2 Projection Induced by the Dictionary . . . 20
C.3 Details of Atom Filtering . . . 21

D Bounded Proximity **22**
D.1 Global bounded proximity . . . 22
D.2 Local bounded proximity . . . 23

E Additional Experimental Results and Analyses **25**
E.1 Datasets and Task Construction . . . 25
E.2 Generalization . . . 26
E.3 Hyperparameter Analysis . . . 28
E.4 Knowledge Distinguishability Through Atom Usage . . . 30
E.5 Computational Complexity and Efficiency . . . 30
E.6 Intervention Analysis of α . . . 31
E.7 Knowledge Addressing in the Original Weight Space . . . 31
E.8 Discussion: Dual-Use Risks and Opportunities for Knowledge Addressing . . . 32

A Related Work

A.1 Knowledge Updating

The interpretation of feed-forward network(FFN) as associative key–value memories provides a foundation for understanding how knowledge can be modified within language models. Geva et al. (2021) show that the input projection of an FFN contains keys that respond to textual patterns, while the output projection contains corresponding value vectors that contribute to vocabulary predictions. The FFN output combines these values according to the activations induced by the input. Under the formulation in Eq. (1), the rows of W_{in}^{l} therefore act as memory keys, and the columns of W_{out}^{l} act as their associated values. This view connects knowledge updating to interventions on the parameters and activations involved in memory retrieval.

Building on this perspective, ROME (Meng et al., 2022) uses causal tracing to identify middle-layer FFNs that mediate factual recall. It treats the FFN output projection as a linear associative memory mapping an activated key representation to a value that supports the target prediction. ROME optimizes a target value and computes a rank-one weight update that maps the selected key to this value while constraining changes to existing associations. MEMIT (Meng et al., 2023) extends this approach to batch editing by incorporating multiple key–value associations and distributing their required changes across a set of FFN layers. Its update balances fitting the new associations against preserving existing mappings, enabling thousands of facts to be edited within the same model.

Subsequent methods refine target value estimation and FFN weight updating. PMET (Li et al., 2024) jointly optimizes attention and FFN hidden states, then uses the optimized FFN states to update FFN weights. EMMET (Gupta et al., b) unifies ROME and MEMIT under a preservation–memorization objective and extends equality-constrained editing to multiple associations. More recently, Gupta et al. (a) identify activation over-optimization and weight norm growth as sources of degradation during sequential editing. They introduce early stopping for activation optimization and a Frobenius norm constraint for weight updating to improve stability.

The FFN memory perspective also informs knowledge unlearning by suppressing the recall of selected information. DEPN (Wu et al., 2023) uses integrated gradients to identify FFN neurons associated with multi-token private information and sets their activations to zero to reduce privacy leakage. Using the KnowUnDo benchmark, Tian et al. (2024) show that existing methods can also cause unintended forgetting of knowledge that should be retained. These findings highlight the difficulty of forgetting target knowledge while preserving other knowledge.

Despite these advances, knowledge entanglement within shared FFN memories remains a challenge for selective updating. An FFN matrix supports multiple associations, so locating a layer or neuron involved in one fact does not establish its exclusive role in that fact. Hoelscher-Obermaier et al. (2023) show that ROME and MEMIT can introduce unintended associations and alter predictions on non-target queries, revealing specificity failures overlooked by standard evaluations. Similarly, Tian et al. (2024) find that methods targeting private or copyrighted information can also cause unintended forgetting of knowledge that should be retained. These findings suggest that locating and updating factual associations remains insufficient to establish knowledge-specific separation within shared parameters.

A.2 Entanglement in Knowledge Updating

Recent methods mitigate knowledge entanglement through representation-level separation, parameter-level isolation, and update-direction constraints.

Representation-level separation. DiKE (Zhang et al., 2026) decomposes subject representations into target-related and target-unrelated components through contrastive learning and knowledge constraints. It modifies the target-related component and incorporates preservation constraints on the unrelated component into the weight update. CRISP (Ashuach et al., 2026) uses sparse autoencoders to identify features activated more strongly by target concepts than by retained knowledge. It then fine-tunes the model to suppress these features in target contexts while preserving activations in retained contexts. ORE (Yu et al., 2026) constructs a general semantic subspace and constrains editing vectors to its orthogonal complement, reducing interference with shared semantic information. These methods reduce knowledge entanglement by separating hidden features or representation di-

rections. However, features associated with target knowledge may also support retained knowledge, and such shared roles receive less explicit attention.

Table 4: Comparison of methods for addressing knowledge entanglement. *Space* denotes where separation, isolation, or update constraints are imposed; it need not coincide with where the underlying representations are extracted. *Granularity* describes the knowledge units explicitly modeled or separated. For existing methods, operations follow their published formulations. SAE: sparse autoencoder.

Method	Space	Knowledge representation	Granularity	Strategy	Supported operations
DiKE (2026)	Activation	Subject representations	Item-level	Target / unrelated representation separation	Editing
CRISP (2026)	Activation	SAE feature activations	Concepts-level	Feature suppression with preservation	Unlearning
ORE (2026)	Activation	Hidden representations	Item-level	Semantic subspace projection	Editing
CLUE (2026b)	Parameter	Forget / retain circuits	Dataset-level	Conflict-guided localization	Unlearning
ALTER (2026c)	Parameter	Token entropy	Domain-level	Entropy-guided LoRA isolation	Unlearning
CAKE (2025)	Parameter	Knowledge graphs and fact embeddings	Item-level	Graph coloring; module isolation	Editing
AlphaEdit (2025)	Parameter	Key representations	Item-level	Null-space projection	Editing
KDE (2025)	Parameter	Input representations of prior edits	Item-level	Orthogonal gradient projection	Editing
KEY	Parameter	**Gradient coefficients in a shared basis**	Item-level	Dictionary decomposition; sparse updates	Editing; unlearning

Parameter-level isolation. CLUE (Chen et al., 2026b) identifies parameter components involved in forgetting and retention through circuit localization and logical conflict analysis. It distinguishes components that can be modified for forgetting from those whose modification also affects retained knowledge, then applies targeted fine-tuning with the corresponding preservation constraints. ALTER (Chen et al., 2026c) combines token entropy with asymmetric LoRA, using a shared matrix for high-entropy foundational tokens and separate matrices for specific forgetting and retention objectives. CAKE (Zhang et al., 2025) constructs a conflict graph using shared subjects and representation similarity, then applies graph coloring to assign conflicting edits to separate LoRA modules. These methods reduce interference by selecting parameter subsets or allocating separate modules. However, different knowledge items may still rely on overlapping parameters within each selected subset or module, and this remaining entanglement receives less explicit attention.

Update-direction constraints. AlphaEdit (Fang et al., 2025) constructs a null-space projector from the key representations of preserved knowledge and applies it to the weight update. Under the linear associative-memory formulation, the projected update leaves the corresponding key–value mappings unchanged. KDE (Xu et al., 2025) introduces an editable copy of the FFN value matrix and routes queries between the original and editable memories. It stores basis vectors derived from previously edited knowledge representations and projects new gradients onto their orthogonal complement. These limitations motivate a finer-grained approach to knowledge disentanglement that distinguishes how different knowledge items depend on shared parameters. Toward this end, *KEY* constructs a shared dictionary of weight-space directions from knowledge-induced gradients. Each knowledge item is expressed through its coefficients over these directions, enabling shared and

knowledge-specific contributions to be compared within the same parameters. Table 4 summarizes the differences between *KEY* and existing methods.

B Can Entangled Knowledge Be Separated in Weight Space?

Dataset. The experiments use TOFU (Maini et al., 2024), which contains synthetic question–answer pairs describing fictional authors. Each pair $c_i = \langle \boldsymbol{q}_i, \boldsymbol{a}_i \rangle$ is treated as one knowledge item, where $\boldsymbol{q}_i$ is the question and $\boldsymbol{a}_i$ is its ground-truth answer. An example is shown below.

Example of a TOFU Knowledge Item

Question. Has Jaime Vasquez taken part in any literary programs or workshops?

Answer. Yes, Jaime Vasquez is a regular at various literary festivals and often engages in workshops to nurture aspiring writers.

Candidate question–answer pairs from the full dataset are examined in a random order. For each candidate, ChatGPT generates two paraphrases of the original question, each preserving its meaning and requesting the same information. Before fine-tuning, the base model is evaluated on the original question and both paraphrases using the same reference answer. The prompt used for paraphrasing is provided below.

Example Prompt for Question Paraphrasing

Generate exactly two paraphrases of the question below. Each paraphrase must preserve the original meaning and request exactly the same information, so that the same reference answer remains valid. Keep all names, entities, and factual constraints unchanged. Vary the wording and sentence structure while keeping each question natural and self-contained. Do not introduce additional facts, answer hints, or new requests. Return only the two paraphrased questions as a numbered list.

Original question: Has Jaime Vasquez taken part in any literary programs or workshops?

For each of the three question formulations, the screening procedure measures target-token accuracy, mean target-token log-probability, and generation F1 against the reference answer, with generation limited to 128 new tokens. An item is treated as operationally unknown only if all three formulations satisfy the following conditions: target-token accuracy of at most 0.50, mean target-token log-probability of at most -1.0, and generation F1 of at most 0.30. The first 20 candidates meeting these criteria are selected and used throughout the parameter-overlap and perturbation analyses.

Independent Fine-Tuning. All runs start from the same `Meta-Llama-3-8B-Instruct` checkpoint, parameterized by W_0. For each knowledge item $c_i = \langle \boldsymbol{q}_i, \boldsymbol{a}_i \rangle$, the model is independently fine-tuned on its question–answer pair, yielding parameters W_i and the corresponding update $\Delta W_i = W_i - W_0$. The model and optimizer are reinitialized before each run, so that no learned parameters or optimizer states are carried over between knowledge items. Inputs are formatted using the model's chat template, with a maximum sequence length of 256 tokens and an answer-token limit of 96. Fine-tuning uses stochastic gradient descent (SGD) with a learning rate of 0.02 and gradient-norm clipping at 1.0. Each run is allowed a maximum of 200 optimization steps, with evaluation performed every 10 steps. Early stopping is enabled after a minimum of 20 steps. A successful evaluation requires a target-token accuracy of at least 0.95, a loss of at most 0.30, and a generation F1 score of at least 0.80. Training terminates once these conditions are satisfied at two consecutive evaluations, or when the maximum step budget is reached. Thus, all knowledge items share the same optimization settings and stopping criteria, while their actual training lengths may differ. The configuration specifies FP16 precision and enables gradient checkpointing. The experiment seed is set to 2026. Table 5 summarizes these settings.

B.1 Parameter Overlap Analysis

Identifying Updated Parameter Locations. After independently fine-tuning the model on each knowledge item c_i, the parameter update relative to the shared initialization W_0 is computed as

$\Delta W_i = W_i - W_0$. The analysis covers all model parameters. Let $\mathcal{P}$ denote the set of scalar parameter locations in the model, and let $\Delta W_i[p]$ denote the change at location $p \in \mathcal{P}$. A location is considered affected by learning knowledge c_i when the absolute parameter change is strictly greater than the threshold $\tau = 10^{-5}$. The affected parameter set is therefore defined as

$$S_i = \mathrm{supp}_\tau(\Delta W_i) = \{p \in \mathcal{P} : |\Delta W_i[p]| > \tau\}. \tag{15}$$

This is a strict element-wise absolute threshold. No top-k selection, percentile-based thresholding, or layer-wise normalization is applied. Consequently, membership in S_i depends only on whether the absolute change of an individual parameter τ exceeds 10^{-5}.

Table 5: Configuration for independent fine-tuning.

Base model	`Meta-Llama-3-8B-Instruct`		
Data and parameter scope		**Optimization**	
Knowledge items	20	Optimizer	SGD
Input format	Chat template	Learning rate	0.02
Maximum sequence length	256 tokens	Momentum	0
Maximum answer length	96 tokens	Weight decay	0
Trainable parameters	Full model	Gradient-norm clipping	1.0
Numerics and evaluation		**Training and early stopping**	
Master parameters	FP32	Minimum steps	20
FSDP computation	FP16	Maximum steps	200
Parameter differences	FP32	Evaluation interval	10 steps
Gradient checkpointing	Enabled	Early stopping	Enabled
Random seed	2026	Success patience	2 evaluations
Final generation F1	≥ 0.80	Early-stop criteria	Acc ≥ 0.95 & CE ≤ 0.30

Pairwise Overlap. Let $S_i = \mathrm{loc}_\tau(\Delta W_i)$ denote the affected parameter locations defined in the main text. For $N = 20$ knowledge items, we evaluate all $\binom{20}{2} = 190$ unordered pairs using Jaccard similarity. The metric depends only on parameter locations, regardless of the magnitudes or signs of their changes.

Visualization and Results. Fig. 2(a) visualizes the pairwise Jaccard similarities as a 20×20 heatmap. Diagonal entries are masked, and the color scale is capped at the largest observed off-diagonal value, with darker colors indicating greater overlap. The off-diagonal similarities span 11%–45%, consistent with the substantial parameter overlap reported in the main text.

B.2 Shared-Parameter Perturbation Analysis

This analysis uses the independently fine-tuned models $\{W_i\}_{i=1}^N$ only to identify the affected parameter sets $\{S_i\}_{i=1}^N$ defined in Appendix. B.1. For perturbation evaluation, W_0 is additionally fine-tuned jointly on all N knowledge items under the same settings, yielding a shared model W_{all}, around which all responses are measured.

Perturbation Construction. For each pair (c_i, c_j), the shared affected locations are $S_{ij} = S_i \cap S_j$, where S_i is obtained from $\Delta W_i = W_i - W_0$ using $\tau = 10^{-5}$. A zero-mean Gaussian perturbation Ξ_{ij} is sampled over S_{ij} as

$$\Xi_{ij}[p] = \begin{cases} \epsilon_p, & p \in S_{ij}, \\ 0, & p \notin S_{ij}, \end{cases} \qquad \epsilon_p \overset{\text{i.i.d.}}{\sim} \mathcal{N}(0, \sigma^2), \quad \sigma = 10^{-5}. \tag{16}$$

The same realization of Ξ_{ij}, including its entry-wise magnitudes and signs, is used to evaluate both knowledge items around W_{all}.

Target-Logit Evaluation. For $c_i = \langle \boldsymbol{q}_i, \boldsymbol{a}_i \rangle$, the reference answer is evaluated under teacher forcing. At answer position t, the model receives $(\boldsymbol{q}_i, \boldsymbol{a}_{i,<t})$. Let $\boldsymbol{z}_t(W; \boldsymbol{q}_i, \boldsymbol{a}_{i,<t})$ denote the corresponding vocabulary-logit vector. The target score is

$$s_i(W) = \frac{1}{|\boldsymbol{a}_i|} \sum_{t=1}^{|\boldsymbol{a}_i|} \left[\boldsymbol{z}_t(W; \boldsymbol{q}_i, \boldsymbol{a}_{i,<t}) \right]_{a_{i,t}} . \tag{17}$$

The perturbation response is $\delta s_i = s_i(W_{\text{all}} + \Xi_{ij}) - s_i(W_{\text{all}})$, with δs_j computed analogously using the same perturbation.

Response Comparison. Fig. 2(b) plots $(\delta s_i, \delta s_j)$ for each pair, where deviations from $y = x$ indicate different sensitivities and $\delta s_i \delta s_j < 0$ indicates opposite response directions. Because both responses are evaluated at the same W_{all} under the identical perturbation Ξ_{ij}, the comparison is not confounded by differences between independently fine-tuned model states.

B.3 Gradient Direction Redundancy.

To further examine whether knowledge-induced raw gradients contain correlated and redundant directions, we compute the pairwise directional similarity among the same $N = 20$ sampled knowledge items, yielding 190 distinct knowledge pairs. Specifically, each gradient G_i is normalized by its Frobenius norm, and the absolute cosine similarity between two knowledge gradients is defined as

$$C_{ij} = \frac{|\langle G_i, G_j \rangle_F|}{\|G_i\|_F \|G_j\|_F}, \qquad i \neq j, \tag{18}$$

where $\langle A, B \rangle_F$ is the Frobenius inner product of two matrices A and B. As shown in Fig. 6, the pairwise similarities are broadly distributed away from zero, with a mean of 0.18 ± 0.09 and a median of 0.16. Moreover, the distribution exhibits a substantial upper tail, with the 95th percentile reaching 0.35 and the maximum similarity reaching 0.47. Thus, directional alignment is not restricted to a few exceptional knowledge pairs; instead, different knowledge-induced gradients frequently share non-trivial parameter-space directions. Since the absolute similarity treats both aligned and oppositely aligned gradients as sharing the same underlying direction, these results indicate considerable directional correlation and redundancy among raw gradients. This provides empirical support for decomposing them into shared orthogonal input- and output-side directions, rather than directly treating individual gradients as independent dictionary atoms.

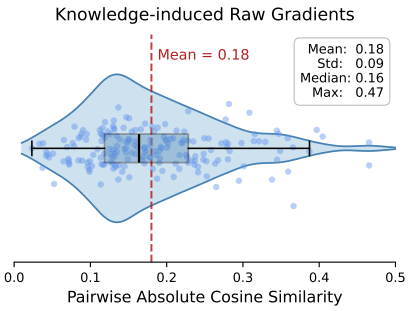


Figure 6: Distribution of pairwise absolute cosine similarities among knowledge-induced raw gradients.

C Method

C.1 Derivation of the Reference Bases

This section provides detailed derivations of the reconstruction objectives in Eq. (8) and their closed-form solutions via singular value decomposition (SVD). The derivation considers the input-side basis V; the output-side basis U follows analogously.

Reconstruction Objective. Let $V \in \mathbb{R}^{d_{\text{in}} \times r_{\text{in}}}$ satisfy $V^\top V = I$. Then $VV^\top$ is the orthogonal projector onto $\text{span}(V)$. The input-side reference basis is obtained by minimizing

$$V^\star = \arg\min_{V^\top V = I} \left\| A - AVV^\top \right\|_F^2 . \tag{19}$$

Expanding the squared Frobenius norm gives

$$\begin{aligned} \left\| A - AVV^\top \right\|_F^2 &= \text{tr}\left[(A - AVV^\top)^\top (A - AVV^\top) \right] \\ &= \|A\|_F^2 - 2\,\text{tr}(A^\top AVV^\top) + \text{tr}(VV^\top A^\top AVV^\top). \end{aligned} \tag{20}$$

Using $V^\top V = I$ and the cyclic property of the trace, both trace terms involving V reduce to $\mathrm{tr}(V^\top A^\top A V)$. Therefore,

$$\left\|A - AVV^\top\right\|_F^2 = \|A\|_F^2 - \mathrm{tr}(V^\top A^\top A V). \tag{21}$$

Since $\|A\|_F^2$ is independent of V, minimizing the reconstruction error is equivalent to

$$V^\star = \arg\max_{V^\top V = I} \mathrm{tr}(V^\top A^\top A V). \tag{22}$$

Closed-form SVD Solution. Let $\boldsymbol{v}_j$ denote the right singular vectors of A, ordered by non-increasing singular values $\sigma_1 \geq \sigma_2 \geq \cdots \geq 0$. These vectors satisfy

$$A^\top A \boldsymbol{v}_j = \sigma_j^2 \boldsymbol{v}_j, \tag{23}$$

and are therefore eigenvectors of $A^\top A$ with corresponding eigenvalues σ_j^2. By the Rayleigh–Ritz/Ky Fan theorem, Eq. (22) is maximized by the eigenvectors associated with the r_{in} largest eigenvalues. Accordingly, an optimal input-side basis is

$$V^\star = [\boldsymbol{v}_1, \ldots, \boldsymbol{v}_{r_{\text{in}}}], \tag{24}$$

with minimum reconstruction error

$$\left\|A - AV^\star V^{\star\top}\right\|_F^2 = \sum_{j > r_{\text{in}}} \sigma_j^2. \tag{25}$$

Thus, $V^\star$ preserves the dominant directions in the input-side factor matrix A. When $r_{\text{in}} \geq \mathrm{rank}(A)$, all directions associated with non-zero singular values are retained, yielding $A = AV^\star V^{\star\top}$ and exact reconstruction of the effective input-side subspace.

Output-side Basis. Applying the same derivation to the output-side factor matrix B yields

$$U^\star = [\boldsymbol{u}_1, \ldots, \boldsymbol{u}_{r_{\text{out}}}], \tag{26}$$

where $\boldsymbol{u}_j$ denotes the right singular vector of B associated with its j-th largest singular value $\tilde{\sigma}_j$. The corresponding reconstruction error is

$$\left\|B - BU^\star U^{\star\top}\right\|_F^2 = \sum_{j > r_{\text{out}}} \tilde{\sigma}_j^2. \tag{27}$$

When $r_{\text{out}} \geq \mathrm{rank}(B)$, all directions associated with non-zero singular values are retained, yielding $B = BU^\star U^{\star\top}$ and exact reconstruction of the effective output-side subspace.

C.2 Projection Induced by the Dictionary

The dictionary $\mathcal{D}$ constructed in Eq. (9) contains all pairwise combinations of the retained input- and output-side directions. For fixed bases $U^\star$ and $V^\star$, these atoms span the tensor-product subspace

$$\mathcal{S} = \mathrm{span}(\mathcal{D}) = \mathrm{span}\left\{\boldsymbol{u}_p \boldsymbol{v}_q^\top \;\middle|\; p \in [1, r_{\text{out}}], q \in [1, r_{\text{in}}]\right\}. \tag{28}$$

The following result characterizes the projection induced by this dictionary before atom filtering.

Proposition C.1 (Tensor-product Projection)**.** *For any matrix $G \in \mathbb{R}^{d_{\text{out}} \times d_{\text{in}}}$, its orthogonal projection onto $\mathcal{S}$ under the Frobenius inner product is*

$$\begin{aligned}\Pi_{\mathcal{S}}(G) &= U^\star U^{\star\top} G V^\star V^{\star\top} \\ &= \sum_{p=1}^{r_{\text{out}}} \sum_{q=1}^{r_{\text{in}}} \left\langle G, \boldsymbol{u}_p \boldsymbol{v}_q^\top \right\rangle_F \boldsymbol{u}_p \boldsymbol{v}_q^\top.\end{aligned} \tag{29}$$

Equivalently,

$$\Pi_{\mathcal{S}}(G) = \arg\min_{X \in \mathcal{S}} \|G - X\|_F^2. \tag{30}$$

Proof. Since the columns of $U^\star$ and $V^\star$ are orthonormal, the dictionary atoms satisfy

$$\left\langle \boldsymbol{u}_p \boldsymbol{v}_q^\top, \boldsymbol{u}_{p'} \boldsymbol{v}_{q'}^\top \right\rangle_F = (\boldsymbol{u}_p^\top \boldsymbol{u}_{p'})(\boldsymbol{v}_q^\top \boldsymbol{v}_{q'}) = \delta_{pp'} \delta_{qq'}. \tag{31}$$

These atoms therefore form an orthonormal basis of $\mathcal{S}$. Since $\langle G, \boldsymbol{u}_p \boldsymbol{v}_q^\top \rangle_F = \boldsymbol{u}_p^\top G \boldsymbol{v}_q$, their orthogonal expansion gives Eq. (29). The residual is orthogonal to $\mathcal{S}$, establishing the unique minimizer in Eq. (30). □

For a token-wise contribution $G_{i,t} = \rho_{i,t}\hat{\boldsymbol{g}}_{i,t}(\hat{\boldsymbol{x}}_{i,t})^\top$ in Eq. (6), the unit-norm factors give

$$\begin{aligned}\|G_{i,t} - \Pi_{\mathcal{S}}(G_{i,t})\|_F^2 &= \|G_{i,t}\|_F^2 - \|\Pi_{\mathcal{S}}(G_{i,t})\|_F^2 \\ &= \rho_{i,t}^2 \left(1 - \|U^{\star\top}\hat{\boldsymbol{g}}_{i,t}\|_2^2 \|V^{\star\top}\hat{\boldsymbol{x}}_{i,t}\|_2^2\right). \end{aligned} \tag{32}$$

Thus, the dictionary spans the joint subspace induced by the retained input- and output-side directions, while the approximation error of each token-wise contribution depends on how well its two factors are preserved by the corresponding bases.

C.3 Details of Atom Filtering

The dictionary construction in Sec. 4.1 is formulated over a generic knowledge collection $\mathcal{C}$, which can be instantiated as either the full knowledge pool $\mathcal{P}$ or the target knowledge $\mathcal{T}$. Given the resulting candidate dictionary $\mathcal{D} = \{D_m\}_{m=1}^M$, atom filtering prioritizes target-relevant directions while accounting for their relevance to retained knowledge when available.

Gradient-based Atom Relevance. For a knowledge item i, consider a small perturbation ηD_m along an atom D_m. A first-order expansion of its target score $s_i(W)$ gives

$$s_i(W + \eta D_m) \approx s_i(W) + \eta \left\langle \nabla_W s_i(W), D_m \right\rangle_F. \tag{33}$$

The magnitude $|\langle \nabla_W s_i(W), D_m \rangle_F|$ therefore characterizes the first-order sensitivity of knowledge item i to perturbations along D_m. Accordingly, when $\mathcal{R} \neq \varnothing$, the average relevance of each atom to the target and retained knowledge is defined as

$$F_m = \frac{1}{|\mathcal{T}|} \sum_{i \in \mathcal{T}} |\langle \nabla_W s_i(W), D_m \rangle_F|, \qquad R_m = \frac{1}{|\mathcal{R}|} \sum_{i \in \mathcal{R}} |\langle \nabla_W s_i(W), D_m \rangle_F|. \tag{34}$$

Here, F_m measures the association with target knowledge, whereas R_m measures the association with knowledge expected to remain unchanged.

Score Normalization. The two relevance scores are normalized using the 90-th percentiles of their respective positive values:

$$\tilde{F}_m = \frac{F_m}{Q_{0.9}(\{F_j \mid F_j > 0\}) + \epsilon}, \qquad \tilde{R}_m = \frac{R_m}{Q_{0.9}(\{R_j \mid R_j > 0\}) + \epsilon}, \tag{35}$$

where $Q_p(\cdot)$ denotes the $100p$-th percentile and $\epsilon > 0$ is a small constant for numerical stability. When a set of positive scores is empty, its percentile is set to zero before adding ϵ.

Filtering Score and Atom Selection. The filtering score is defined as the stabilized target-to-retain relevance ratio:

$$S_m = \frac{\tilde{F}_m}{\tilde{R}_m + \epsilon}. \tag{36}$$

Higher scores favor atoms with stronger target relevance relative to retained-knowledge relevance. Up to N_{atom} atoms are retained to form the compact dictionary:

$$\mathcal{I} = \text{Top}_{\min\{N_{\text{atom}}, M\}}\{S_m\}_{m=1}^M, \qquad \mathcal{D}^\star = \{D_m \mid m \in \mathcal{I}\}, \tag{37}$$

where Top_k returns the indices of the k highest scores, and N_{atom} specifies the maximum number of retained atoms per weight matrix. All candidates are retained when $M \leq N_{\text{atom}}$.

Case without Retained Knowledge. When $\mathcal{R} = \varnothing$, the retained-knowledge term is omitted by setting $\tilde{R}_m = 1$ for all atoms. Equation (36) then reduces to

$$S_m = \tilde{F}_m. \tag{38}$$

In this case, atoms are ranked solely according to their relevance to the target knowledge. Unless otherwise specified, filtering is applied independently to each selected weight matrix before sparse knowledge addressing.

D BOUNDED PROXIMITY

Notation and correspondence with the method. We use simplified notation corresponding to Sec. 4. Enumerate the filtered dictionary as $\mathcal{D}^\star = \{D_{m_1}, \dots, D_{m_n}\}$, where $n = |\mathcal{D}^\star|$, and define $D = [d_1, \dots, d_n] \in \mathbb{R}^{m \times n}$, $d_j = \mathrm{vec}(D_{m_j})$, $b = \mathrm{vec}(W)$, $z = \boldsymbol{\alpha}$. Here each matrix valued atom becomes a column of D. Eq. (11) therefore becomes

$$w = \mathrm{vec}(W_{\boldsymbol{\alpha}}) = b + Dz.$$

For multiple layers, concatenate the vectorized editable weights and embed each atom in its corresponding layer block, with zeros elsewhere. In either case, Frobenius orthonormality of the atoms implies $D^\top D = I$.

For a single target item, let $x = \mathrm{vec}(\boldsymbol{q}_i)$, where $\boldsymbol{q}_i$ denotes the *continuous query embeddings*. For multiple target items, x concatenates their vectorized query embeddings. Thus x is distinct from the intermediate activation $\boldsymbol{x}_{i,t}$. We fix the sequence lengths, attention masks, answer sequences (including their target vocabulary indices), update type, retained knowledge, original weights, filtered dictionary, and regularization weights. Only the query embeddings vary. Define

$$g(z, x) = \ell(b + Dz, x), \; F(z, x) = g(z, x) + \lambda \|z\|_1,$$

where ℓ denotes the smooth part $\mathcal{L}_{\mathrm{upd}} + \lambda_{\mathrm{ret}} \mathcal{L}_{\mathrm{ret}}$ of Eq. (14), and $\lambda = \lambda_{\mathrm{sel}} > 0$. An optimized coefficient vector $a(x)$ therefore corresponds to the value of $\boldsymbol{\alpha}$ for query embeddings x.

The bounds below concern perturbations of continuous query embeddings with other quantities fixed. They do not directly compare tasks with different target answers or different filtered dictionaries. We write $x \in \mathcal{X} \subseteq \mathbb{R}^d$, fix $q \in [1, \infty]$, and use $\| \cdot \|_{q \to 2}$ for the induced operator norm. The scalar q specifying this norm is unrelated to the query $\boldsymbol{q}_i$.

D.1 GLOBAL BOUNDED PROXIMITY

Theorem D.1 (Global bounded proximity)**.** *Suppose a global minimizer*

$$a(x) \in \operatorname*{argmin}_{z \in \mathbb{R}^n} F(z, x)$$

exists for every $x \in \mathcal{X}$. Assume $g(\cdot, x)$ is continuously differentiable and satisfies:

1. *Uniform strong monotonicity. For some $\mu > 0$,*

$$\langle \nabla_z g(z_1, x) - \nabla_z g(z_2, x), z_1 - z_2 \rangle \geq \mu \|z_1 - z_2\|_2^2 \tag{39}$$

   for all $z_1, z_2 \in \mathbb{R}^n$ and $x \in \mathcal{X}$.

2. *Uniform input regularity. For some $L_x < \infty$,*

$$\|\nabla_z g(z, x_1) - \nabla_z g(z, x_2)\|_2 \leq L_x \|x_1 - x_2\|_q \tag{40}$$

   for all $z \in \mathbb{R}^n$ and $x_1, x_2 \in \mathcal{X}$.

Then the global minimizer is unique and

$$\|a(x_1) - a(x_2)\|_2 \leq \frac{L_x}{\mu} \|x_1 - x_2\|_q, \qquad x_1, x_2 \in \mathcal{X}.$$

Proof. Strong monotonicity of $\nabla_z g(\cdot, x)$ implies μ-strong convexity of $g(\cdot, x)$. Adding the convex ℓ_1 penalty preserves strong convexity, so the minimizer is unique. Write $a_i = a(x_i)$ and $\Delta = a_1 - a_2$. Optimality provides $s_i \in \partial \|a_i\|_1$ such that

$$\nabla_z g(a_i, x_i) + \lambda s_i = 0, \qquad i = 1, 2.$$

Subtracting these identities and taking the inner product with Δ gives

$$\langle \nabla_z g(a_1, x_1) - \nabla_z g(a_2, x_2), \Delta \rangle + \lambda \langle s_1 - s_2, \Delta \rangle = 0.$$

Monotonicity of the convex subdifferential, *i.e.* $\partial\|\cdot\|_1$, gives $\langle s_1 - s_2, \Delta\rangle \geq 0$ and hence

$$\langle \nabla_z g(a_2, x_2) - \nabla_z g(a_1, x_1), \Delta\rangle \geq 0. \tag{41}$$

Consequently,

$$\begin{aligned}\mu\|\Delta\|_2^2 &\leq \langle \nabla_z g(a_1, x_1) - \nabla_z g(a_2, x_1), \Delta\rangle \quad \text{(using (39))}\\ &\leq \langle \nabla_z g(a_2, x_2) - \nabla_z g(a_2, x_1), \Delta\rangle \quad \text{(adding (41))}\\ &\leq L_x\|x_1 - x_2\|_q\|\Delta\|_2. \quad \text{(using Cauchy-Schwarz and (40))}\end{aligned}$$

If $\Delta = 0$, the result is immediate. Otherwise, division by $\mu\|\Delta\|_2$ proves the bound. □

When ℓ is twice continuously differentiable, the chain rule gives $\nabla^2_{zz} g(z, x) = D^\top \nabla^2_{ww}\ell(b + Dz, x)D$, and $\nabla^2_{zx} g(z, x) = D^\top \nabla^2_{wx}\ell(b + Dz, x)$. Thus $\nabla^2_{zz} g(z, x) \succeq \mu I$ uniformly in z, x implies (39). If $\mathcal{X}$ is convex, the uniform bound $\|\nabla^2_{zx} g(z, x)\|_{q\to 2} \leq L_x$ implies (40) by integration along the segment joining x_1 and x_2. These global conditions need not hold for the neural network loss. The following result requires positive curvature only on the active coordinates near a local minimizer.

D.2 Local bounded proximity

Theorem D.2 (Local bounded proximity)**.** *Let a_0 be a local minimizer of $F(\cdot, x_0)$, and define*

$$S = \operatorname{supp}(a_0),\ s_S = \operatorname{sign}((a_0)_S).$$

Assume g satisfies the following:

1. *Local smoothness. The function g is twice continuously differentiable on an open neighborhood of (a_0, x_0).*
2. *Positive active curvature. The active Hessian satisfies*
$$H_S = \nabla^2_{SS} g(a_0, x_0) \succ 0. \tag{42}$$
3. *Strict complementarity. The inactive coordinates satisfy*
$$\|\nabla_{S^c} g(a_0, x_0)\|_\infty < \lambda. \tag{43}$$

For nonempty S, there exist a convex neighborhood U of x_0, a neighborhood V of a_0, and a continuously differentiable map $a : U \to V$, with $a(x_0) = a_0$, such that $a(x)$ is the unique stationary point in V and a strict local minimizer of $F(\cdot, x)$ for every $x \in U$. Here stationarity means $0 \in \nabla_z g(a(x), x) + \lambda\partial\|a(x)\|_1$. Moreover, $\operatorname{supp}(a(x)) = S$, $\operatorname{sign}(a_S(x)) = s_S$, *and*

$$\frac{\partial a_S}{\partial x}(x) = -\left[\nabla^2_{SS} g(a(x), x)\right]^{-1}\nabla^2_{Sx} g(a(x), x),\ \frac{\partial a_{S^c}}{\partial x}(x) = 0. \tag{44}$$

The neighborhood U can be chosen such that, for constants $\mu_S > 0$ and $M_{Sx} < \infty$,

$$\nabla^2_{SS} g(a(x), x) \succeq \mu_S I,\ \|\nabla^2_{Sx} g(a(x), x)\|_{q\to 2} \leq M_{Sx}, \qquad x \in U. \tag{45}$$

Consequently,

$$\|a(x_1) - a(x_2)\|_2 \leq \frac{M_{Sx}}{\mu_S}\|x_1 - x_2\|_q, \qquad x_1, x_2 \in U.$$

If $S = \varnothing$, the corresponding branch is $a(x) = 0$ and its Lipschitz constant is zero. Conditions on empty coordinate sets are vacuous.

Proof. First suppose $S \neq \varnothing$. Local optimality gives

$$0 \in \nabla_z g(a_0, x_0) + \lambda\partial\|a_0\|_1,\ \nabla_S g(a_0, x_0) + \lambda s_S = 0.$$

For $u \in \mathbb{R}^{|S|}$, define $z(u)_S = u$, $z(u)_{S^c} = 0$, and

$$\Phi(u, x) = \nabla_S g(z(u), x) + \lambda s_S.$$

Then

$$\Phi((a_0)_S, x_0) = 0, \qquad \nabla_u \Phi((a_0)_S, x_0) = H_S.$$

By (42), H_S is invertible. The implicit function theorem therefore gives a unique local C^1 solution $u(x)$ of $\Phi(u(x), x) = 0$. Set $a_S(x) = u(x)$ and $a_{S^c}(x) = 0$.

Since $(a_0)_S$ has no zero coordinates, continuity preserves the active signs in a sufficiently small neighborhood. By (42), (43), and continuity, we may choose a convex neighborhood U of x_0 and a product of convex coordinate neighborhoods $V = V_S \times V_{S^c}$ of a_0 such that

$$\operatorname{sign}(z_S) = s_S, \qquad \nabla^2_{SS} g(z, x) \succeq \mu_S I, \qquad \|\nabla_{S^c} g(z, x)\|_\infty \le \lambda - \eta \tag{46}$$

for all $(z, x) \in V \times U$, where $\mu_S, \eta > 0$. Shrink U further if necessary so that $a(x) \in V$.

The equation $\Phi(a_S(x), x) = 0$ and (46) give the full stationarity conditions at $a(x)$. Conversely, any stationary point $z \in V$ must satisfy $z_{S^c} = 0$: if $z_j \neq 0$ for some $j \in S^c$, stationarity requires $|\nabla_j g(z, x)| = \lambda$, contradicting (46). Its active coordinates then solve $\Phi(z_S, x) = 0$. Local uniqueness from the implicit function theorem therefore makes $a(x)$ the unique stationary point in V.

To prove strict local minimality, fix $x \in U$ and $z \in V$, and set $\widetilde{z}_S = z_S$, $\widetilde{z}_{S^c} = 0$. The segment $z_t = \widetilde{z} + t(z - \widetilde{z})$ lies in V, so

$$\begin{aligned} F(z, x) - F(\widetilde{z}, x) &= \int_0^1 \langle \nabla_{S^c} g(z_t, x), z_{S^c} \rangle \, dt + \lambda \|z_{S^c}\|_1 \\ &\ge -(\lambda - \eta)\|z_{S^c}\|_1 + \lambda \|z_{S^c}\|_1 \quad \text{(using Hölder's inequality and (46))} \\ &= \eta \|z_{S^c}\|_1. \end{aligned}$$

On the active slice, $\|z_S\|_1 = s_S^\top z_S$, so the restricted objective has Hessian $\nabla^2_{SS} g$. Strong convexity and active stationarity give

$$\begin{aligned} F(\widetilde{z}, x) - F(a(x), x) &\ge \langle \nabla_S g(a(x), x) + \lambda s_S, z_S - a_S(x) \rangle + \frac{\mu_S}{2} \|z_S - a_S(x)\|_2^2 \\ &= \frac{\mu_S}{2} \|z_S - a_S(x)\|_2^2. \end{aligned}$$

Adding the two inequalities yields

$$F(z, x) - F(a(x), x) \ge \eta \|z_{S^c}\|_1 + \frac{\mu_S}{2} \|z_S - a_S(x)\|_2^2 > 0$$

whenever $z \neq a(x)$. Thus $a(x)$ is a strict local minimizer.

Differentiating $\Phi(a_S(x), x) = 0$ gives

$$\nabla^2_{SS} g(a(x), x) \frac{\partial a_S}{\partial x}(x) + \nabla^2_{Sx} g(a(x), x) = 0.$$

Inverting the active Hessian proves (44); the inactive derivative is zero because $a_{S^c}(x) = 0$. Continuity gives a finite bound M_{Sx} on a smaller convex neighborhood U, establishing (45). Writing $J_a(x) = \partial a(x)/\partial x$, we obtain

$$\begin{aligned} \|J_a(x)\|_{q \to 2} &\le \| \left[\nabla^2_{SS} g(a(x), x)\right]^{-1} \|_{2 \to 2} \|\nabla^2_{Sx} g(a(x), x)\|_{q \to 2} \quad \text{(using (44))} \\ &\le \frac{M_{Sx}}{\mu_S}. \quad \text{(using (45))} \end{aligned} \tag{47}$$

For $x_1, x_2 \in U$, let $x_t = x_2 + t(x_1 - x_2)$. Convexity of U gives $x_t \in U$, and hence

$$\begin{aligned} \|a(x_1) - a(x_2)\|_2 &= \left\| \int_0^1 J_a(x_t)(x_1 - x_2) \, dt \right\|_2 \quad \text{(fundamental theorem of calculus)} \\ &\le \int_0^1 \|J_a(x_t)\|_{q \to 2} \|x_1 - x_2\|_q \, dt \\ &\le \frac{M_{Sx}}{\mu_S} \|x_1 - x_2\|_q. \quad \text{(using (47))} \end{aligned}$$

If $S = \varnothing$, strict complementarity and continuity give $\|\nabla_z g(z, x)\|_\infty \le \lambda - \eta$ near $(0, x_0)$. The same inactive-coordinate argument gives $F(z, x) - F(0, x) \ge \eta \|z\|_1$ and excludes every nonzero nearby stationary point. Thus $a(x) = 0$ is the unique nearby stationary point and a strict local minimizer, with Lipschitz constant zero. □

Remark D.3 (Interpretation of the local constants)**.** *Let D_S contain the columns of D indexed by S. The chain rule gives*

$$\nabla^2_{SS} g(z, x) = D_S^\top \nabla^2_{ww} \ell(b + Dz, x) D_S,$$
$$\nabla^2_{Sx} g(z, x) = D_S^\top \nabla^2_{wx} \ell(b + Dz, x).$$

Thus μ_S measures curvature along the active dictionary directions, while M_{Sx} bounds the sensitivity of their gradients to query-embedding perturbations. A separate active-coordinate margin assumption is unnecessary: $\min_{j \in S} |(a_0)_j| > 0$ holds for every finite nonempty support.

Remark D.4 (Scope and consequence)**.** *Orthonormality and ℓ_1 regularization alone do not imply the curvature assumptions. The local theorem concerns the branch through a_0; it does not ensure that an optimization algorithm selects this branch or that global minimizers remain on it. Its differentiability argument requires fixed support and signs, whereas the global theorem permits support changes.*

For the fixed dictionary, $D^\top D = I$ gives

$$\|(b + Da(x_1)) - (b + Da(x_2))\|_2 = \|D(a(x_1) - a(x_2))\|_2 = \|a(x_1) - a(x_2)\|_2.$$

Thus the same bounds hold for the represented parameter updates. For a single target item and $q = 2$,

$$\|x_1 - x_2\|_2 = \|\boldsymbol{q}^{(1)} - \boldsymbol{q}^{(2)}\|_F,$$

where $\boldsymbol{q}^{(1)}$ and $\boldsymbol{q}^{(2)}$ are the query-embedding matrices. The results establish stability under query-embedding perturbations with fixed answers and dictionary; they do not by themselves establish semantic disentanglement or knowledge preservation.

E Additional Experimental Results and Analyses

E.1 Datasets and Task Construction

We evaluate *KEY* on four benchmarks covering knowledge editing and knowledge unlearning. Beyond the benchmark configurations, we explicitly distinguish the data used to instantiate the knowledge pool $\mathcal{P}$, target knowledge $\mathcal{T}$, and retained knowledge $\mathcal{R}$ from data used only for evaluation. Table 6 summarizes the corresponding data-access protocol. Unless otherwise stated, the dictionary is constructed once from $\mathcal{P}$ and reused across update requests; for each request, atom filtering and coefficient optimization access only the corresponding $\mathcal{T}$ and $\mathcal{R}$.

TOFU (Maini et al., 2024). directly provides question–answer pairs and therefore naturally matches the formulation $\langle \boldsymbol{q}_i, \boldsymbol{a}_i \rangle$ used by *KEY*. For each forgetting configuration, $\mathcal{T}$ is given by the corresponding forget split, i.e., forget01, forget05, or forget10, and the original answers are treated as the knowledge to be suppressed. The retained set $\mathcal{R}$ is constructed from the corresponding retain split and is used for retain-aware atom filtering and coefficient optimization. The standard TOFU evaluation protocol is subsequently used to compute FQ, MU, and EM. Any overlap between the retained examples accessible during optimization and those used for utility evaluation is therefore treated explicitly as supervised retention rather than as held-out evaluation.

MUSE (Shi et al., 2025). differs from the QA-based benchmarks because the knowledge to be unlearned is provided as continuous text rather than explicit question–answer pairs. To instantiate the formulation used by *KEY*, a tokenized sequence $\boldsymbol{x} = (x_1, \ldots, x_L)$ is decomposed into autoregressive prediction pairs $\langle \boldsymbol{q}_t, \boldsymbol{a}_t \rangle = \langle x_{<t}, x_t \rangle$, $t = 1, \ldots, L$, where $\boldsymbol{q}_t$ denotes the prefix preceding token x_t, and $\boldsymbol{a}_t = x_t$ is the corresponding next-token target. Knowledge-induced gradients are computed over the valid answer-token positions under this formulation. The target set $\mathcal{T}$ is constructed from the forget corpus, whereas $\mathcal{R}$ is constructed from the corresponding retain corpus. The benchmark-specific evaluation data used to compute KM_r, KM_f, VM_f, and PL are handled according to the access protocol summarized in Table 6.

CounterFact (Meng et al., 2022). contains an original factual association together with a counterfactual replacement for each editing request. For dictionary construction, the knowledge pool $\mathcal{P}$ is formed from the original factual mappings: the formatted rewrite prompt is used as the query $\boldsymbol{q}_i$, while the corresponding original answer is used as $\boldsymbol{a}_i$. The counterfactual replacement answer $\boldsymbol{a}_i^+$ is not used to represent the original knowledge in $\mathcal{P}$; it is introduced when the corresponding item

becomes part of $\mathcal{T}$ and is used in the target-specific atom scoring and update objective. For each edit, $\mathcal{R}$ is constructed from non-target knowledge according to the retain protocol used in our implementation. The rewrite, paraphrase, and neighborhood queries are used to compute ES, PS, and NS, respectively. Their accessibility during optimization is reported explicitly in Table 6.

FINE-KED (Zhang et al., 2026). represents each factual item as a subject–relation–object triple (s, r, o). We convert each original fact into a query–answer pair $\langle \boldsymbol{q}_i, \boldsymbol{a}_i \rangle$ and use these pairs to form $\mathcal{P}$. For an editing request, $\mathcal{T}$ contains the target fact together with its prescribed replacement answer, which is used in the update objective. The retained set $\mathcal{R}$ is constructed from non-target factual knowledge according to the same retain protocol. The Easy, Middle, and Hard subsets contain same-subject but different-relation facts and are used to evaluate relational locality (RL). Whether these locality examples are accessible during optimization is specified in Table 6.

Table 6: Data protocol for dictionary construction, task-specific updating, and evaluation. "Eval.-only" denotes data that are used only for final evaluation and are not accessed during dictionary construction, atom filtering, or coefficient optimization.

Dataset	$\mathcal{P}$	$\mathcal{T}$	$\mathcal{R}$	**Evaluation data**
CounterFact	Original rewrite queries paired with the original answers.	Target edit requests with replacement answers.	Non-target knowledge sampled according to the retain protocol.	Rewrite queries for ES; paraphrase queries for PS; neighborhood queries for NS.
FINE-KED	Original factual query–answer pairs.	Target facts together with their prescribed replacement answers.	Non-target factual knowledge selected according to the retain protocol.	Target facts for Eff.; Easy, Middle, and Hard related facts for RL.
TOFU	Question–answer pairs from the corresponding TOFU knowledge pool.	The prescribed forget set, i.e., forget01, forget05, or forget10.	Knowledge from the corresponding retain set.	Standard TOFU evaluation data for FQ, MU, and EM.
MUSE	Token-level autoregressive prediction pairs constructed from raw text.	Token-level pairs derived from the forget corpus.	Token-level pairs derived from the retain corpus.	Benchmark data for KM_r, KM_f, VM_f, and PL.

Data Access Across Stages. Unless otherwise specified, the dictionary is constructed once from the benchmark-specific knowledge pool $\mathcal{P}$ and subsequently reused across update requests. For a particular request, $\mathcal{T}$ is used to compute target-specific atom relevance and the update objective, while $\mathcal{R}$ is used for retain-aware atom filtering and $\mathcal{L}_{\mathrm{ret}}$. Evaluation-only questions or answers do not affect the learned coefficient vector $\boldsymbol{\alpha}$ unless they are explicitly identified as supervised retention data. This distinction specifies the information available to *KEY* at each stage without requiring all benchmark subsets to be mutually disjoint.

Implementation and Hyperparameter Settings. For knowledge unlearning, we evaluate Llama-2-7B and Llama-3.1-8B. For knowledge editing, we use Llama-3-8B and GPT-J-6B. All experiments are conducted on NVIDIA A100 GPUs with 80GB memory. Unless otherwise specified, results are averaged over five independent runs.

For *KEY*, the dictionary is constructed from the full knowledge pool $\mathcal{P}$. We retain $r_{\mathrm{in}} = r_{\mathrm{out}} = 48$ input- and output-side basis directions and form rank-one candidates through all pairwise combinations. Atom filtering uses a budget of $N_{\mathrm{atom}} = 1024$ per weight matrix, retaining the $\min\{N_{\mathrm{atom}}, |\mathcal{D}|\}$ highest-scoring atoms for coefficient optimization. The same default dictionary configuration is used across the main editing and unlearning experiments unless otherwise stated. CounterFact Avg. is the arithmetic mean of ES, PS, and NS for all methods. FINE-KED Avg.(RL) is weighted by the evaluated sample counts of the Easy, Middle, and Hard subsets. Table 7 summarizes the implementation settings.

Reproducibility. Algorithm 1 summarizes the complete procedure. The dictionary is constructed once from the knowledge pool and subsequently reused for individual editing or unlearning requests. For each target set, atoms are screened according to their relevance to target knowledge and interference with retained knowledge, after which only the coefficient vector $\boldsymbol{\alpha}$ is optimized.

E.2 Generalization

To further evaluate the generalization of *KEY* beyond the settings reported in the main experiments, we extend the unlearning evaluation to different forgetting ratios and data domains. Specifically, in addition to the TOFU 5% setting used in the main text, we evaluate on the 1% and 10% forget sets.

Table 7: Default implementation and hyperparameter settings of *KEY*.

Dictionary construction		**Coefficient optimization**	
Dictionary source	$\mathcal{P}$	Optimizer(Edit / Forget)	`Adam/AdamW`
Input rank r_{in}	48	Learning rate	0.03
Output rank r_{out}	48	Epochs / steps	200 steps
Candidate pairing	All pairs	Batch size	2
Atom budget N_{atom}	1024	λ_{ret}	0.6
Gradient signal	Pre-softmax logit	λ_{sel}	0.4
Atom filtering	Enabled	Gradient clipping	1.0
Model and numerics		**Evaluation**	
GPU	NVIDIA A100 80 GB	Independent runs	5
Computation precision	FP16	Random seeds	[2026,...,2030]
Maximum sequence length	256	Editing datasets	CounterFact, FINE-KED
Updated layers	All layers	Unlearning datasets	TOFU, MUSE
Trainable variables	Coefficients $\boldsymbol{\alpha}$	Reported statistics	Mean $\pm$ Std.

Require: Base model parameters $\boldsymbol{\theta}$; knowledge pool $\mathcal{P}$; target knowledge $\mathcal{T}$; retained knowledge $\mathcal{R}$; ranks $r_{\text{in}}, r_{\text{out}}$; atom budget N_{atom}; weights $\lambda_{\text{ret}}, \lambda_{\text{sel}}$
Ensure: Updated model parameters $\boldsymbol{\theta}'$
// Stage 1: Knowledge addressing dictionary construction
for each editable weight matrix $W \in W_{\text{full}}$ **do**
  Compute token-wise gradient factors $(\hat{\boldsymbol{x}}_k, \hat{\boldsymbol{g}}_k, \rho_k)$ from $\mathcal{P}$ and stack them into A and B (Eqs. (6)–(7))
  Obtain the top-r_{in} input-side bases $V^\star = [\boldsymbol{v}_1, \ldots, \boldsymbol{v}_{r_{\text{in}}}]$ from A and the top-r_{out} output-side bases $U^\star = [\boldsymbol{u}_1, \ldots, \boldsymbol{u}_{r_{\text{out}}}]$ from B via SVD (Eq. (8))
  Construct candidate atoms $\mathcal{D} = \{D_m = \boldsymbol{u}_p \boldsymbol{v}_q^\top\}_{m=1}^{M}, p \in [1, r_{\text{out}}], q \in [1, r_{\text{in}}]$ (Eq. (9))
  Compute target and retain relevance F_m, R_m (Eq. (10)) and the filtering score $S_m = \bar{F}_m / \bar{R}_m$
  Retain the top-N_{atom} atoms by S_m to obtain $\mathcal{D}_W^\star \subseteq \mathcal{D}$
end for
// Stage 2: Task-specific sparse addressing
Initialize coefficient vector $\boldsymbol{\alpha}$
while not converged **do**
  Construct $\Delta W_{\boldsymbol{\alpha}} = \sum_{D_m \in \mathcal{D}_W^\star} \alpha_m D_m$ for each $W \in W_{\text{full}}$ (Eq. (11))
  Update $\boldsymbol{\alpha}$ by minimizing $\mathcal{L}_{\text{upd}}(\boldsymbol{\alpha}) + \lambda_{\text{ret}} \mathcal{L}_{\text{ret}}(\boldsymbol{\alpha}) + \lambda_{\text{sel}} \|\boldsymbol{\alpha}\|_1$ (Eq. (14))
end while
Apply $W \leftarrow W + \Delta W_{\boldsymbol{\alpha}}$ for each $W \in W_{\text{full}}$ to obtain $\boldsymbol{\theta}'$
return $\boldsymbol{\theta}'$

Algorithm 1: Dictionary-Based Selective Knowledge Updating

We also extend MUSE from the NEWS domain to BOOKS. These experiments examine whether the balance between target forgetting and retained-knowledge preservation remains consistent under different unlearning scales and domain distributions.

Generalization across Forgetting Ratios. Fig. 7 shows that *KEY* maintains a favorable forgetting–utility trade-off across different forget ratios and base models. Although its FQ is slightly lower than PALU in several settings, it remains competitive with or superior to most other baselines. More importantly, *KEY* consistently achieves strong MU, indicating better preservation of retained knowledge as the forgetting ratio changes. These results suggest that the proposed method generalizes robustly across different unlearning scales without sacrificing model utility.

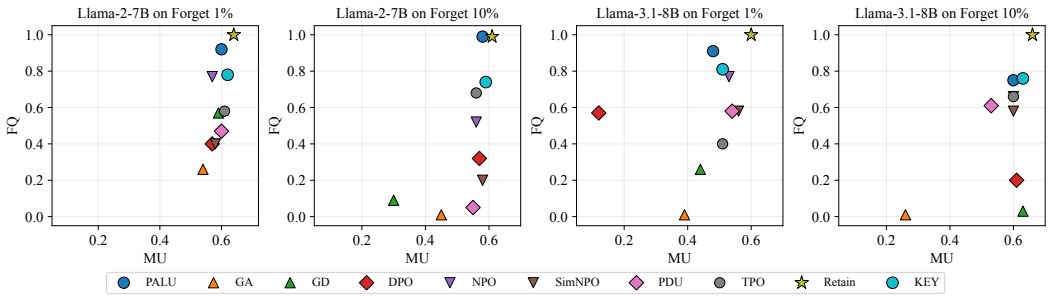


Figure 7: Generalization across different forgetting ratios on TOFU. Each point represents the trade-off between Forget Quality (FQ) and Model Utility (MU) under different base models and forgetting ratios.

Generalization across Domains. To further examine domain generalization, we evaluate *KEY* on the MUSE–Books subset, while MUSE–News is reported in the main experiments. As shown in Table 8, *KEY* achieves a KM_r of 0.6511, while maintaining KM_f and VM_f at 0.3015 and 0.2257, respectively. Compared with PALU, *KEY* preserves more retained knowledge (0.6511 vs. 0.6163) while maintaining competitive forgetting performance and achieving lower privacy leakage (-50.2433 vs. -55.7544). Together with the MUSE–News results, this indicates that the proposed selective-unlearning behavior generalizes across different data domains.

E.3 Hyperparameter Analysis

To examine the sensitivity of *KEY* to its key hyperparameters, we conduct two groups of experiments: ❶ varying the numbers of retained input and output singular vectors, and ❷ varying the trade-off between retention and sparsity regularization through λ_{ret}, with $\lambda_{\mathrm{sel}} = 1 - \lambda_{\mathrm{ret}}$. These experiments assess how the available update directions and the balance between knowledge preservation and sparse atom selection affect target updating and interference with non-target knowledge. We use 5% TOFU with Llama-2-7B for knowledge unlearning and CounterFact with Llama-3-8B for knowledge editing. All other experimental settings follow those used in the main experiments.

Table 8: Performance on MUSE–Books subset.

Method	$\mathbf{KM}_r \uparrow$	$\mathbf{KM}_f \downarrow$	$\mathbf{VM}_f \downarrow$	**PL**$\rightarrow 0$
NPO	0.6424	0.4414	0.6011	-55.7692
SimNPO	0.5969	0.3009	0.2364	-51.7018
PDU	0.0709	0.0529	0.1193	-76.1834
TPO	0.6016	0.3607	0.4234	-57.3639
PALU	0.6163	0.2896	0.2052	-55.7544
***KEY* (Ours)**	0.6511	0.3015	0.2257	-50.2433

Numbers of retained singular vectors. To examine how the available update directions affect target updating and non-target knowledge preservation, we vary the numbers of retained input and output singular vectors, r_{in} and r_{out}, over $\{8, 16, 24, 32, 48, 64\}$. Three settings are considered: fixing $r_{\mathrm{in}} = 48$ while varying r_{out}, fixing $r_{\mathrm{out}} = 48$ while varying r_{in}, and jointly increasing $r_{\mathrm{in}} = r_{\mathrm{out}}$. The separate sweeps examine the contribution of each subspace dimension, while the joint sweep examines their combined effect. Each configuration retains $\min\{1024, r_{\mathrm{in}} r_{\mathrm{out}}\}$ atoms per weight matrix, so lower-rank configurations may use fewer coefficients. Fig. 8 shows that increasing the retained dimensions improves target updating, with gains saturating around 48. Under the joint sweep, increasing both dimensions from 8 to 48 raises editing ES and PS from 42.8% and 41.5% to 99.9% and 98.1%, respectively, while NS also improves from 75.4% to 83.3%. For unlearning, FQ increases from 0.089 to 0.691 and EM decreases from 0.898 to 0.593. Beyond 48, editing ES and PS change by at most 0.3 percentage points across all three settings. For joint-sweep unlearning, increasing both dimensions to 64 yields only a small FQ gain to 0.700, while MU decreases from 0.655 to 0.636.

The saturation is consistent with the rank ablations in LoRA (Hu et al., 2022), where increasing rank does not consistently improve performance. In our method, retaining more singular vectors expands the candidate atom set formed by $u_p v_q^\top$, providing coefficient optimization with more directions to

combine. The simultaneous gains in editing success and neighborhood preservation suggest that this additional flexibility can help construct updates that modify target knowledge while limiting interference with non-target knowledge. The limited gains beyond 48 suggest that further expansion offers little additional benefit under the evaluated settings. These results support $r_{\text{in}} = r_{\text{out}} = 48$ as a practical balance between target updating, knowledge preservation, and candidate dictionary size.

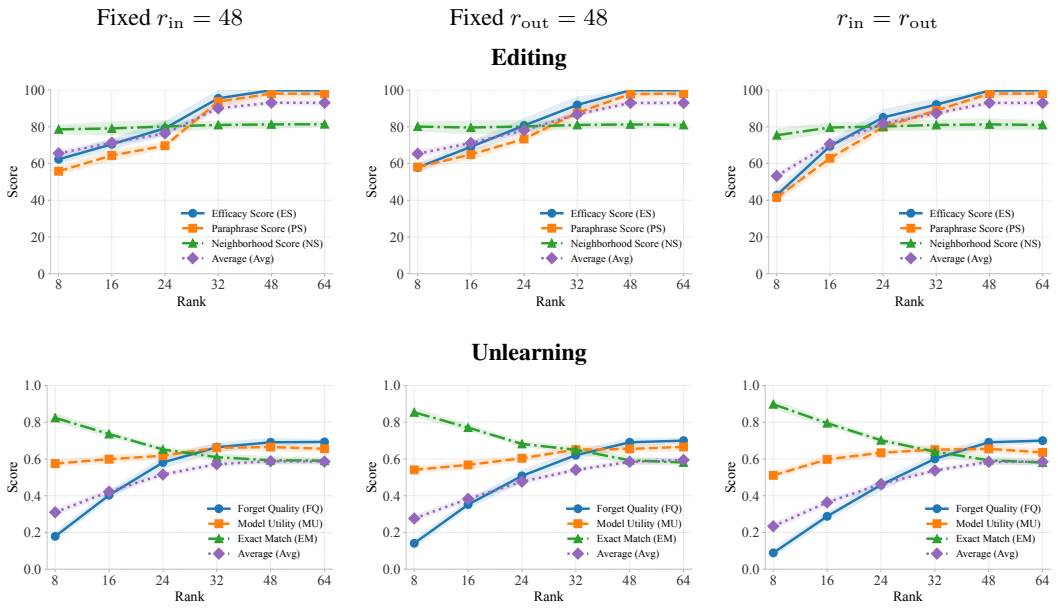


Figure 8: Effect of retained input and output dimensions on knowledge editing and unlearning.

Balance between retention and sparsity. To examine the trade-off between knowledge preservation and sparse atom selection, we vary λ_{ret} from 0.1 to 1.0 with an interval of 0.1, while setting $\lambda_{\text{sel}} = 1 - \lambda_{\text{ret}}$. This sweep controls the relative contribution of the retention objective and ℓ_1 regularization, allowing us to assess their joint effect on target updating and non-target knowledge preservation.

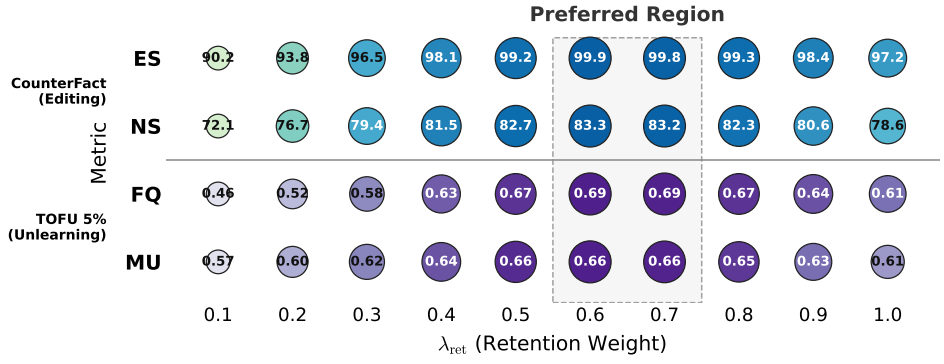


Figure 9: Effect of the retention–sparsity trade-off on knowledge editing and unlearning. Bubble size and color indicate relative performance within each metric.

As shown in Fig. 9, performance improves consistently as λ_{ret} increases from 0.1 to 0.6, after which it remains nearly unchanged around 0.6–0.7 before gradually decreasing. On CounterFact, $\lambda_{\text{ret}} = 0.6$ achieves the highest ES and NS of 99.9 and 83.3, respectively, while $\lambda_{\text{ret}} = 0.7$ yields nearly identical performance (99.8 ES and 83.2 NS). A similar pattern is observed on TOFU, where both settings achieve an FQ of 0.69 and an MU of 0.66. In contrast, emphasizing sparsity too strongly ($\lambda_{\text{ret}} = 0.1$) substantially degrades all metrics, whereas removing sparsity regularization

($\lambda_{\text{ret}} = 1.0$) also reduces both target updating and knowledge preservation. These results indicate that retention and sparsity provide complementary constraints, with $\lambda_{\text{ret}} \in [0.6, 0.7]$ offering a stable balance across both tasks.

E.4 Knowledge Distinguishability Through Atom Usage

Knowledge Items and Question Formulations. This analysis follows the knowledge-level setting used in the motivation study in Appendix B. We use the same 20 knowledge items and the same prompting procedure to construct two semantically equivalent question formulations for each item while keeping the target answer unchanged. Together with the original question, this yields three formulations per knowledge item. The dictionary is constructed from TOFU-full using Llama-2-7B, with all remaining settings following the main experiments. The resulting filtered dictionary $\mathcal{D}^\star$ is fixed throughout this analysis, including its atom set and ordering. Each of the 60 question formulations is then independently addressed over the same $\mathcal{D}^\star$ without re-filtering the dictionary, such that every coordinate of $\boldsymbol{\alpha}$ consistently corresponds to the same atom. This results in $20 \times 3 = 60$ directly comparable coefficient vectors $\boldsymbol{\alpha}$.

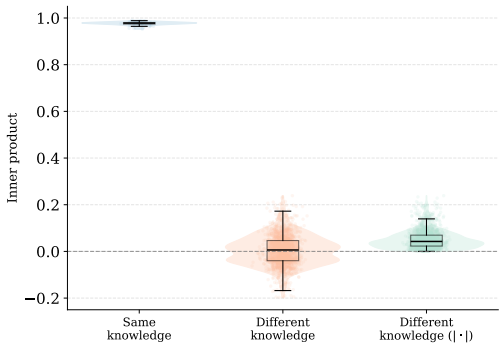
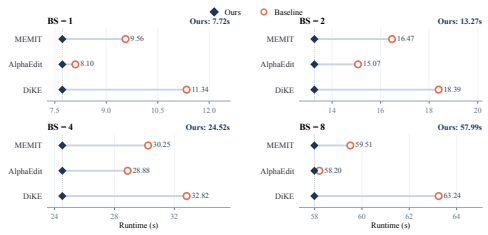


Figure 10: Distribution of pairwise cosine similarities between normalized coefficient vectors.

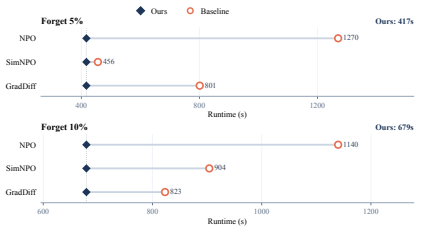
Additional Quantitative Results. To complement the qualitative results in Sec. 5.3, we quantify the inner products between normalized coefficient vectors. As summarized in Fig. 10, the 60 within-knowledge pairs have an average inner product of 0.976 ± 0.008, with all values ranging from 0.952 to 0.991. In contrast, the 1,710 cross-knowledge pairs have a signed mean inner product of only 0.005 ± 0.064. Their mean absolute inner product is 0.051 ± 0.039, with a median of 0.043 and 95% of the values below 0.133. These results show that coefficient vectors for the same knowledge are highly aligned, whereas those for different knowledge items are close to orthogonal.

E.5 Computational Complexity and Efficiency

The computational cost is examined in terms of dictionary construction and subsequent knowledge updates. For n gradient factors, maximum matrix dimension d, and k retained atoms, dense-SVD dictionary construction requires $O(C_{\text{D}} + nd\min\{n, d\})$, while factorized coefficient optimization requires $O(C_{\text{U}} + kd)$ per iteration, where C_{D} and C_{U} denote the respective model-computation costs.



(a) Editing efficiency across different batch sizes.



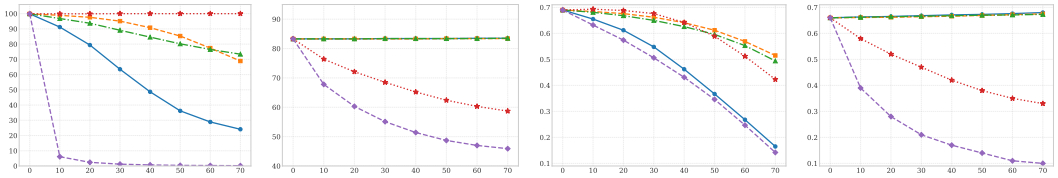
(b) Unlearning efficiency across different forget ratios.

Figure 11: Runtime comparison for knowledge editing and unlearning.

Fig. 11 further compares the empirical runtime for editing and unlearning. For editing, *KEY* achieves the lowest runtime across all batch sizes, reducing runtime by up to 15.1% over AlphaEdit. For unlearning, *KEY* requires 417 s and 679 s on the 5% and 10% TOFU settings, respectively, outperforming the fastest reported baselines by 8.6% and 17.5%. These results show that, once the dictionary is constructed, optimizing only the coefficient vector enables efficient knowledge updates across both tasks.

E.6 Intervention Analysis of α

To examine the functional role of the learned sparse coefficients, five interventions on $\boldsymbol{\alpha}$ are considered: (i) *Top-Magnitude Masking (**TM**)*, setting the top 10%–70% coefficients ranked by absolute magnitude to zero; (ii) *Magnitude Amplification (**MA**)*, multiplying the corresponding top-ranked coefficients by 2; (iii) *Magnitude Attenuation (**MT**)*, scaling the corresponding coefficients by 0.5; (iv) *Sign Flipping (**SF**)*, reversing the signs of the corresponding coefficients while preserving their magnitudes; and (v) *Random Masking (**RM**)*, randomly selecting the same proportion of active coefficients and setting them to zero. These interventions examine whether coefficient importance is concentrated in high-magnitude entries and whether the learned magnitudes and signs are functionally important for knowledge updating.



(a) CounterFact – ES (b) CounterFact – NS (c) TOFU – FQ (d) TOFU – MU

TM RM MT MA SF

Figure 12: Intervention analysis of the learned coefficients $\boldsymbol{\alpha}$ on CounterFact and TOFU. TM, MA, MT, and SF are applied to the top 10%–70% coefficients ranked by $|\alpha_m|$, while RM randomly masks the same number of coefficients.

As shown in Fig. 12, the interventions reveal clear differences between high-magnitude and randomly selected coefficients. On CounterFact, masking the top-magnitude coefficients causes ES to decrease rapidly from 99.9 to 24.1 as the intervention ratio increases to 70%, while NS remains nearly unchanged (83.3–83.5). In contrast, random masking leads to a substantially milder ES decrease, reaching 68.9 at 70%. This gap indicates that the target-editing effect is concentrated in high-magnitude coefficients rather than being uniformly distributed across the coefficient vector.

The coefficient magnitudes are also functionally meaningful. Magnitude attenuation produces a gradual ES decrease to 73.4 while largely preserving NS, whereas magnitude amplification maintains nearly perfect ES but substantially decreases NS from 83.3 to 58.7. This suggests that the learned magnitudes are calibrated to balance update efficacy and locality. Sign flipping produces the strongest degradation, reducing ES to 0.2 and NS to 45.9, demonstrating that coefficient signs encode essential directional information for realizing the intended update.

A consistent pattern is observed on TOFU. Top-magnitude masking decreases FQ from 0.690 to 0.165, substantially faster than random masking, which retains an FQ of 0.515 at 70%, while MU remains largely preserved. Magnitude attenuation shows a similar but milder degradation, reaching 0.494 FQ and 0.673 MU. In contrast, magnitude amplification increasingly harms both FQ and MU, reducing them to 0.423 and 0.330, respectively. Sign flipping is the most destructive intervention, reducing FQ to 0.142 and MU to 0.100. Overall, these results show that both high-magnitude coefficient locations and their signed magnitudes are critical for realizing effective and selective knowledge updates.

E.7 Knowledge Addressing in the Original Weight Space

We further investigate how knowledge addressing manifests in the original model weights through two complementary analyses: ❶ *single-knowledge localization*, which maps addressed dictionary atoms back to their layers and parameter modules; and ❷ *pairwise weight-space alignment*, which examines how different knowledge units interact after this mapping. These analyses are intended as interpretability diagnostics rather than performance evaluations.

For a compact diagnostic analysis, we sample 20 knowledge instances from TOFU and use Llama-2-7B with $N_{\text{atom}} = 256$ per weight matrix, rather than 1024 as in the main experiments, to facilitate visualization while keeping the dictionary construction and layer–module mapping procedure un-

changed. For pairwise analysis, a *related* pair contains two distinct facts about the same fictitious author, whereas an *unrelated* pair contains knowledge from different authors with neither a shared subject nor a shared target fact; question forms are matched where possible. For visualization, we operationally define the active atoms of knowledge item i as the top 10% ranked by $|\alpha_{i,m}|$. This definition is used only for analysis and does not threshold or truncate the coefficients used to construct $\Delta W_{\boldsymbol{\alpha}}$.

Single-Knowledge Localization. Fig. 13 provides two complementary views of how a target knowledge unit is localized in the original model weights. The upper panel shows its active atoms in the global dictionary space, as defined above, together with their signed coefficients indicating the direction and magnitude of their contributions. Rather than being uniformly distributed along the dictionary index, these high-magnitude atoms form several distinct regions. The lower panel maps the same atoms back to their original parameter locations, revealing a structured layer–module distribution. The salient activity is concentrated in a subset of layers and modules, with pronounced presence in several MLP and value-projection weights. These observations reveal a structured and traceable parameter-space footprint for an individual knowledge unit.

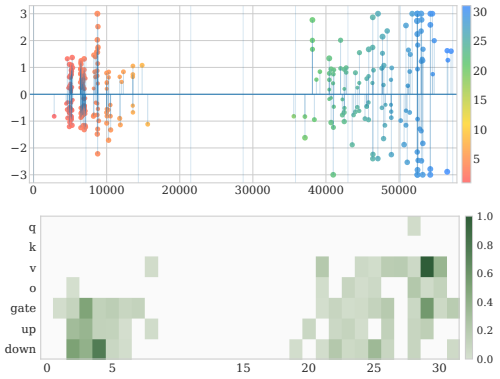


Figure 13: Localization of knowledge addressing in the original model weights. The upper panel shows the top-magnitude active atoms in the global dictionary, and the lower panel shows their layer–module distribution after mapping them back to the original weight matrices.

Pairwise Weight-Space Entanglement. We next examine whether such localized footprints remain distinguishable across different knowledge units. Fig. 14 compares representative related and unrelated pairs after mapping their active atoms back to the original weight space. Both pairs exhibit overlap across multiple layer–module locations, including the unrelated pair, indicating that different knowledge units can rely on shared parameter regions even when they are semantically unrelated. This indicates that different knowledge units can remain entangled at the level of the original model weights, and that layer–module localization alone is insufficient to separate individual knowledge units.

However, the fine-grained alignment within these shared regions differs substantially. Related knowledge exhibits strong and coherent weight-space alignment, whereas the unrelated pair shows a much weaker and more fragmented pattern. This distinction suggests that the proposed dictionary does not eliminate parameter sharing itself; instead, it alleviates knowledge entanglement by decomposing shared parameter regions into finer-grained, distinguishable atom directions. Thus, knowledge units may remain entangled in the original weight space while becoming substantially more separable through dictionary-based addressing.

E.8 Discussion: Dual-Use Risks and Opportunities for Knowledge Addressing

Knowledge updating is inherently dual-use. While it is typically designed to correct false or outdated knowledge, the same capability can be repurposed to inject harmful information. Chen et al. (2026a) formulate this threat as an *Editing Attack* and show that existing editing methods can inject misinformation or biased associations while largely preserving general model capabilities. This raises a natural question for *KEY*: whether the fine-grained locality introduced by knowledge addressing also persists when the desired update itself is malicious.

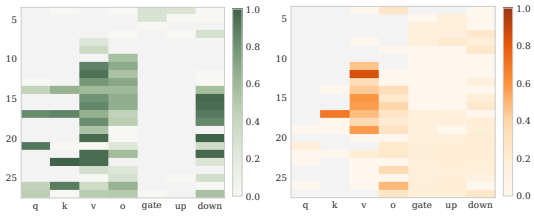


Figure 14: Pairwise alignment in the weight space. The left panel shows a related knowledge pair, while the right panel shows an unrelated pair.

Preliminary malicious-editing analysis. To examine this question, we consider the commonsense misinformation injection setting of Chen et al. (2026a). A false fact is treated as the desired editing target, and attack effectiveness is evaluated by *Efficacy*, *Generalization*, and *Portability*, measuring whether the injected misinformation can be recalled from the original query, paraphrased queries, and related reasoning contexts, respectively. We additionally evaluate general knowledge and reasoning capabilities using BoolQ, NaturalQuestions, GSM8K, and NLI. The latter metrics provide a coarse measure of the behavioral footprint left by the malicious edit: an attack that successfully injects the target misinformation while preserving these capabilities is less visible through aggregate model degradation. For reference, Table 9 includes the reported Llama3-8B results of ROME and FT from Chen et al. (2026a), together with *KEY* under the same evaluation settings.

Table 9: Preliminary evaluation under commonsense misinformation injection on Llama3-8B. Efficacy (Eff.), Generalization (Gen.), and Portability (Port.) measure malicious editing effectiveness, while BoolQ, NaturalQuestions (NQ), GSM8K, and NLI evaluate general capabilities after editing. Results for ROME and FT are taken from Chen et al. (2026a).

	Misinformation Injection			**General Capability**			
Method	**Eff.** ↑	**Gen.** ↑	**Port.** ↑	**BoolQ** ↑	**NQ** ↑	**GSM8K** ↑	**NLI** ↑
No Editing	–	–	–	62.40	35.81	99.60	85.00
ROME	90.0	70.0	72.0	61.12	35.24	99.56	84.96
FT	88.0	72.0	86.0	62.00	35.20	99.52	85.16
KEY	94.0	77.0	85.0	62.78	34.82	99.56	85.04

According to Table 9, *KEY* achieves the highest Efficacy and Generalization scores, reaching 94.0% and 77.0%, respectively, while attaining a Portability score of 85.0%, close to the 86.0% of FT and substantially above ROME. Meanwhile, its BoolQ, GSM8K, and NLI performance remains close to the unedited model, although NaturalQuestions decreases from 35.81 to 34.82. These results suggest that fine-grained knowledge addressing can support highly effective misinformation injection while inducing only limited degradation on several broad capability metrics, although the behavioral footprint is not uniformly negligible.

A dual-use implication. *KEY* is not designed as an attack method; this experiment instead probes whether its locality property depends on the intent of the target update. The results suggest that it does not: the same mechanism that enables precise benign editing can also support targeted malicious updates with relatively limited collateral changes. This highlights a dual-use risk of fine-grained knowledge addressing, since harmful knowledge may be injected without necessarily producing large degradation on broad capability benchmarks. Such results motivate complementary detection or auditing mechanisms that operate directly on the edited knowledge representation rather than relying solely on aggregate model performance.

Addressing structure as a security interface. Beyond the parameter update itself, *KEY* exposes a structured representation through the sparse coefficients $\boldsymbol{\alpha}$ and the mapping from active atoms to model layers and parameter modules. Such information could provide an interface for future auditing mechanisms. For an active-atom set $\mathcal{A}$, a conceptual audit representation may be written as

$$\begin{aligned} \boldsymbol{r}_{\text{audit}} &= \Big[\|\boldsymbol{\alpha}\|_0,\ \|\boldsymbol{\alpha}\|_1,\ \pi_{\text{layer}}(\mathcal{A}),\ \pi_{\text{module}}(\mathcal{A}),\ \text{ovlp}(\mathcal{A}, \mathcal{A}_{\text{prot}})\Big], \\ \mathcal{D}(\boldsymbol{r}_{\text{audit}}) &\in \{\text{Accept},\ \text{Reject},\ \text{Filter}\}. \end{aligned} \tag{48}$$

where $\mathcal{A}_{\text{prot}}$ denotes atoms associated with protected knowledge, and π_{layer} and π_{module} summarize the spatial distribution of active atoms. Neither $\mathcal{D}$ nor the protected-atom set is instantiated in the current method; Eq. (48) only illustrates how explicit knowledge addressing could support future detection and defense mechanisms.