HoReN: Normalized Hopfield Retrieval for Large-Scale Sequential Model Editing

Anonymous authors
Paper under double-blind review

Abstract

Large language models encode vast factual knowledge that can become outdated or incorrect after deployment, yet retraining is prohibitively costly. This motivates lifelong model editing, which updates targeted behavior while preserving the rest of the model. Existing editors, both parameter-modifying and parameter-preserving, degrade severely as edits accumulate and struggle to generalize across paraphrases. We propose **HoReN**, a codebook-based parameter-preserving editor that wraps a single MLP layer with a discrete key–value memory. HoReN treats each codebook entry as both a knowledge key and a Hopfield stored pattern, retrieves edits by angular similarity on the unit hypersphere, and refines queries through damped Hopfield dynamics so paraphrases move toward the correct memory basin while unrelated inputs remain stable. HoReN achieves strong editing performance with consistent gains across diverse benchmarks spanning standard ZsRE, structured WikiBigEdit, and unstructured UnKE evaluations. Moreover, HoReN scales to 50K sequential edits on ZsRE with stable overall performance above 0.93, while the evaluated baselines collapse or degrade severely before reaching 10K. Our code is available at `https://anonymous.4open.science/r/elephant-4B7F`.

1 Introduction

Large language models (LLMs) (Dubey et al., 2024; Yang et al., 2024; Team, 2025; Agarwal & Others, 2025) are routinely deployed against a world whose facts drift, whose deployments surface hallucinations, and whose operators need to inject corrections that were absent at pre-training time. Re-training end-to-end for every update is prohibitively expensive and risks catastrophic forgetting (Kirkpatrick et al., 2017; Ran et al., 2024; Xue et al., 2025; Luo et al., 2026b), motivating *model editing*: targeted procedures that revise an LLM's behavior on a specified prompt while leaving its other behaviors intact, with edits arriving as a continual stream (Meng et al., 2022a; Hartvigsen et al., 2023; Wang et al., 2024; 2025; Luo et al., 2026a). Existing editors fall into two paradigms: **parameter-modifying** methods alter base weights, including the locate-and-edit line (ROME (Meng et al., 2022a), MEMIT (Meng et al., 2022b), AlphaEdit (Fang et al., 2024)) and one-shot direct shifts (UltraEdit (Gu et al., 2025)); **parameter-preserving** methods leave the base weights frozen and route edits through external memory at a chosen layer, as in GRACE's discrete codebook (Hartvigsen et al., 2023) and WISE's gated side-memory (Wang et al., 2024).

Parameter-modifying editors have shown notable success on small batches of edits, but directly perturbing weights trained on trillions of tokens disrupts preserved knowledge as edits accumulate, and constraint-based remedies such as AlphaEdit's null-space projection (Fang et al., 2024) only partially mitigate this drift in a sequential regime (Figure 1). Parameter-preserving editors take a different route. GRACE (Hartvigsen et al., 2023) writes new mappings into a chosen layer's latent space as a discrete key–value codebook of adaptors and routes by a hard-radius nearest-neighbor lookup that defers to a stored value whenever an incoming activation falls within the radius of a stored key. WISE (Wang et al., 2024) maintains a dual parametric memory, a frozen main memory of pretrained weights and a trainable side memory holding edits with a router selecting between them, and uses a knowledge-sharding mechanism that places distinct edit subsets in disjoint subspaces before merging them into the side memory. Both designs protect existing knowledge by construction, yet their generalization and overall performance degrade as the edit stream scales (Figure 1). For these evaluated parameter-preserving methods, the observed failure is a routing problem: the lookup or router cannot reliably map a paraphrased query to the correct stored edit.

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

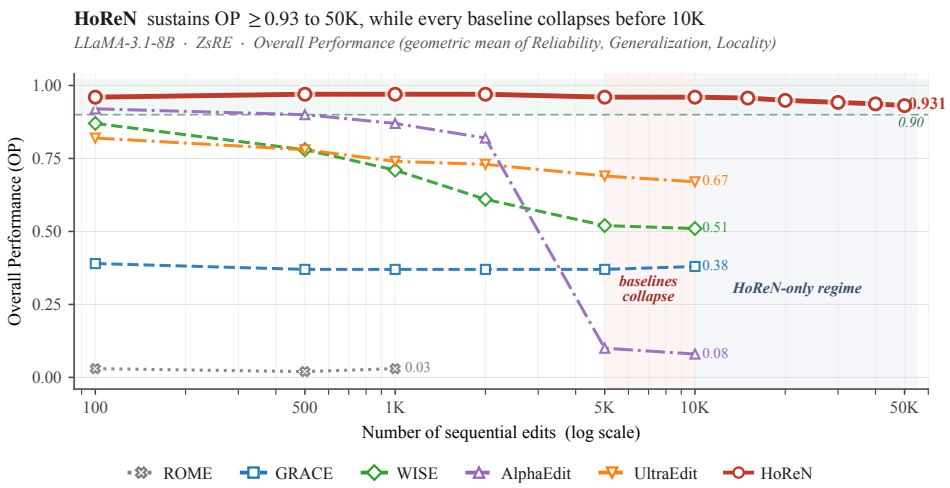


Figure 1: Overall Performance (OP, geometric mean of Reliability, Generalization, and Locality) scaling from 100 to 50K sequential edits on ZsRE (LLaMA-3.1-8B). HoReN sustains OP ≥ 0.93 through 50K edits, while every baseline evaluated in this scaling experiment collapses or plateaus before 10K: ROME fails immediately (0.03), GRACE flatlines at 0.37–0.38, WISE drifts down to 0.51, AlphaEdit cliffs from 0.82 at 2K to 0.08 at 10K, and UltraEdit drifts down to 0.67. Beyond 10K, among these evaluated methods, only HoReN remains within the 0.90 stability band. Per-metric breakdowns and the per-checkpoint table are in Appendix F.2.

To address these routing failures in parameter-preserving editors, we propose **HoReN** (**Ho**pfield **Re**trieval with **N**ormalized representations), which combines three components. (i) The *codebook architecture* of GRACE (Hartvigsen et al., 2023): a frozen base model with one MLP layer wrapped by a discrete key–value memory. (ii) *Normalization*: keys and queries are projected onto the unit hypersphere so retrieval depends on angular similarity rather than activation magnitude. (iii) *Hopfield dynamics* (Ramsauer et al., 2020; Krotov, 2023): the codebook is read as a field of energy basins, and one damped attractor step before matching lets a paraphrase relax into the basin of the correct memory while leaving unrelated queries essentially undisturbed. The matched edit is realized by a lightweight value adaptor at the same layer l, so the entire intervention is concentrated on the routing decision.

The two routing components play complementary roles: normalization protects locality and enables generalization by keeping unrelated queries separated on the hypersphere, while one damped Hopfield step closes the generalization gap by pulling rephrased queries into the basin of the correct edit. Neither operation is individually new; our contribution is the diagnosis that routing selectivity is the first bottleneck in this parameter-preserving architecture and the demonstration that these two simple changes substantially extend its usable editing horizon. The result is a sequential editor that scales without catastrophic degradation: on a controlled ZsRE stress test on LLaMA-3.1-8B, HoReN keeps reliability, generalization, and locality all above 89% out to 50K accumulated edits (Figure 1), where every evaluated scaling baseline has cliffed (AlphaEdit, ROME), drifted (UltraEdit, WISE), or flat-lined on generalization (GRACE) well before 10K. The same advantage carries over to structured WikiBigEdit (Thede et al., 2025) and unstructured UnKE (Deng et al., 2024). Related works are shown in Appendix B.

Our main contributions are:

- We characterize complementary failure modes among the evaluated lifelong editors: parameter-modifying methods progressively disrupt preserved knowledge, while GRACE-style codebook retrieval loses generalization because magnitude-sensitive nearest-neighbor matching cannot bridge the representation gap between an edit prompt and its rephrasings.
- We propose HoReN, a parameter-preserving editor that unifies the ROME key–value memory view with the GRACE codebook architecture under a normalized Hopfield retrieval rule, and provide theoretical analysis of the standard, unnormalized Hopfield operator.

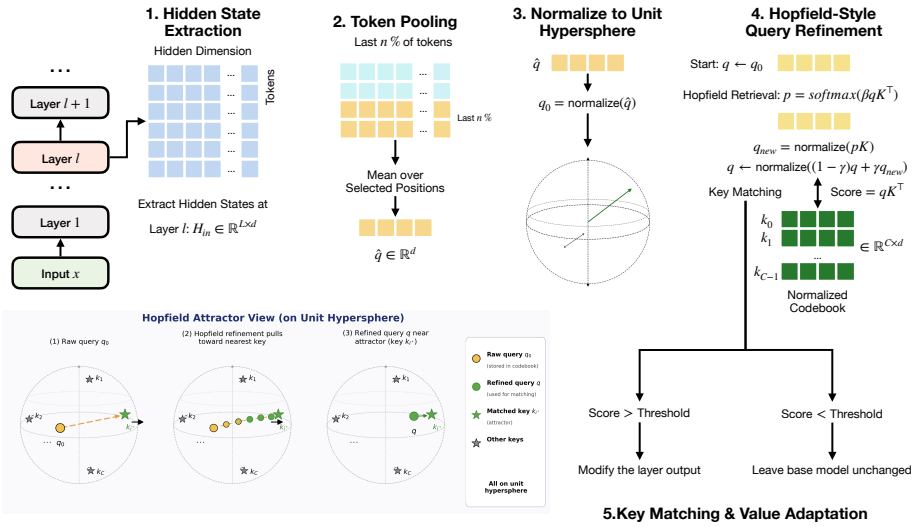


Figure 2: Overall architecture of HoReN combining normalized representations with Hopfield-style retrieval. The pipeline: **(1)** extract hidden states from layer l; **(2)** pool token representations to construct the query; **(3)** normalize query to the unit hypersphere; **(4)** apply one-step Hopfield-style refinement over the normalized codebook; **(5)** perform key matching and value adaptation at layer l. This design achieves complementary benefits: normalization preserves locality through angular similarity matching, while Hopfield-style refinement improves paraphrase routing by moving the query toward edit-key attractors.

- We demonstrate large-scale stability of HoReN on a 50K-edit ZsRE stress test on LLaMA-3.1-8B, where the evaluated scaling baselines cliff, drift, or flat-line well before 10K edits, and confirm consistent gains on structured WikiBigEdit and unstructured UnKE across seven LLMs spanning five model families.

2 METHOD

HoReN inherits the architectural skeleton of GRACE (Hartvigsen et al., 2023): the base model f_{θ_0} is frozen and a single MLP layer l is wrapped with a discrete codebook $\mathcal{C} = \{(k_i, v_i, y_i)\}_{i=1}^{C}$ (Figure 2). At inference, an incoming query is routed through the codebook; a successful match modifies the layer's output by adding the matched value to the output, while an unmatched query passes through the base model unchanged. Editing reduces to two questions: *what is stored* and *how is a query routed to the right entry*. HoReN's answers to both flow from treating each stored key simultaneously as a ROME-style knowledge-memory index (Meng et al., 2022a) and as an attractor basin in a modern Hopfield network (Ramsauer et al., 2020): keys and queries are projected onto the unit hypersphere so that retrieval is purely angular (Section 2.2), and routing proceeds by a single controlled relaxation under the codebook's attractor field before any matching decision (Section 2.3).

2.1 PROBLEM FORMULATION

Setup. Following the lifelong editing setup of (Wang et al., 2024), let $f_\theta : \mathcal{X} \to \mathcal{Y}$ be an LLM with pre-edit parameters θ_0. Each edit index t carries an original prompt x_t, a paraphrase $\tilde{x}_t$, a shared target y_t^*, and an unrelated locality query x_t^{loc}. At step t the editor produces

$$f_{\theta_t} = \text{ModelEditing}(f_{\theta_{t-1}}, x_t, y_t^*), \quad \text{so that} \quad f_{\theta_t}(x) = \begin{cases} y_t^*, & x \in \{x_t, \tilde{x}_t\}, \\ f_{\theta_0}(x), & x = x_t^{\text{loc}}. \end{cases} \tag{1}$$

After N sequential edits, the final model must satisfy this condition for *all* $t \in \{1, \ldots, N\}$, measured by reliability, generalization, and locality (definitions and dataset-specific instantiations in Section 3.1). The difficulty of the problem grows with N: each new edit must coexist with every prior one, so the

editor needs a memory that scales gracefully and a routing rule that remains discriminative even as nearby memories accumulate.

2.2 The Codebook: Normalized Keys on the Unit Hypersphere

Each codebook entry (k_i, v_i, y_i) at editing layer l binds a key $k_i \in \mathbb{R}^d$ to a value vector $v_i \in \mathbb{R}^d$ and the target label y_i that originally created the entry. Keys are constructed from the layer-l hidden states of the prompt that triggered the edit, and queries are constructed in the same way at routing time, so that storage and retrieval live in a common representation space.

Concretely, given an input x we extract $H_{\text{in}} \in \mathbb{R}^{L\times d}$ from the frozen base at layer l and pool the last $n\%$ of token positions into a raw query $\hat{q} = \text{select}(H_{\text{in}}) \in \mathbb{R}^d$. Pooling a suffix of tokens, rather than a single position, summarises the semantic content of the prompt while remaining robust to surface variation such as padding and minor rephrasing. We then project to the unit hypersphere,

$$q_0 = \text{normalize}(\hat{q}) \in \mathbb{R}^{1\times d}, \qquad K \in \mathbb{R}^{C\times d}, \;\; \|k_i\|_2 = 1,$$

where $C = |\mathcal{C}| \le N$ is the current codebook size. The motivation is the geometric fact that, for post-activation MLP tensors, the direction of the vector is what binds to a particular concept while the magnitude largely reflects prompt-dependent activation strength; normalising removes this nuisance variation and exposes the angular structure that the keys are actually selecting on.

Once a query has been routed (Section 2.3), the matching decision itself is a single inner product: the editor scores keys as $\mathbf{a} = qK^\top$, takes $i^\star = \arg\max_i \mathbf{a}_i$, and accepts the match when $\mathbf{a}_{i^\star} > c$. Because keys and queries lie on the unit sphere, c is a cosine threshold and admits a precise geometric reading: it is the angular boundary of the attractor basin around $k_{i^\star}$. This replaces GRACE's deferral radius, a Euclidean tolerance whose meaning shifts with activation magnitude, with a single, scale-free notion of what it means for a query to fall "inside" a stored memory.

2.3 Querying via Hopfield Attractor Dynamics

Treating each stored key as an attractor suggests that routing should not be a one-shot nearest-neighbour lookup against the raw query q_0. A paraphrase $\tilde{x}_t$ produces an activation that is angularly close to, but not identical to, the activation of the original prompt x_t; under a hard argmax this small angular gap can be enough to send the query to the wrong key once the codebook is dense. The right object to compare against the codebook is therefore not q_0 itself but the result of letting q_0 relax briefly under the codebook's attractor field, so that paraphrases drift toward the basin of the correct memory while unrelated queries are left essentially untouched.

We make this precise using the modern Hopfield framework (Ramsauer et al., 2020). With energy

$$E(q, K) \;=\; \tfrac{1}{2}\|q\|_2^2 - \text{lse}_\beta\big(qK^\top\big), \qquad \text{lse}_\beta(\mathbf{z}) = \tfrac{1}{\beta}\log\textstyle\sum_i e^{\beta z_i},$$

the standard update $T(q) = \text{softmax}(\beta\, qK^\top)\, K$ descends E and generates a sequence that converges to a codebook-induced fixed point in the convex hull of the stored patterns, with $\beta > 0$ controlling retrieval sharpness. HoReN deploys a *normalized, damped* variant of this dynamic. Starting from $q \leftarrow q_0$, for $m = 1, \ldots, M$ we compute

$$p = \text{softmax}(\beta\, qK^\top) \in \mathbb{R}^{1\times C}, q_{\text{new}} = \text{normalize}(pK) \in \mathbb{R}^{1\times d}, \tag{2}$$

and update

$$q \;\leftarrow\; \text{normalize}\big((1-\gamma)\, q + \gamma\, q_{\text{new}}\big), \tag{3}$$

terminating early once $\|q_{\text{new}} - q\|_2 \le \epsilon$. Two deviations from the standard update matter. The intermediate re-normalisation $q_{\text{new}} = \text{normalize}(pK)$ keeps the iterate on the same hypersphere as the keys, so the dynamics never leave the geometry on which the codebook is defined. The damping factor $\gamma \in (0, 1]$ moves the query only fractionally toward q_{new} at each step, so unrelated queries, whose attention weights p are diffuse, are perturbed only slightly, while paraphrases of an existing edit, whose p concentrates on a single key, are pulled into the corresponding basin.

The single damped step is an empirically selected stopping rule motivated by the over-attraction behavior of standard Hopfield retrieval. Iteration maps every query, including unrelated or genuinely novel ones, into the convex hull of the stored patterns and ultimately toward a codebook-induced

fixed point; for large β, that fixed point can be dominated by a nearby stored key, creating a false activation and harming locality. We characterise this behaviour for the standard operator T with two complementary results, stated here and proved in Appendix D.

Theorem 2.1 (Asymptotic convergence of standard Hopfield retrieval)**.** *Let $K \in \mathbb{R}^{C\times d}$ be a normalized codebook ($\|k_i\|_2 = 1$), let $q^{(0)} \in \mathbb{R}^{1\times d}$ satisfy $\|q^{(0)}\|_2 = 1$, and let $\beta > 0$. Define the standard Hopfield update map $T(q) := \mathrm{softmax}(\beta\, qK^\top)\, K$ and generate the infinite sequence $q^{(s+1)} = T(q^{(s)})$. With $F(q) := \frac{1}{\beta}\log\sum_{i=1}^{C}\exp(\beta\, qk_i^\top)$ and energy $E(q,K) := \frac{1}{2}\|q\|_2^2 - F(q)$, the energy is monotonically non-increasing,*

$$E(q^{(s+1)},K) - E(q^{(s)},K) \;\le\; -\tfrac{1}{2}\,\|q^{(s+1)} - q^{(s)}\|_2^2,$$

and the sequence converges to a fixed point: $q^{(s)} \to q^*$ with $q^* = T(q^*)$.

Proposition 2.2 (Finite-step descent and residual bound)**.** *Under the assumptions of Theorem 2.1, for any finite $M \ge 1$ the truncated iterates $q^{(s+1)} = T(q^{(s)})$, $s = 0,\dots,M-1$, satisfy the cumulative descent inequality*

$$\sum_{s=0}^{M-1}\|q^{(s+1)} - q^{(s)}\|_2^2 \;\le\; 2\big(E(q^{(0)},K) - E(q^{(M)},K)\big),$$

and consequently the iterate with smallest fixed-point residual obeys the explicit bound

$$\min_{0\le s<M}\|T(q^{(s)}) - q^{(s)}\|_2 \;\le\; \frac{2}{\sqrt{M}}.$$

The bound applies to the best *of the first M iterates; the last iterate $q^{(M)}$ is only guaranteed to satisfy $E(q^{(M)},K) \le E(q^{(0)},K)$ and may not itself be an approximate fixed point.*

Theorem 2.1 formalises the fixed-point attraction of standard Hopfield retrieval: after the first update, every iterate lies in the convex hull of the stored codes and the sequence converges to a codebook-induced fixed point, regardless of whether the original query corresponds to an edit. Proposition 2.2 shows that this contraction is already operative at finite step counts in an averaged sense.

2.4 Updating the Codebook

The codebook grows online as edits arrive. After routing, three cases can arise. (i) No-match: the best score $\mathbf{a}_{i^\star} \le c$ falls outside every existing basin, so the edit is genuinely new and we insert a fresh entry $(q_0, v_{\text{new}}, y_t^*)$, where v_{new} is trained by cross-entropy on the target y_t^* with the base model frozen. (ii) Match with consistent label ($y_{i^\star} = y_t^*$): the edit is a paraphrase or a reassertion, and we fine-tune the existing value adaptor $v_{i^\star}$ on the new prompt while keeping its key $k_{i^\star}$ fixed. (iii) Match with conflicting label ($y_{i^\star} \ne y_t^*$): the basin is contested and we insert a new entry to preserve the new fact, leaving the prior memory in place. At inference, a successful match adds the matched value to the layer output at the index of the last prompt token to be passed to the next layer; on a no-match the base model is left unchanged. An alternative LoRA-based adaptor is described in Appendix I.3, and the full procedure is given in Algorithm 1 (Appendix C.1).

Routing mechanism. The routing mechanism has a key structural asymmetry: stored keys are always the *pre-refinement* query q_0, while the routing comparison uses the refined q output by the Hopfield step. This asymmetry is not a tuning choice but follows directly from the attractor geometry. If newly stored keys were the output of the Hopfield refinement, each insertion would pull subsequent insertions toward existing attractors, basins would co-move with the codebook, and the angular separation that retrieval depends on would progressively collapse. Pinning storage to q_0 anchors the codebook to the model's native representational geometry, so the attractor field is fixed by what the base model actually produces; the refinement step then moves paraphrases toward the correct basin at routing time without contaminating what is stored.

3 Experiments

3.1 Experimental Setup

We briefly outline the base LLMs, baselines, datasets, and evaluation metrics. Full setup descriptions and implementation details are in Appendix E.

Table 1: Performance comparison on ZsRE benchmark at different editing scales ($N \in \{100, 500, 1000\}$) on four models.

Model	Method	$N = 100$				$N = 500$				$N = 1000$			
		Rel.	**Gen.**	**Loc.**	**OP**	**Rel.**	**Gen.**	**Loc.**	**OP**	**Rel.**	**Gen.**	**Loc.**	**OP**
LLaMA-3 8B	ROME	0.17	0.16	0.01	0.06	0.02	0.03	0.00	0.00	0.02	0.02	0.00	0.00
	GRACE	0.99	0.01	1.00	0.21	0.99	0.02	1.00	0.27	0.99	0.02	1.00	0.27
	WISE	0.54	0.50	1.00	0.65	0.56	0.54	1.00	0.67	0.55	0.54	0.99	0.66
	AlphaEdit	0.98	0.92	0.89	0.93	0.99	0.94	0.83	0.92	0.98	0.92	0.78	0.89
	UltraEdit	0.91	0.88	0.69	0.82	0.90	0.88	0.56	0.76	0.87	0.85	0.54	0.74
	HoReN	**1.00**	**0.81**	**1.00**	**0.93**	**1.00**	**0.84**	**1.00**	**0.94**	**0.99**	**0.84**	**1.00**	**0.94**
LLaMA-3.1 8B	ROME	0.04	0.05	0.01	0.03	0.02	0.02	0.02	0.02	0.03	0.02	0.03	0.03
	GRACE	1.00	0.06	1.00	0.39	1.00	0.05	1.00	0.37	0.99	0.05	1.00	0.37
	WISE	0.83	0.79	1.00	0.87	0.71	0.66	1.00	0.78	0.62	0.59	0.99	0.71
	AlphaEdit	0.98	0.90	0.88	0.92	0.98	0.93	0.80	0.90	0.97	0.92	0.74	0.87
	UltraEdit	0.91	0.89	0.69	0.82	0.89	0.87	0.61	0.78	0.88	0.86	0.54	0.74
	HoReN	**0.99**	**0.88**	**1.00**	**0.96**	**1.00**	**0.92**	**1.00**	**0.97**	**0.99**	**0.93**	**0.99**	**0.97**
Qwen-2.5 7B	ROME	0.51	0.51	0.32	0.44	0.30	0.28	0.06	0.17	0.19	0.19	0.02	0.09
	GRACE	1.00	0.02	1.00	0.27	1.00	0.02	1.00	0.27	1.00	0.02	1.00	0.27
	WISE	0.90	0.87	1.00	0.92	0.87	0.80	0.80	0.82	0.63	0.58	0.60	0.60
	AlphaEdit	0.86	0.80	0.98	0.88	0.85	0.79	0.91	0.85	0.99	0.88	0.88	0.92
	UltraEdit	0.92	0.80	0.99	0.90	0.87	0.81	0.96	0.88	0.80	0.75	0.89	0.81
	HoReN	**0.99**	**0.93**	**1.00**	**0.97**	**1.00**	**0.93**	**0.99**	**0.97**	**1.00**	**0.93**	**0.98**	**0.97**
SEA-LION 32B	ROME	0.02	0.02	0.02	0.02	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
	GRACE	0.46	0.02	1.00	0.21	0.41	0.02	1.00	0.20	0.43	0.02	1.00	0.21
	WISE	0.94	0.72	1.00	0.88	0.80	0.68	1.00	0.82	0.70	0.63	1.00	0.76
	AlphaEdit	0.96	0.76	0.99	0.90	0.97	0.77	0.96	0.89	0.97	0.77	0.95	0.89
	UltraEdit	0.76	0.70	0.98	0.80	0.75	0.72	0.95	0.80	0.72	0.69	0.93	0.77
	HoReN	**1.00**	**0.91**	**1.00**	**0.97**	**1.00**	**0.90**	**0.99**	**0.96**	**1.00**	**0.90**	**0.98**	**0.96**

Base LLMs & Baseline Methods. Our experiments are conducted on seven LLMs spanning 1.5B to 32B parameters across five families: DeepSeek-R1-Distill-Qwen-1.5B (Team, 2025), LLaMA-3-8B-Instruct and LLaMA-3.1-8B-Instruct (Dubey et al., 2024), Qwen2.5-7B-Instruct (Yang et al., 2024), DeepSeek-R1-Distill-Llama-8B (hereafter DeepSeek-R1-8B) (Team, 2025), GPT-OSS-20B (Agarwal & Others, 2025), and Qwen-SEA-LION-v4-32B-IT (Ng et al., 2025). We compare HoReN against representative methods from two paradigms: (1) *parameter-modifying* methods, including ROME (Meng et al., 2022a), AlphaEdit (Fang et al., 2024), and UltraEdit (Gu et al., 2025); and (2) *parameter-preserving* methods, including GRACE (Hartvigsen et al., 2023), WISE (Wang et al., 2024), and REPAIR (Wang et al., 2025). We additionally include the learned-retriever editor RECIPE (Chen et al., 2024) and the learned codebook-router LOKA (Zhang et al., 2025) in Appendix F.1.

For unstructured editing we additionally compare against the domain-specific UnKE baseline (Deng et al., 2024).

Datasets & Evaluation Metrics. We evaluate HoReN on three benchmarks that progressively stress its core separation guarantee: **ZsRE** (Levy et al., 2017), the standard lifelong editing benchmark with semantically well-separated edit and locality queries; **WikiBigEdit** (Thede et al., 2025), a harder regime in which locality queries share subjects and relations with edited facts; and **UnKE** (Deng et al., 2024), an unstructured benchmark with free-form edit targets. Following prior work (Hartvigsen et al., 2023; Wang et al., 2024; 2025), we report **Reliability**, **Generalization**, and **Locality**, together with their geometric mean (Overall Performance, OP). For UnKE we follow its original protocol (Deng et al., 2024) and report BLEU, ROUGE-1/2/L and BERTScore. Full dataset and metric definitions are in Appendices E.1 and E.3.

3.2 ZsRE Baseline Failure Modes and Large-Scale Stability

3.2.1 Baseline Comparison on ZsRE

Each baseline in Table 1 fails in one or more of reliability, generalization, and locality as edits accumulate.

Parameter-modifying editors. ROME collapses on all three metrics almost immediately because it overfits the rank-one update to one prompt rather than to the underlying fact, and stacking such updates

Table 2: Scaling to 10K edits on ZsRE across four models (Qwen2.5-7B, DeepSeek-R1-8B, GPT-OSS-20B, and Qwen-SEA-LION-v4-32B). Full LLaMA-3.1-8B results at large scale, including AlphaEdit and UltraEdit, are in Appendix F.2.

Model	Method	$N = 2000$				$N = 5000$				$N = 10000$			
		Rel.	**Gen.**	**Loc.**	**OP**	**Rel.**	**Gen.**	**Loc.**	**OP**	**Rel.**	**Gen.**	**Loc.**	**OP**
Qwen2.5 7B	GRACE	1.00	0.02	1.00	0.27	1.00	0.03	1.00	0.31	1.00	0.03	1.00	0.31
	WISE	0.30	0.28	0.81	0.41	0.24	0.22	0.79	0.35	0.21	0.20	0.76	0.32
	HoReN	**1.00**	**0.92**	**1.00**	**0.97**	**0.99**	**0.89**	**1.00**	**0.96**	**0.99**	**0.89**	**0.99**	**0.96**
DeepSeek 8B	GRACE	1.00	0.02	1.00	0.27	1.00	0.02	1.00	0.27	1.00	0.02	1.00	0.27
	WISE	0.46	0.44	1.00	0.59	0.36	0.33	1.00	0.49	0.31	0.30	1.00	0.45
	HoReN	**1.00**	**0.95**	**0.98**	**0.98**	**1.00**	**0.94**	**0.97**	**0.97**	**1.00**	**0.93**	**0.97**	**0.97**
GPT-OSS 20B	GRACE	0.61	0.01	1.00	0.18	0.60	0.02	1.00	0.23	0.60	0.02	1.00	0.23
	WISE	0.14	0.14	0.08	0.12	0.12	0.11	0.12	0.12	0.12	0.11	0.12	0.12
	HoReN	**0.99**	**0.69**	**1.00**	**0.88**	**0.99**	**0.67**	**0.99**	**0.87**	**0.99**	**0.66**	**0.98**	**0.86**
SEA-LION 32B	GRACE	0.43	0.02	1.00	0.20	0.43	0.02	1.00	0.20	0.43	0.02	1.00	0.20
	WISE	0.64	0.59	1.00	0.72	0.58	0.55	1.00	0.68	0.57	0.54	1.00	0.68
	HoReN	**1.00**	**0.91**	**0.98**	**0.96**	**1.00**	**0.90**	**0.97**	**0.96**	**1.00**	**0.90**	**0.97**	**0.96**

disturbs not only unrelated knowledge but also the edited entries (Rel. 0.03, Gen. 0.02, Loc. 0.03 at N=1000 on LLaMA-3.1-8B). AlphaEdit and UltraEdit retain high reliability and generalization but their locality erodes as the edit stream grows: AlphaEdit's null-space projector is estimated from a finite sample of pre-training-style activations, so it protects directions in its proxy preservation sample but cannot include directions added by later edits, and locality therefore degrades fast ($0.88 \rightarrow 0.74$ from N=100 to 1000 on LLaMA-3.1-8B); UltraEdit's running normalization stabilizes the update geometry but still disrupts preserved knowledge, resulting in the same gradual erosion in locality ($0.69 \rightarrow 0.54$ on LLaMA-3.1-8B).

Parameter-preserving editors. GRACE inverts the failure: reliability and locality stay ≥ 0.99 on every row, but generalization is stuck at 0.01–0.06, because unnormalized nearest-neighbor lookup conflates the prompt-specific gain (magnitude) with the semantic identity (direction). WISE's gated side-memory drifts as the codebook accumulates, dragging both reliability and generalization down.

By capturing edit-semantic identity through normalization and refining queries through Hopfield dynamics, HoReN maintains high reliability, generalization, and locality jointly among the methods evaluated in Table 1; Appendix F.1 reports the additional routing-oriented comparisons.

3.2.2 Scaling to 10K and 50K Edits

Two failure shapes are diagnostic in Tables 2–5 and Figure 1. On LLaMA-3.1-8B, AlphaEdit cliffs (OP $0.82 \rightarrow 0.10$ from N=2000 to 5000 as Locality falls $0.65 \rightarrow 0.03$) because a fixed null-space projector cannot include directions added by later edits, while UltraEdit slopes (OP $0.73 \rightarrow 0.67$ over N=2000–10000, Locality $0.53 \rightarrow 0.42$). On Qwen2.5-7B, DeepSeek-R1-8B, GPT-OSS-20B, and Qwen-SEA-LION-v4-32B (Table 2), GRACE's Generalization stays in $[0.01, 0.03]$ across scales (the codebook absorbs 10K entries, so the failure is geometric rather than capacity-bound) and WISE's OP decays monotonically (collapsing to ≤ 0.13 on GPT-OSS-20B), localizing its weakness to a drifting side-memory whose performance on preserving edits and generalization degrades with scale. HoReN is flat across N in reliability 0.99–1.00, generalization 0.89–0.95 excluding the GPT-OSS-20B model, and locality 0.97–1.00 across all four models, and all three metrics are above 0.89 at 50K on LLaMA-3.1-8B, because storage is per-entry and routing depends only on the directional geometry of stored keys, so adding entries does not move existing decision boundaries. Note that HoReN's generalization on GPT-OSS-20B is comparatively lower. Since GPT-OSS-20B employs the MoE structure, HoReN is applied on an attention layer, not an MLP layer with knowledge keys, resulting in the degraded generalization performance. HoReN demonstrates stable performance on reliability, generalization, and locality up to 50K edits on the ZsRE benchmark. Targeted robustness checks are reported at N=1000 in Appendix H: five streams yield OP 0.962 ± 0.003, a shuffled order yields OP 0.964 versus 0.971 for the default order, and the false-positive memory-activation rate remains 0.6% or lower through 1000 edits. These checks are deliberately labeled as N=1000 experiments and are not presented as repeated 50K runs.

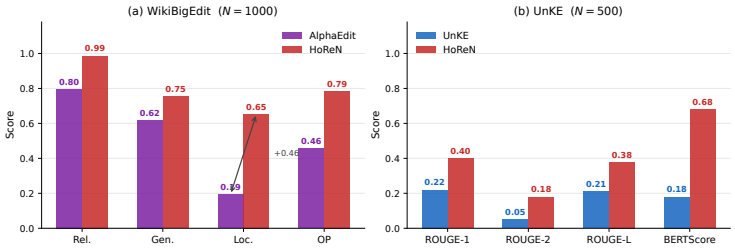


Figure 3: Cross-dataset generalization (LLaMA-3.1-8B). **(a) WikiBigEdit** (N=1000): HoReN outperforms AlphaEdit on all three metrics; AlphaEdit's locality collapses to 0.19 while HoReN retains 0.65, a gap of +0.46. **(b) UnKE** (N=500): The four scores in panel (b) are the *Original*-split values from UnKEBench (Deng et al., 2024) and therefore quantify reliability on the original edit question; the complementary *Paraphrase* (generalization) and *Sub-question* (factual coverage) splits, on which HoReN also leads at N=500, are reported in Table 8. Full results across all N in Tables 7–9 (Appendix G.1).

3.3 Generalization Across Datasets and Models

ZsRE provides a favorable separation regime; we now test whether HoReN's advantage holds under structurally different conditions along two orthogonal dimensions: dataset type (Section 3.3.1) and model family (Section 3.3.2).

3.3.1 Cross-Dataset Generalization

On WikiBigEdit, where locality queries share subjects and relations with edits, HoReN sustains Reliability 0.986–0.992, Generalization 0.753–0.767, and Locality 0.631–0.682 across $N \in \{500, 1000, 3000\}$ while AlphaEdit's Locality collapses to 0.169–0.267 in the same grid; HoReN still outperforms AlphaEdit on this benchmark. However, the editing samples from WikiBigEdit do not have clear (subject, relation, object) structures as in ZsRE, meaning there is considerable noise besides the knowledge representation at the layer being edited by HoReN, resulting in degraded performance in generalization and locality. On UnKE, whose targets are multi-sentence free-form passages, HoReN at N=500 leads the domain-specific UnKE baseline by +0.18 ROUGE-1 (0.40 vs. 0.22), +0.13 ROUGE-2, +0.17 ROUGE-L, and +0.50 BERTScore on the *Original* split, and also leads on *Paraphrase* (ROUGE-L 0.338 vs. 0.197, BERTScore 0.659 vs. 0.166) and *Sub-question* (ROUGE-L 0.332 vs. 0.137, ROUGE-1 0.338 vs. 0.141) splits at N=500 (Table 8), showing the advantage of HoReN's routing mechanism on unstructured data over the UnKE editor.

3.3.2 Cross-Model Generalization

Across Qwen2.5-7B, DeepSeek-R1-Distill-Llama-8B, GPT-OSS-20B, and Qwen-SEA-LION-v4-32B at $N \in \{2000, 5000, 10000\}$ (Table 2), HoReN reaches OP 0.86–0.98 on every cell (Rel. $\geq$ 0.99, Gen. $\geq$ 0.66, Loc. $\geq$ 0.97), while GRACE is stuck at OP 0.18–0.31 with Generalization 0.01–0.03 and WISE ranges from OP $\leq$ 0.13 on GPT-OSS-20B to 0.68–0.72 on SEA-LION-32B. The four backbones come from independent training pipelines with different tokenizers, pretraining corpora, and post-training procedures, so GRACE's near-identical flat-Generalization failure across all four (a span of 0.02) confirms the bottleneck is in the key-query comparison rather than in any model-specific representation property, and WISE's family-dependent collapse localizes its weakness to model-specific activation structure rather than to scale alone. HoReN's recipe is held fixed at β=20, γ=0.1, M=1 across all backbones; only the matching threshold and token pooling ratio change, listed once per family in Table 3, a single recipe transferring across seven backbones from 1.5B to 32B parameters reflects the directional structure of post-nonlinearity activations discussed in Section 3.4.

3.4 Why HoReN Works: Direction Is the Semantic Key

Direction, not magnitude, carries the fact. ROME framed MLP down-projections as a key–value associative memory, with the post-nonlinearity activation at the subject token serving as

the *key* into a linear store of facts and editing as the insertion of one (k, v) pair without disturbing the rest. GRACE adopts the codebook-of-edits paradigm but is deliberately layer-agnostic, attaching to any chosen layer and keying on the cached input activation h^{l-1} matched by Euclidean distance; HoReN takes one further step and recommends placing the codebook at the MLP, where ROME's analysis identifies the activations that actually behave as fact keys. Table 14 then probes whether magnitude or direction of these MLP activations carries the fact identity: holding codebook location, value adaptor, and matching threshold fixed and varying only the key–query comparison on LLaMA-3.1-8B (ZsRE, $N{=}1000$), unit-norm cosine matching alone lifts Generalization from 0.07 to 0.33 ($+0.26$) without measurable cost to Reliability or Locality (Loc. ≥ 0.99). Stripping magnitude is therefore informative: post-nonlinearity activation magnitude behaves like a per-prompt gain (driven by surface form, length, and upstream normalization), while the *direction* on the unit hypersphere carries the identity of the semantic.

Hopfield dynamics on the unit sphere control the generalization–locality tradeoff. Cosine matching alone leaves a residual angular gap: paraphrase queries still sit off-axis from their target keys, and one-shot argmax misroutes whenever that angle exceeds the matching threshold c. A damped softmax-attention update over the unit-norm codebook closes the gap, lifting Generalization from 0.33 to 0.93 in Table 14 (Figure 4 visualizes the two-step ladder), and the update is direction-selective rather than uniformly attractive: an unrelated query is roughly equidistant from all stored keys, the softmax is near-uniform, and the update barely moves it; a paraphrase query is close to one key, the softmax is sharply peaked, and the update nudges it into the basin. The selectivity requires both ingredients jointly, as Table 15 (Appendix I.1) shows that Hopfield refinement *without* normalization collapses Locality to ≤ 0.08, since unbounded magnitudes drag unrelated queries into spurious basins.

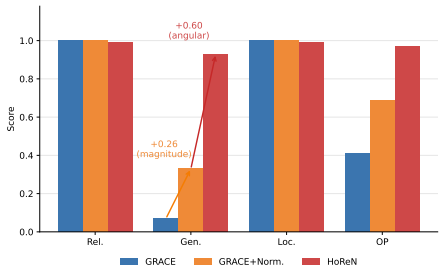


Figure 4: Representation gap diagnosis at $N{=}1000$ (ZsRE, LLaMA-3.1-8B). Generalization improves in two steps: normalization removes the magnitude component ($+0.26$); the Hopfield update closes the angular gap ($+0.60$). Reliability and Locality remain flat, confirming orthogonality of the two ingredients.

Direction is a stable property of the base model. Together these reads explain Tables 1 and 2: the codebook plus value adaptor delivers Reliability ≥ 0.99 on every row, while the routing layer sustains Generalization and Locality at scale by reading only direction. The cross-model parameter bands in Table 3 provide an independent confirmation: matching thresholds cluster in two tight, family-coherent bands (LLaMA-3 / LLaMA-3.1 at 0.80–0.85, Qwen-2.5 / SEA-LION at 0.55–0.60) that survive heavy reasoning post-training, while the rest of the recipe ($\beta{=}20$, $\gamma{=}0.1$, $M{=}1$) is held fixed across all seven backbones from 1.5B to 32B parameters. Token-pooling and value-adaptor (direct vs. LoRA) ablations are deferred to Appendices I.4 and I.3.

4 Conclusion

We studied lifelong model editing through the lens of memory-based editors and identified paraphrase routing under accumulated edits as the bottleneck of this paradigm: existing methods can store edits faithfully but cannot reliably activate the right one when the codebook grows. We proposed **HoReN**, a routing-layer mechanism that combines L2-normalized angular matching with a single damped Hopfield-style query refinement, motivated by an attractor-basin reading in which the deferral threshold serves as the basin boundary. Empirically, HoReN sustains all three editing metrics above 0.89 on a controlled 50K-edit ZsRE stress test, reaches OP 0.94–0.97 at $1{,}000$ edits across LLaMA and Qwen backbones, generalizes to structured WikiBigEdit and unstructured UnKE, and matches GRACE on edit/inference latency and parameter overhead. Limitations and future directions, linear side-memory growth, sensitivity to prompt structure, the asymptotic–single-step theory gap, and out-of-scope edit interactions, are discussed in Appendix A.

AI USE STATEMENT

We used AI assistants for language editing, LaTeX formatting, consistency checks, refining the presentation of mathematical arguments, and implementing and aligning parts of our method with the design algorithm. The authors manually reviewed and verified all AI-assisted manuscript content and reviewed the resulting code, and take full responsibility for the manuscript, implementation, and reported results.

REFERENCES

OpenAI Sandhini Agarwal and Others. gpt-oss-120b&gpt-oss-20b model card. 2025. URL https://api.semanticscholar.org/CorpusID:280671456.

Qizhou Chen, Taolin Zhang, Xiaofeng He, Dongyang Li, Chengyu Wang, Longtao Huang, and Hui Xue'. Lifelong knowledge editing for LLMs with retrieval-augmented continuous prompt learning. In Yaser Al-Onaizan, Mohit Bansal, and Yun-Nung Chen (eds.), *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing*, pp. 13565–13580, Miami, Florida, USA, November 2024. Association for Computational Linguistics. doi: 10.18653/v1/2024.emnlp-main.751. URL https://aclanthology.org/2024.emnlp-main.751/.

Jingcheng Deng, Zihao Wei, Liang Pang, Hanxing Ding, Huawei Shen, and Xueqi Cheng. Unke: Unstructured knowledge editing in large language models. *ArXiv*, abs/2405.15349, 2024. URL https://api.semanticscholar.org/CorpusID:282065712.

Abhimanyu Dubey et al. The llama 3 herd of models. 2024. URL https://api.semanticscholar.org/CorpusID:271571434.

Junfeng Fang, Houcheng Jiang, Kun Wang, Yunshan Ma, Xiang Wang, Xiangnan He, and Tat-Seng Chua. Alphaedit: Null-space constrained knowledge editing for language models. *ArXiv*, abs/2410.02355, 2024. URL https://api.semanticscholar.org/CorpusID:273098148.

Xiaojie Gu, Ziying Huang, Jia-Chen Gu, and Kai Zhang. Ultraedit: Training-, subject-, and memory-free lifelong editing in language models. 2025. URL https://api.semanticscholar.org/CorpusID:281659488.

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

Neil Houlsby, Andrei Giurgiu, Stanislaw Jastrzebski, Bruna Morrone, Quentin de Laroussilhe, Andrea Gesmundo, Mona Attariyan, and Sylvain Gelly. Parameter-efficient transfer learning for nlp. In *International Conference on Machine Learning*, 2019. URL https://api.semanticscholar.org/CorpusID:59599816.

Edward J Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Liang Wang, Weizhu Chen, et al. Lora: Low-rank adaptation of large language models. *Iclr*, 1(2):3, 2022.

Urvashi Khandelwal, Omer Levy, Dan Jurafsky, Luke Zettlemoyer, and Mike Lewis. Generalization through memorization: Nearest neighbor language models. *ArXiv*, abs/1911.00172, 2019. URL https://api.semanticscholar.org/CorpusID:207870430.

James Kirkpatrick, Razvan Pascanu, Neil Rabinowitz, Joel Veness, Guillaume Desjardins, Andrei A Rusu, Kieran Milan, John Quan, Tiago Ramalho, Agnieszka Grabska-Barwinska, et al. Overcoming catastrophic forgetting in neural networks. *Proceedings of the national academy of sciences*, 114 (13):3521–3526, 2017.

Dmitry Krotov. A new frontier for hopfield networks. *Nature Reviews Physics*, 5:366–367, 2023. URL https://api.semanticscholar.org/CorpusID:258812300.

Omer Levy, Minjoon Seo, Eunsol Choi, and Luke Zettlemoyer. Zero-shot relation extraction via reading comprehension. In *Proceedings of the 21st Conference on Computational Natural Language Learning (CoNLL 2017)*, pp. 333–342, 2017.

Patrick Lewis, Ethan Perez, Aleksandra Piktus, Fabio Petroni, Vladimir Karpukhin, Naman Goyal, Heinrich Küttler, Mike Lewis, Wen-tau Yih, Tim Rocktäschel, et al. Retrieval-augmented generation for knowledge-intensive nlp tasks. *Advances in neural information processing systems*, 33: 9459–9474, 2020.

Xiang Lisa Li and Percy Liang. Prefix-tuning: Optimizing continuous prompts for generation. *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pp. 4582–4597, 2021. URL https://api.semanticscholar.org/CorpusID:230433941.

Haihua Luo, Xuming Ran, Tommi J. Karkkainen, Zhong Chen, Jiangrong Shen, Qi Xu, and Fengyu Cong. Reversible lifelong model editing via semantic routing-based lora. 2026a. URL https://api.semanticscholar.org/CorpusID:286489211.

Haihua Luo, Xuming Ran, Zheng Li, Huiyan Xue, Tingting Jiang, Jiangrong Shen, Tommi J. Karkkainen, Qi Xu, and Fengyu Cong. Key-value pair-free continual learner via task-specific prompt-prototype. *Neural networks : the official journal of the International Neural Network Society*, 198:108576, 2026b. URL https://api.semanticscholar.org/CorpusID:284544134.

Kevin Meng, David Bau, Alex Andonian, and Yonatan Belinkov. Locating and editing factual associations in gpt. *Advances in Neural Information Processing Systems 35*, 2022a. URL https://api.semanticscholar.org/CorpusID:255825985.

Kevin Meng, Arnab Sen Sharma, Alex Andonian, Yonatan Belinkov, and David Bau. Mass-editing memory in a transformer. *ArXiv*, abs/2210.07229, 2022b. URL https://api.semanticscholar.org/CorpusID:252873467.

Eric Mitchell, Charles Lin, Antoine Bosselut, Chelsea Finn, and Christopher D. Manning. Fast model editing at scale. *ArXiv*, abs/2110.11309, 2021. URL https://api.semanticscholar.org/CorpusID:239050360.

Eric Mitchell, Charles Lin, Antoine Bosselut, Christopher D Manning, and Chelsea Finn. Memory-based model editing at scale. In *International Conference on Machine Learning*, pp. 15817–15831. PMLR, 2022.

Raymond Ng, Thanh Ngan Nguyen, et al. Sea-lion: Southeast asian languages in one network. In *IJCNLP-AACL*, 2025. URL https://api.semanticscholar.org/CorpusID:277450612.

Hubert Ramsauer, Bernhard Schafl, Johannes Lehner, Philipp Seidl, Michael Widrich, Lukas Gruber, Markus Holzleitner, Milena Pavlovi'c, Geir Kjetil Ferkingstad Sandve, Victor Greiff, David P. Kreil, Michael Kopp, Günter Klambauer, Johannes Brandstetter, and Sepp Hochreiter. Hopfield networks is all you need. *ArXiv*, abs/2008.02217, 2020. URL https://api.semanticscholar.org/CorpusID:220968978.

Xuming Ran, Jun Yao, Yusong Wang, Mingkun Xu, and Dianbo Liu. Brain-inspired continual pre-trained learner via silent synaptic consolidation. *ArXiv*, abs/2410.05899, 2024. URL https://api.semanticscholar.org/CorpusID:273228983.

DeepSeek-AI Team. Deepseek-r1 incentivizes reasoning in llms through reinforcement learning. *Nature*, 645:633 – 638, 2025. URL https://api.semanticscholar.org/CorpusID:275789950.

Lukas Thede, Karsten Roth, Matthias Bethge, Zeynep Akata, and Tom Hartvigsen. Wikibigedit: Understanding the limits of lifelong knowledge editing in llms. In *International Conference on Machine Learning*, 2025. URL https://api.semanticscholar.org/CorpusID:276885341.

Peng Wang, Zexi Li, Ningyu Zhang, Ziwen Xu, Yunzhi Yao, Yong Jiang, Pengjun Xie, Fei Huang, and Huajun Chen. Wise: Rethinking the knowledge memory for lifelong model editing of large language models. *ArXiv*, abs/2405.14768, 2024. URL https://api.semanticscholar.org/CorpusID:269982715.

Yisu Wang, Ming Wang, Haoyuan Song, Wenjie Huang, Chaozheng Wang, Yi Xie, and Xuming Ran. Repair: Robust editing via progressive adaptive intervention and reintegration. *arXiv preprint arXiv:2510.01879*, 2025.

Huiyan Xue, Xuming Ran, Yaxin Li, Qi Xu, Enhui Li, Yi Xu, and Qiang Zhang. Distillation-guided structural transfer for continual learning beyond sparse distributed memory. In *AAAI Conference on Artificial Intelligence*, 2025. URL https://api.semanticscholar.org/CorpusID:283920124.

Qwen An Yang, Baosong Yang, et al. Qwen2.5 technical report. *ArXiv*, abs/2412.15115, 2024. URL https://api.semanticscholar.org/CorpusID:274859421.

Binchi Zhang, Zhengzhang Chen, Zaiyi Zheng, Jundong Li, and Haifeng Chen. Resolving editing-unlearning conflicts: A knowledge codebook framework for large language model updating, 2025. URL https://arxiv.org/abs/2502.00158.

Ningyu Zhang, Yunzhi Yao, Bo Tian, Peng Wang, Shumin Deng, Mengru Wang, Zekun Xi, Shengyu Mao, Jintian Zhang, Yuansheng Ni, Siyuan Cheng, Ziwen Xu, Xin Xu, Jia-Chen Gu, Yong Jiang, Pengjun Xie, Fei Huang, Lei Liang, Zhiqiang Zhang, Xiaowei Zhu, Jun Zhou, and Huajun Chen. A comprehensive study of knowledge editing for large language models. *ArXiv*, abs/2401.01286, 2024. URL https://api.semanticscholar.org/CorpusID:266725300.

A Limitations and Future Directions

Four limitations frame the next steps for HoReN. *(i) Memory and retrieval scaling.* The codebook grows linearly at $\sim$4.08M parameters per 1K edits and exact retrieval costs $O(Kd)$ for K entries (Appendix F.3). We evaluate through 50K edits, not 500K–1M, and do not test whether approximate nearest-neighbor indexing preserves reliability, generalization, and locality. Compression, quantization, shared-basis value adaptors, and ANN indexing are complementary systems directions rather than solved contributions of this work. *(ii) Prompt and fact complexity.* HoReN is strongest on clean ZsRE facts. Its lower absolute WikiBigEdit scores indicate that single-layer, single-key routing can under-represent prompts with multiple entities, temporal qualifiers, long discourse, or compositional structure; multi-token/multi-layer keys and structure-aware pooling are promising extensions. *(iii) Theory–deployment gap.* Theorem 2.1 analyzes the standard, unnormalized Hopfield map, whereas HoReN deploys one normalized, damped step. The theory therefore motivates the over-attraction mechanism but does not bound routing error or locality for the actual operator. *(iv) Reasoning trajectories.* We evaluate final-answer reliability, generalization, and locality, but do not analyze intermediate reasoning trajectories, generated rationales, or chain-of-thought consistency after a hidden-state intervention; we make no claim that these remain unchanged.

B Related Work

Parameter-modifying editors. Locate-and-edit methods such as ROME (Meng et al., 2022a) and MEMIT (Meng et al., 2022b) modify internal weights via causal tracing or rank-constrained updates but suffer interference under sequential application, a variant of the catastrophic forgetting problem long studied in continual learning (Kirkpatrick et al., 2017). AlphaEdit (Fang et al., 2024) mitigates this with null-space projection, and UltraEdit (Gu et al., 2025) studies training-free unstructured editing. These methods concern how edits are *written* into the base weights; HoReN concerns how accumulated edits are *routed* at inference time.

Parameter-preserving / memory-based editors. Memory-based editors store updates externally and activate them on matching queries, with each method innovating on a different axis: DEFER (Mitchell et al., 2022) learns a refinement network for retrieval, GRACE (Hartvigsen et al., 2023) maintains a discrete codebook keyed on hidden states with a fixed ε-radius match, WISE (Wang et al., 2024) adds activation-based importance weighting, and REPAIR (Wang et al., 2025) enforces cross-layer consistency. RECIPE (Chen et al., 2024) jointly trains a retriever and continuous-prompt encoder with a dynamic relevance threshold, whereas LOKA (Zhang et al., 2025) uses similarity-aware codebook allocation and a learned activation router for combined editing and unlearning. These are especially relevant routing-oriented comparators and are included at $N{=}1000$ in Table 4. Value payloads are typically lightweight modules, LoRA (Hu et al., 2022), adapters (Houlsby et al., 2019), or prefix tuning (Li & Liang, 2021), recently surveyed by Zhang et al. (2024). Conceptually this family is close to retrieval-augmented language modeling (Khandelwal et al., 2019; Lewis et al., 2020), but with keys and values constructed per edit. HoReN renovates the routing rule along two specific axes: (i) it L2-normalizes keys and queries so matching depends on angular similarity rather than magnitude, and (ii) it applies a single damped Hopfield-style refinement step on the query before the final argmax; value adaptors are kept modular (direct vector or LoRA).

Associative retrieval with Hopfield networks. Modern Hopfield networks (Ramsauer et al., 2020) extend classical associative memory with continuous energy functions and high storage capacity, retrieving patterns through attractor dynamics that pull noisy queries toward stored memories, with connections to attention and few-shot learning (Krotov, 2023). To our knowledge, this perspective has not been brought to large-scale sequential model editing; HoReN borrows the energy/attractor view to motivate a lightweight retrieval refinement rather than to deploy full Hopfield dynamics.

C METHOD DETAILS

C.1 ALGORITHM PSEUDOCODE

Algorithm 1: Codebook Update at Layer l

Input: Codebook $\mathcal{C} = \{(k_i, v_i, y_i)\}_{i=1}^{|C|}$, possibly empty. Model with codebook params f_{θ_C}.
Input: Edit sample $(\mathbf{x}_t, \mathbf{y}_t^*)$
Input: Hopfield params $(\gamma, \beta, \epsilon)$, iterations M, match threshold c, adaptor steps U
Output: Updated codebook $\mathcal{C}$

$(H_{\text{in}}, H_{\text{out}}) \leftarrow f_\theta(\mathbf{x}_t)$; `// H_in: layer l input , H_out: layer l output`
$q_0 \leftarrow \text{normalize}(\text{select}(H_{\text{in}}))$;
$q \leftarrow q_0$;
if $\mathcal{C} = \emptyset$ **then**
  $v_{\text{new}} \leftarrow$ INITVALUEADAPTOR();
  for $u \leftarrow 1$ **to** U **do**
    $H_{\text{out}} \leftarrow H_{\text{out}} + v_{\text{new}}$;
    Compute $\mathcal{L}_{\text{CE}}(f_{\theta_C}(\mathbf{x}_t), \mathbf{y}_t^*)$ and update only v_{new} by gradient descent;
    if *early stopping criterion is met* **then**
      break;
  $\mathcal{C} \leftarrow \mathcal{C} \cup \{(q_0, v_{\text{new}}, \mathbf{y}_t^*)\}$;
  return $\mathcal{C}$;
$K \leftarrow \text{stack}(\{k_i : (k_i, v_i, y_i) \in \mathcal{C}\})$;
for $m \leftarrow 1$ **to** M **do**
  $p \leftarrow \text{softmax}(\beta\, qK^\top)$;
  $q_{\text{new}} \leftarrow \text{normalize}(pK)$;
  if $||q_{new} - q||_2 \leq \epsilon$ **then**
    `// Early Stop`
    break;
  $q \leftarrow \text{normalize}\big((1-\gamma)q + \gamma q_{\text{new}}\big)$;
$\mathbf{a} \leftarrow qK^\top$;
$i^\star \leftarrow \arg\max_i \mathbf{a}_i$;
if $\mathbf{a}_{i^\star} \leq c$ ***or*** $y_{i^\star} \neq \mathbf{y}_t^*$ **then**
  `// No match or label conflict: add a new entry`
  $k_{\text{new}} \leftarrow q_0$;
  $v_{\text{new}} \leftarrow$ INITVALUEADAPTOR();
  for $u \leftarrow 1$ **to** U **do**
    $H_{\text{out}} \leftarrow H_{\text{out}} + v_{\text{new}}$;
    Compute $\mathcal{L}_{\text{CE}}(f_{\theta_C}(\mathbf{x}_t), \mathbf{y}_t^*)$ and update only v_{new} by gradient descent;
    if *early stopping criterion is met* **then**
      break;
  $\mathcal{C} \leftarrow \mathcal{C} \cup \{(k_{\text{new}}, v_{\text{new}}, \mathbf{y}_t^*)\}$;
else
  `// Match with consistent label: use existing entry`
  for $u \leftarrow 1$ **to** U **do**
    $H_{\text{out}} \leftarrow H_{\text{out}} + v_{i^\star}$;
    Compute $\mathcal{L}_{\text{CE}}(f_{\theta_C}(\mathbf{x}_t), \mathbf{y}_t^*)$ and update only $v_{i^\star}$ by gradient descent;
    if *early stopping criterion is met* **then**
      break;
return $\mathcal{C}$

D PROOFS OF THEORETICAL RESULTS

D.1 SETUP AND DESCENT INEQUALITY

The two results below share a common Lyapunov construction. We isolate the per-step descent inequality first; the asymptotic theorem and finite-step proposition then follow as two complementary readings of it.

Lemma D.1 (Per-step descent of standard Hopfield retrieval)**.** *Let $K \in \mathbb{R}^{C\times d}$ be a codebook with rows satisfying $\|k_i\|_2 = 1$, let $q^{(0)} \in \mathbb{R}^{1\times d}$ satisfy $\|q^{(0)}\|_2 = 1$, and let $\beta > 0$. Define*

$$T(q) := \mathrm{softmax}(\beta q K^\top)\, K, \quad F(q) := \tfrac{1}{\beta}\log\sum_{i=1}^{C}\exp(\beta\, q k_i^\top), \quad E(q,K) := \tfrac{1}{2}\|q\|_2^2 - F(q).$$

Then for every $s \geq 0$ the iterate $q^{(s+1)} = T(q^{(s)})$ satisfies

$$E(q^{(s+1)}, K) - E(q^{(s)}, K) \;\leq\; -\tfrac{1}{2}\,\|q^{(s+1)} - q^{(s)}\|_2^2.$$

Proof. We construct a Lyapunov function. For $q \in \mathbb{R}^{1\times d}$, the potential F is the log-sum-exp of the similarities,

$$F(q) := \frac{1}{\beta}\log\left(\sum_{i=1}^{C}\exp(\beta\, q k_i^\top)\right) = \mathrm{lse}_\beta(qK^\top), \tag{4}$$

where $qk_i^\top$ denotes the dot product between the query and the i-th key. The function $F(q)$ is convex in q. Its gradient with respect to q is:

$$\nabla_q F(q) = \frac{\sum_{i=1}^{C} k_i \exp(\beta\, q k_i^\top)}{\sum_{i=1}^{C}\exp(\beta\, q k_i^\top)} = \mathrm{softmax}(\beta\, qK^\top)K = pK, \tag{5}$$

where $p = \mathrm{softmax}(\beta\, qK^\top) \in \mathbb{R}^{1\times C}$. Consequently, the standard Hopfield update corresponds exactly to the gradient mapping:

$$q^{(s+1)} = \nabla_q F(q^{(s)}). \tag{6}$$

Define the energy function as $E(q,K) := \frac{1}{2}\|q\|_2^2 - F(q)$. Since F is convex, it satisfies the first-order inequality:

$$F(y) \geq F(x) + \langle \nabla_q F(x), (y-x)^\top\rangle \tag{7}$$

for all $x, y \in \mathbb{R}^{1\times d}$. Letting $x = q^{(s)}$ and $y = q^{(s+1)}$, and using (6), we obtain:

$$F(q^{(s+1)}) - F(q^{(s)}) \geq \langle q^{(s+1)}, (q^{(s+1)} - q^{(s)})^\top\rangle = q^{(s+1)}(q^{(s+1)} - q^{(s)})^\top. \tag{8}$$

We now evaluate the change in energy between consecutive iterations:

$$\begin{aligned} E(q^{(s+1)},K) - E(q^{(s)},K) &= \frac{1}{2}\|q^{(s+1)}\|_2^2 - \frac{1}{2}\|q^{(s)}\|_2^2 - \Big(F(q^{(s+1)}) - F(q^{(s)})\Big) \\ &\leq \frac{1}{2}\|q^{(s+1)}\|_2^2 - \frac{1}{2}\|q^{(s)}\|_2^2 - q^{(s+1)}(q^{(s+1)} - q^{(s)})^\top \\ &= -\frac{1}{2}\|q^{(s+1)} - q^{(s)}\|_2^2 \leq 0. \end{aligned} \tag{9}$$

This establishes the descent inequality. □ □

The descent inequality has two complementary readings. *Asymptotically* ($s \to \infty$), it forces $\|q^{(s+1)} - q^{(s)}\|_2 \to 0$ and, combined with real-analyticity of E, gives single-limit convergence to a fixed point of T. *Finitely* (any $M \geq 1$), it gives a cumulative bound on the squared increments, hence a $1/\sqrt{M}$ residual rate for the best of the first M iterates (Proposition 2.2). We state and prove the two results separately below.

D.2 Proof of Theorem 2.1: Asymptotic Convergence

Theorem 2.1 (restated). *Under the assumptions above, the sequence $\{q^{(s)}\}_{s=0}^{\infty}$ generated by $q^{(s+1)} = T(q^{(s)})$ converges to a fixed point q^*, i.e., $q^{(s)} \to q^*$ as $s \to \infty$, where $q^* = T(q^*) = \mathrm{softmax}(\beta q^* K^\top) K$.*

Proof. Step 1: Boundedness of iterates and lower-boundedness of E. Since $q^{(s+1)} = \mathrm{softmax}(\beta q^{(s)} K^\top)K$ is a convex combination of unit-norm rows of K, $q^{(s+1)} \in \mathrm{conv}\{k_1, \dots, k_C\}$ and $\|q^{(s+1)}\|_2 \le 1$. Together with $\|q^{(0)}\|_2 = 1$, the full sequence is bounded. For any q,

$$F(q) = \tfrac{1}{\beta} \log \sum_{i=1}^{C} \exp(\beta\, q k_i^\top) \le \max_i q k_i^\top + \frac{\log C}{\beta} \le \|q\|_2 + \frac{\log C}{\beta},$$

so $E(q, K) \ge \frac{1}{2}\|q\|_2^2 - \|q\|_2 - \frac{\log C}{\beta} \ge -\frac{1}{2} - \frac{\log C}{\beta}$. Thus E is bounded below.

Step 2: Square-summability of increments. Set $E_s := E(q^{(s)}, K)$ and $d_s := q^{(s+1)} - q^{(s)}$. Lemma D.1 gives $E_s - E_{s+1} \ge \frac{1}{2}\|d_s\|_2^2$. Since $\{E_s\}$ is monotone and bounded below, it converges to some E_∞, and

$$\sum_{s=0}^{\infty} \|d_s\|_2^2 \le 2(E_0 - E_\infty) < \infty,$$

so $\|d_s\|_2 \to 0$.

Step 3: Accumulation points are fixed points. Let Ω be the set of accumulation points of $\{q^{(s)}\}$; boundedness gives $\Omega \neq \emptyset$. For $q^* \in \Omega$, pick a subsequence $q^{(s_j)} \to q^*$. Since $\|d_{s_j}\|_2 \to 0$, $q^{(s_j+1)} \to q^*$, and $q^{(s_j+1)} = T(q^{(s_j)}) \to T(q^*)$ by continuity of T. Hence $q^* = T(q^*)$.

Step 4: Single-limit convergence via Kurdyka-Łojasiewicz. Since F is real analytic, so is E, and E satisfies the Kurdyka-Łojasiewicz (KL) property. Note that $\nabla E(q) = q - \nabla F(q) = q - T(q)$, so $\nabla E(q^{(s)}) = q^{(s)} - q^{(s+1)} = -d_s$. Since $E_s \to E_\infty$, every $q^* \in \Omega$ satisfies $E(q^*, K) = E_\infty$. Moreover, boundedness of the sequence and the definition of its accumulation set imply $\mathrm{dist}(q^{(s)}, \Omega) \to 0$. By the uniformized KL property on the compact set Ω, there exist $\eta > 0$, $\varepsilon > 0$, and a concave desingularizing φ with $\varphi(0) = 0$, $\varphi' > 0$, such that for all sufficiently large s with $E_s > E_\infty$,

$$\varphi'(E_s - E_\infty)\, \|\nabla E(q^{(s)})\|_2 \ge 1.$$

(If $E_s = E_\infty$ for some s, the descent inequality forces $d_s = 0$ and the sequence is constant from that point on.) By concavity of φ,

$$\begin{aligned}\varphi(E_s - E_\infty) - \varphi(E_{s+1} - E_\infty) &\ge \varphi'(E_s - E_\infty)(E_s - E_{s+1}) \\ &\ge \frac{E_s - E_{s+1}}{\|\nabla E(q^{(s)})\|_2} = \frac{E_s - E_{s+1}}{\|d_s\|_2} \ge \tfrac{1}{2}\|d_s\|_2.\end{aligned}$$

Hence $\|d_s\|_2 \le 2\big(\varphi(E_s - E_\infty) - \varphi(E_{s+1} - E_\infty)\big)$, and summing gives $\sum_{s=0}^{\infty} \|d_s\|_2 < \infty$. Therefore $\{q^{(s)}\}$ has finite length, is Cauchy, and converges to some q^*. Combined with Step 3, $q^* = T(q^*) = \mathrm{softmax}(\beta q^* K^\top) K$. □ □

D.3 Proof of Proposition 2.2: Finite-Step Descent and Residual Bound

Proposition 2.2 (restated). *Under the assumptions of Theorem 2.1, let $M \ge 1$ be a finite integer and generate $q^{(s+1)} = T(q^{(s)})$, $s = 0, 1, \dots, M-1$. Then, for every s, $E(q^{(s+1)}, K) \le E(q^{(s)}, K) - \frac{1}{2}\|q^{(s+1)} - q^{(s)}\|_2^2$, and consequently $\sum_{s=0}^{M-1} \|q^{(s+1)} - q^{(s)}\|_2^2 \le 2\big(E(q^{(0)}, K) - E(q^{(M)}, K)\big)$. Let $r_s := \|T(q^{(s)}) - q^{(s)}\|_2 = \|q^{(s+1)} - q^{(s)}\|_2$ be the fixed-point residual at step s. For any lower bound $E_{\inf}$ of E, $\frac{1}{M}\sum_{s=0}^{M-1} r_s^2 \le \frac{2(E(q^{(0)}, K) - E_{\inf})}{M}$ and $\min_{0 \le s < M} r_s \le \sqrt{2(E(q^{(0)}, K) - E_{\inf})/M}$. 20With the explicit constants $E_{\inf} = -\frac{1}{2} - \frac{\log C}{\beta}$ and the normalization $\|q^{(0)}\|_2 = 1$, $\|k_i\|_2 = 1$, this yields $\min_{0 \le s < M} \|T(q^{(s)}) - q^{(s)}\|_2 \le 2/\sqrt{M}$.*

Proof. The per-step descent inequality is exactly Lemma D.1. Summing it from $s = 0$ to $M - 1$ gives

$$E(q^{(M)}, K) - E(q^{(0)}, K) \le -\tfrac{1}{2} \sum_{s=0}^{M-1} \|q^{(s+1)} - q^{(s)}\|_2^2,$$

or equivalently $\sum_{s=0}^{M-1} r_s^2 \le 2\big(E(q^{(0)}, K) - E(q^{(M)}, K)\big)$.

If $E_{\inf} \le E(q, K)$ for all q, then $E(q^{(M)}, K) \ge E_{\inf}$, so

$$\sum_{s=0}^{M-1} r_s^2 \;\le\; 2\big(E(q^{(0)}, K) - E_{\inf}\big).$$

Dividing by M and using $\min_s r_s^2 \le \frac{1}{M} \sum_s r_s^2$ yields the residual bound.

For the explicit constant, Step 1 of the proof of Theorem 2.1 gives $E(q, K) \ge -\frac{1}{2} - \frac{\log C}{\beta}$, so we may take $E_{\inf} = -\frac{1}{2} - \frac{\log C}{\beta}$. Under the normalization $\|q^{(0)}\|_2 = 1$, $\|k_i\|_2 = 1$, $q^{(0)} k_i^\top \ge -1$ for every i, hence

$$\sum_{i=1}^{C} \exp(\beta\, q^{(0)} k_i^\top) \ge C e^{-\beta}, \qquad F(q^{(0)}) \ge -1 + \frac{\log C}{\beta},$$

and thus

$$E(q^{(0)}, K) = \tfrac{1}{2} - F(q^{(0)}) \le \tfrac{3}{2} - \frac{\log C}{\beta}.$$

Combining with $E_{\inf} = -\frac{1}{2} - \frac{\log C}{\beta}$ gives $E(q^{(0)}, K) - E_{\inf} \le 2$, and therefore

$$\min_{0 \le s < M} r_s \;\le\; \sqrt{\frac{2 \cdot 2}{M}} = \frac{2}{\sqrt{M}}. \qquad \square$$

□

Interpretation of the finite-step bound. Proposition 2.2 controls the *best* of the first M iterates: there exists $\bar{s} \in \arg\min_{0 \le s < M} \|T(q^{(s)}) - q^{(s)}\|_2$ with residual at most $2/\sqrt{M}$. It does *not* certify that the last iterate $q^{(M)} = T^M(q^{(0)})$ is itself an approximate fixed point, only $E(q^{(M)}, K) \le E(q^{(0)}, K)$ is guaranteed. Whether $\|T(q^{(M)}) - q^{(M)}\|_2$ is small at the last step must be checked separately. This distinction matters for HoReN: the algorithm fixes $M = 1$ and outputs $q^{(1)}$ rather than the residual-minimizing iterate, so the design relies on the asymptotic-attraction picture of Theorem 2.1 as a warning, namely, that running $M \gg 1$ steps would contract *every* query, including unrelated ones, rather than as a route to a tighter approximate fixed point.

E Supplementary Experimental Setup Details

In this appendix, we provide a detailed description of the experimental configuration, including an introduction to the datasets, a comprehensive explanation of the evaluation metrics, the implementation details, and a discussion of the baselines.

E.1 Datasets

Here, we provide a detailed introduction to the datasets used in this paper:

- **ZsRE** (Levy et al., 2017) is a question-answering dataset that has become the standard benchmark for lifelong knowledge editing. Edit prompts are factual questions paired with target answers, and rephrased prompts are produced via back-translation to evaluate generalization. Following prior work (Mitchell et al., 2021; Hartvigsen et al., 2023), natural questions unrelated to the edited fact are used as out-of-scope queries to evaluate locality. Each sample contains (i) an editing prompt, (ii) the target answer, (iii) a paraphrased prompt, and (iv) a locality prompt with its own ground-truth answer. We use the split from Mitchell et al. (2021) (>15K samples); the 50K stress test is constructed from the same source distribution (Appendix E.4.4).

- **WikiBigEdit** (Thede et al., 2025) is a large-scale benchmark constructed from Wikidata triples that introduces a substantially harder locality regime: locality queries are sampled from triples sharing the same subject and related relations as the edited facts, placing them semantically close to edit queries. This setting tests whether an editor can preserve clean angular separation between edits and unrelated knowledge even when the two are topically adjacent, a property that is trivially satisfied on ZsRE but fails for many existing methods on WikiBigEdit.
- **UnKE** (Deng et al., 2024) is an unstructured knowledge editing benchmark in which edits are free-form passages rather than triples and target answers are multi-sentence generations rather than short tokens. UnKE serves as a probe into the robustness of an editor beyond the structured (subject, relation, object) regime. Because targets are free-form, evaluation uses surface- and semantic-similarity metrics (ROUGE-1/2/L and BERTScore) instead of token-level exact match (see Appendix E.3.2).

E.2 BASELINES

We describe the baselines used across the experiments. The descriptions are grouped by the binary categorization introduced in Section 3.1: (I) parameter-modifying methods, which write closed-form updates into the base model's feed-forward weights, and (II) parameter-preserving methods, which leave every base feed-forward weight frozen and store edits in an external/side module consulted at inference. The UnKE baseline is included only for the unstructured editing benchmark, as stated in Section 3.1.

(I) Parameter-modifying baselines.

- **ROME** (Meng et al., 2022a): a single-fact editor for structured (subject, relation, object) triples. ROME first runs a causal-tracing analysis to identify the single mid-layer feed-forward block most responsible for recalling a given fact, then treats that block as an associative key–value memory in which the "key" is the internal representation of the subject and the "value" is the representation that triggers the object. To install a new fact it computes a closed-form, rank-one adjustment to that block's output projection so that the subject's key is mapped to the new object's representation while the responses for all other previously cached keys are left as unchanged as possible. ROME edits one fact at a time and modifies the base model's weights in place.
- **AlphaEdit** (Fang et al., 2024): a closed-form batch editor designed for sequential editing. AlphaEdit performs a MEMIT-style update to a feed-forward block, but before applying the update it first projects the proposed weight change onto the subspace orthogonal to the internal representations of a sample of preserved facts drawn from a general corpus. Because the update is forced to lie in this "do-not-disturb" subspace, applying it to the model leaves the outputs for those preserved representations exactly unchanged, which lets many edits be stacked sequentially with reduced interference on previously stored knowledge.
- **UltraEdit** (Gu et al., 2025): a training-free, subject-free, and memory-free editor. UltraEdit requires no optimizer training, no identification of a subject token, and stores no record of past edits. From a single forward pass it reads the hidden representation at a chosen layer and one gradient signal that points in the direction of the desired output, and combines them in closed form into a one-step adjustment of the layer's weights. As successive edits arrive it maintains a running normalization of the statistics of those hidden representations so that later edits remain well-scaled relative to earlier ones, enabling long edit streams.
- **UnKE** (Deng et al., 2024): a domain-specific baseline for unstructured edits whose targets are free-form multi-sentence passages rather than short entity strings. UnKE extends the locate-and-edit recipe by working at the granularity of an entire layer's input/output rather than a single subject token: the "key" becomes the chosen layer's input representation of the whole question and the "value" becomes the layer-output representation that, if produced, would cause the rest of the network to generate the desired passage. UnKE first solves a causal optimization to determine that target layer-output, then updates the feed-forward weights of that layer so the question's input representation produces it.

(II) Parameter-preserving baselines.

- **GRACE** (Hartvigsen et al., 2023): a lifelong editor that leaves all model weights frozen and instead maintains an external lookup table whose entries are pairs of (a stored hidden representation, the replacement hidden representation that should be produced at one chosen layer). Each entry also owns a small "acceptance radius" around its stored representation. At inference, the current activation at that layer is compared against every stored entry; if it falls inside any entry's acceptance radius, the activation is overwritten by that entry's replacement, otherwise the model runs unchanged. Entries are added, shrunk, or merged as new edits arrive, so the lookup table grows monotonically with the edit stream.
- **WISE** (Wang et al., 2024): a lifelong editor that picks one mid-to-late feed-forward block of the base model and never modifies its weights. WISE allocates an extra copy of just that block's value-projection matrix as a "side memory", initialized identically to the original. All edits are written into the side copy only. To prevent edits from interfering with each other, the edit history is split into shards: each shard is trained on a different random subset of the side copy's parameters, and the resulting shards are then combined into one side copy via the Ties-merging procedure – this merging happens across the side shards only and the side copy is never folded back into the original block. At inference, a routing rule measures how strongly each input token activates the difference between the side copy and the original block; tokens whose activation crosses a learned threshold are sent through the side copy, all others go through the untouched original block.
- **REPAIR** (Wang et al., 2025): a closed-loop extension of the WISE side-memory paradigm. REPAIR keeps the base feed-forward weights frozen and stores edits in K shard copies of one mid-to-late FFN value matrix, routed at inference by an activation-score margin between each shard and the original block. On top of pure side-memory routing it adds three components to stabilize long edit streams: (i) distribution-aware inner-batch knowledge distillation that groups semantically similar edits and aligns their hidden representations before writing, (ii) a closed-loop error-feedback controller that monitors per-shard accuracy after each edit window and prunes-and-retrains the worst-performing shard on the accumulated error pool, and (iii) loss-aware Weighted TIES merging that fuses the shard deltas into a single side memory with weights inversely proportional to each shard's training loss.

E.3 METRICS

We evaluate knowledge editing methods along three axes, reliability, generalization, and locality, following the evaluation framework of Wang et al. (2024). All metrics are computed over a set of N edit instances $\{(x_i, y_i)\}_{i=1}^{N}$, where x_i is the edit prompt and y_i is the desired target answer. We denote the original unedited model by f_{θ_0} and the model after sequentially applying all N edits by f_{θ_N}. Let $\mathcal{V}$ denote the vocabulary. Given a model f and prompt x, we obtain a predicted sequence $\hat{y} = (\hat{t}_1, \ldots, \hat{t}_L)$ via greedy autoregressive decoding,

$$\hat{t}_j = \arg\max_{v \in \mathcal{V}} \; P_f\big(v \mid x, \hat{t}_1, \ldots, \hat{t}_{j-1}\big)\,, \tag{10}$$

where L is fixed to the number of tokens in the target y_i. We define the per-instance task measure

$$m(y, \hat{y}) = \frac{1}{L} \sum_{j=1}^{L} \mathbf{1}[\hat{t}_j = t_j], \tag{11}$$

which computes the fraction of correctly predicted tokens.

E.3.1 STRUCTURED METRICS (ZSRE & WIKIBIGEDIT)

Following the previous work (Mitchell et al., 2021; Hartvigsen et al., 2023; Wang et al., 2024), the three metrics on structured benchmarks are defined as follows.

- **Reliability** measures whether the edited model produces the target answer for the original edit prompt:

$$\text{Reliability} = \frac{1}{N} \sum_{i=1}^{N} m\Big(y_i,\ \hat{y}_i^{\,f_{\theta_N}, x_i}\Big)\,. \tag{12}$$

- **Generalization** evaluates whether the edit transfers to semantically equivalent rephrasings x'_i of the edit prompt:

$$\text{Generalization} = \frac{1}{N}\sum_{i=1}^{N} m\Big(y_i,\ \hat{y}_i^{f_{\theta_N}, x'_i}\Big). \tag{13}$$

- **Locality** quantifies the extent to which the edit preserves the model's behavior on unrelated inputs. For each locality prompt x_i^ℓ with associated ground-truth length L_i^ℓ, we extract the last L_i^ℓ tokens from the full output sequence of both the original and edited models, denoted $z_i^{f_{\theta_0}}$ and $z_i^{f_{\theta_N}}$ respectively, and compute

$$\text{Locality} = \frac{1}{N}\sum_{i=1}^{N}\frac{1}{L_i^\ell}\sum_{j=1}^{L_i^\ell}\mathbf{1}\Big[z_{i,j}^{f_{\theta_0}} = z_{i,j}^{f_{\theta_N}}\Big]. \tag{14}$$

  Notably, locality compares the original and edited model outputs to each other rather than against a ground-truth label, directly measuring behavioral preservation.

The Overall Performance (OP) score reported alongside the three axes is their geometric mean, $\text{OP} = (\text{Reliability} \cdot \text{Generalization} \cdot \text{Locality})^{1/3}$.

E.3.2 Unstructured Metrics (UnKE)

Because UnKE targets are multi-sentence free-form passages, the per-token measure $m(y, \hat{y})$ is replaced by surface-level and semantic-level similarity scores between the generated answer $\hat{y}_i$ and the reference y_i. Following the original UnKE protocol (Deng et al., 2024), the main text reports BLEU, ROUGE-1, ROUGE-2, ROUGE-L, and BERTScore. Appendix G.1 additionally includes the full UnKEBench view with Original, Paraphrase, and Sub-question splits and metric descriptions; these correspond to reliability, generalization, and factual coverage for unstructured passages, rather than to the structured locality metric used for ZsRE and WikiBigEdit.

E.4 Implementation Details

E.4.1 Baseline Hyperparameters

For every baseline we adopt the original hyperparameters published by the corresponding authors and only re-map architecture-specific module names (e.g., `mlp.down_proj` for LLaMA/Qwen, `self_attn.o_proj` for GPT-OSS) when porting a recipe to a model that the original paper did not study. Appendix H separately reports five-seed HoReN and AlphaEdit runs and a shuffled-order test.

ROME. We use the Meng et al. (2022a) recipe verbatim: rewrite layer ℓ=5, `subject_last` fact-token, 25 value-optimization steps with learning rate 0.5, weight decay 10^{-3}, KL factor 0.0625, clamp norm factor 4, and second-moment adjustment with K_0 statistics estimated from 10^5 Wikipedia samples in float32. The value-loss layer is set to the last transformer block, i.e., $v_{\text{loss}} = n_{\text{layers}} - 1$ for every LLM.

AlphaEdit. We follow the Fang et al. (2024) LLaMA-3 configuration: a 5-layer rewrite window $\{4, 5, 6, 7, 8\}$, λ=15,000 for K_0 adjustment, 25 value-optimization steps with learning rate 0.1, weight decay 0.5, clamp norm 0.75, KL factor 0.0625, L_2 regularization 10, and null-space threshold 2×10^{-2}. The null-space projection P is precomputed once per model from the same 10^5-sample Wikipedia covariance. The value-loss layer is again $n_{\text{layers}} - 1$ (31 for the 8B LLaMAs, 27 for both 28-layer Qwen-based models, 63 for the 64-layer SEA-LION-32B).

GRACE. We use the Hartvigsen et al. (2023) defaults: codebook learning rate 1.0 (SGD) with 100 inner steps, Euclidean distance, cold value initialization, `replace_last` token replacement, coverage-based ϵ-expansion with initial ϵ=1 and early-stop regularization. Edits are applied to a single MLP `down_proj` layer per LLM (see the cross-method layer alignment below).

WISE. We adopt the Wang et al. (2024) Tab. 10 recipe: edit learning rate 1.0 (SGD), 70 inner iterations, mask ratio ρ=0.2, activation margins (α, β, γ)=$(5, 20, 10)$, activation ratio 0.88, side-memory merge frequency 1000 with TIES merging at density 0.53 and weight 1.0, and retrieval enabled. Norm constraint is 1.0. Edits are applied to a single MLP `down_proj` layer per LLM (see the cross-method layer alignment below).

REPAIR. We follow the Wang et al. (2025) configuration verbatim, using their official released hyperparameters (Wang et al. (2025), Appendix C, Table 6) for every backbone we share with that paper: edit learning rate 0.9 (SGD) with up to 10,000 inner iterations, mask ratio ρ=0.2, error threshold τ_{correct}=0.85, distillation weight λ_{KD}=1.0 with temperature 2.0, routing margins $(\gamma_1, \gamma_2, \gamma)$ and activation ratio set to the per-model values listed in their Table 6 (γ_1=2, γ_2=20, γ=10, act. ratio 0.20 for LLaMA-3 layer 29; γ_1=5, γ_2=20, γ=0, act. ratio 0.88 for Qwen2.5 layer 23), and Weighted-TIES shard merging with loss-temperature λ=0.2. Edits are applied to the same MLP `down_proj` layer used by GRACE/WISE/HoReN (see the cross-method layer alignment below); for backbones not studied by Wang et al. (2025) we transfer the same-family recipe (LLaMA-3 settings for the LLaMA-derived models and GPT-OSS, Qwen-2.5 settings for Qwen-derived models).

RECIPE and LOKA. For Table 4, we run the released RECIPE (Chen et al., 2024) and LOKA (Zhang et al., 2025) implementations exactly matching the original papers under the same ZsRE stream and N=1000 metric protocol used by HoReN.

UltraEdit. We use the Gu et al. (2025) recipe: lifelong learning rate 10^{-6}, `token=mask`, editor batch size 1024, with the published `gate_proj` + `up_proj` layer ranges for LLaMA-3-8B (layers 11–15 / 18–24) and Qwen2.5-7B (layers 18–26). For DeepSeek-R1-Distill-Qwen-1.5B (28 layers) and Qwen-SEA-LION-v4-32B (64 layers), where no published recipe exists, we transfer the Qwen recipe (layers 16–20 and 18–26 respectively).

E.4.2 HoReN Hyperparameters

HoReN uses β=20, γ=0.1, and M=1 across all experiments for the Hopfield-style refinement.

When a new entry is created (Algorithm 1), the base model is frozen and only the codebook value adaptor parameters are optimized by Adam on the per-token cross-entropy loss of the target answer. Per-edit adaptor optimization uses learning rate 0.1, up to U=50 optimization steps per edit, and early stopping when the per-token loss falls below 10^{-2} or fails to improve for 3 consecutive steps. No gradient flows back into the base model or into stored codebook keys; each codebook value adaptor is trained in isolation. The choice of layer where the HoReN codebook resides is specified in Appendix E.4.3.

Two other model-specific parameters are required: the matching threshold c and the token pooling ratio. They are selected once per model using the reported ablations and then held fixed across datasets and edit scales. Per-model values are listed in Table 3. HoReN uses tensor values as the default codebook value adaptor; the optional LoRA variant uses rank r=4.

Table 3 lists the two model-specific parameters, matching threshold c and token pooling ratio, for each backbone. Both are selected once using the calibration procedure in Section 3.4 and held fixed across all datasets and edit scales. The fixed-threshold runs retain locality above 0.96 through 10K edits and 0.89 at 50K on LLaMA-3.1-8B; this demonstrates scale transfer of one calibration, not a full threshold-sensitivity sweep at 50K.

Table 3: Per-model hyperparameters used in all experiments.

Family	Model	Threshold c	Pooling ratio
LLaMA-3/3.1	LLaMA-3-8B-Instruct	0.80	50%
	LLaMA-3.1-8B-Instruct	0.85	60%
	DeepSeek-R1-Distill-Llama-8B	0.85	60%
Qwen-2.5/3	Qwen2.5-7B-Instruct	0.55	60%
	DeepSeek-R1-Distill-Qwen-1.5B	0.60	60%
	Qwen-SEA-LION-v4-32B-IT	0.55	60%
GPT-OSS	GPT-OSS-20B	0.80	60%

Code, commands, and logs are available at `https://anonymous.4open.science/r/elephant-4B7F`. Portability guidelines are in Appendix E.4.5.

E.4.3 Cross-method alignment and Hardware

Cross-method layer alignment. To avoid confounding HoReN's gains with a more favorable layer choice, GRACE, WISE, and HoReN are configured to edit the same layer per LLM: layer 29 (LLaMA-3 / 3.1 / DeepSeek-R1-Distill-LLaMA-8B), layer 24 (Qwen2.5-7B-Instruct), layer 27 (DeepSeek-R1-Distill-Qwen-1.5B), layer 56 (SEA-LION-32B), and layer 20 of `self_attn.o_proj` (GPT-OSS-20B). These choices follow the published WISE/GRACE convention of editing a late MLP `down_proj` block ($\approx n_{\text{layers}} - 3$ to $n_{\text{layers}} - 8$). ROME and AlphaEdit are excluded from this alignment because their original methodology prescribes different layer locations (causal-tracing layer 5 for ROME; the rewrite window $\{4, \ldots, 8\}$ for AlphaEdit) and changing them would deviate from those papers.

Hardware. Experiments run on RTX 5090 in bf16, with GPT-OSS-20B run on RTX Pro 6000 WS, and QWEN-SEA-LION-32B on H200.

E.4.4 50K ZsRE Stress-Test Construction

The 50K stress test in Figure 1 is drawn from a pool of 75,658 ZsRE-style samples aggregated from the standard ZsRE train split released by Mitchell et al. (2021) together with additional entries drawn from the original ZsRE source corpus (Levy et al., 2017) that do not overlap with the standard split, yielding a pool large enough to support a stream of 50K non-overlapping edits. Each pool entry carries (i) an original prompt, (ii) a target answer, (iii) a paraphrased query, and (iv) a locality query, so the three evaluation sets ($\mathcal{D}_{\text{edit}}, \mathcal{D}_{\text{rephrase}}, \mathcal{D}_{\text{locality}}$) are aligned one-to-one with each edit, matching the Section 2.1 formulation. For the stress test we sample 50K entries from this pool (with a fixed seed) and apply them as a single uninterrupted sequential stream on LLaMA-3.1-8B, chosen as a representative 8B-scale model; results on other backbones at smaller scales are consistent (Table 2). Evaluation is carried out at checkpoints $N \in \{1\text{K}, 10\text{K}, 15\text{K}, 20\text{K}, 30\text{K}, 40\text{K}, 50\text{K}\}$ over the matching prefix of each dataset.

E.4.5 Portability Guidelines

Model-specific parameters are limited to two choices: (1) the matching threshold c and (2) the token pooling ratio. Both are selected once per model and then held fixed across datasets and edit scales. The resulting values (Table 3) cluster tightly within each family while shifting cleanly between families, indicating that they reflect genuine embedding-geometry differences rather than dataset- or benchmark-specific tuning.

Concretely, the calibrated thresholds fall into three distinct, family-coherent bands:

- **LLaMA-3 / 3.1 family** ($c \in [0.80, 0.85]$, span 0.05): LLaMA-3-8B-Instruct uses 0.80, while LLaMA-3.1-8B-Instruct and DeepSeek-R1-Distill-Llama-8B both use 0.85. The fact that DeepSeek's distilled LLaMA inherits the LLaMA-3.1 threshold despite having undergone substantial reasoning-oriented post-training is direct evidence that c tracks the underlying representation geometry of the base model rather than the downstream task.

- **Qwen-2.5 / 3 family** ($c \in [0.55, 0.60]$, span 0.05): Qwen2.5-7B-Instruct and the 32B Qwen-SEA-LION-v4 both use 0.55, and DeepSeek-R1-Distill-Qwen-1.5B uses 0.60 – again, the distilled variant remains within the family band. The consistent ~ 0.25 gap from the LLaMA band reflects the smaller cosine spread of Qwen hidden states at the edited layer, not a per-experiment correction.

- **GPT-OSS** ($c = 0.80$): the only model in its family, sitting in the LLaMA range, consistent with its larger residual stream norms.

Pooling ratio is even more stable: 60% for every model except LLaMA-3-8B-Instruct (50%), and chosen by a single ablation (Figure 11) rather than re-tuned per benchmark or scale.

This level of calibration is consistent with existing methods: GRACE requires per-model ε, WISE requires per-model importance weights, and REPAIR requires per-model cross-layer settings – in each case the per-model knob also reflects representation-level properties rather than dataset-specific tuning. For a new model we recommend: (i) identify the family of the base model and initialize c to the family's central value (0.85 for LLaMA-derived models, 0.55 for Qwen-derived models); (ii) run the token-selection ablation at moderate scale (N=100) to confirm the 60% pooling default; and (iii) if needed, refine c with a short sweep over $\{0.55, 0.60, 0.80, 0.85\}$ on 30 edits, anchored at the family-default value.

F Additional ZsRE Results

F.1 Routing-Oriented Baselines at 1K Edits

Table 4 directly tests the closest routing-oriented alternatives. At the clearly delimited N=1000 scale, HoReN attains the highest OP on all three backbones. The result supports a superiority of the overall performance of the HoReN model.

Table 4: Comparisons with learned-retriever/codebook-router editors on ZsRE at exactly N=1000 sequential edits. LLaMA-3-8B and Qwen2.5-7B are the N=1000 results are reported by the REPAIR paper.

Model	Method / protocol	Rel.	Gen.	Loc.	OP
LLaMA-3.1-8B	**HoReN**	**0.994**	**0.926**	0.993	**0.970**
	RECIPE	0.496	0.463	**0.999**	0.612
	LOKA	0.850	0.464	**1.000**	0.733
	REPAIR	0.228	0.185	0.614	0.296
LLaMA-3-8B	**HoReN**	**0.994**	**0.836**	0.996	**0.939**
	RECIPE	0.355	0.343	0.946	0.487
	LOKA	0.787	0.453	**1.000**	0.709
	REPAIR	0.680	0.650	0.890	0.730
Qwen2.5-7B	**HoReN**	**0.996**	**0.931**	0.984	**0.970**
	RECIPE	0.353	0.333	0.960	0.483
	LOKA	0.866	0.462	**1.000**	0.737
	REPAIR	0.720	0.700	0.670	0.690

F.2 LLaMA-3.1-8B Large-Scale Stability

Table 5 reports full LLaMA-3.1-8B results at N=2000, 5000, and 10000 edits, including AlphaEdit and UltraEdit which are omitted from Table 2 due to prohibitive runtime on other models. AlphaEdit collapses between N=2000 and N=5000 (OP drops from 0.82 to 0.10) due to near-zero locality; UltraEdit degrades more gradually (OP from 0.73 to 0.67). HoReN maintains OP above 0.95 throughout all scales.

Table 5: LLaMA-3.1-8B results at large scale on ZsRE (N=2000, 5000, 10000). AlphaEdit and UltraEdit are included here to document their scaling behavior; they are excluded from Table 2 due to prohibitive runtime on other models.

Method	$N = 2000$				$N = 5000$				$N = 10000$			
	Rel.	**Gen.**	**Loc.**	**OP**	**Rel.**	**Gen.**	**Loc.**	**OP**	**Rel.**	**Gen.**	**Loc.**	**OP**
GRACE	0.99	0.05	1.00	0.37	0.99	0.05	1.00	0.37	0.99	0.06	1.00	0.38
WISE	0.49	0.47	0.99	0.61	0.40	0.37	0.98	0.52	0.37	0.36	0.98	0.51
AlphaEdit	0.95	0.88	0.65	0.82	0.18	0.15	0.03	0.10	0.21	0.17	0.02	0.08
UltraEdit	0.87	0.85	0.53	0.73	0.86	0.85	0.45	0.69	0.86	0.84	0.42	0.67
HoReN	**0.99**	**0.94**	**0.98**	**0.97**	**0.99**	**0.92**	**0.96**	**0.96**	**0.99**	**0.92**	**0.96**	**0.96**

Figure 5 tracks the full sequential trajectory up to 10K edits: HoReN's generalization stays flat while GRACE and WISE decay monotonically, and HoReN retains 99% accuracy on early edits after the full 10K sequence with no catastrophic forgetting.

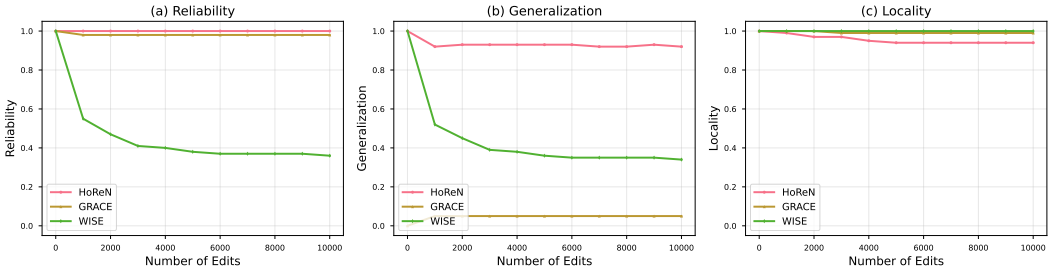


Figure 5: Stability of editing performance over sequential edits on ZsRE with LLaMA-3.1-8B (up to 10K edits). HoReN maintains stable generalization while GRACE and WISE degrade.

Figures 6–8 decompose the OP curves from Figure 1 into the three constituent metrics for all methods on ZsRE (LLaMA-3.1-8B). HoReN maintains Reliability above 0.99 and Generalization above 0.91 throughout 50K edits, while Locality declines gradually from 1.00 to 0.89, which is the primary source of the slight OP drop. GRACE sustains near-perfect Reliability and Locality but has effectively zero Generalization at all scales, confirming the routing bottleneck rather than a weight-editing limitation. AlphaEdit's Locality collapse between 2K and 5K drives its OP cliff.

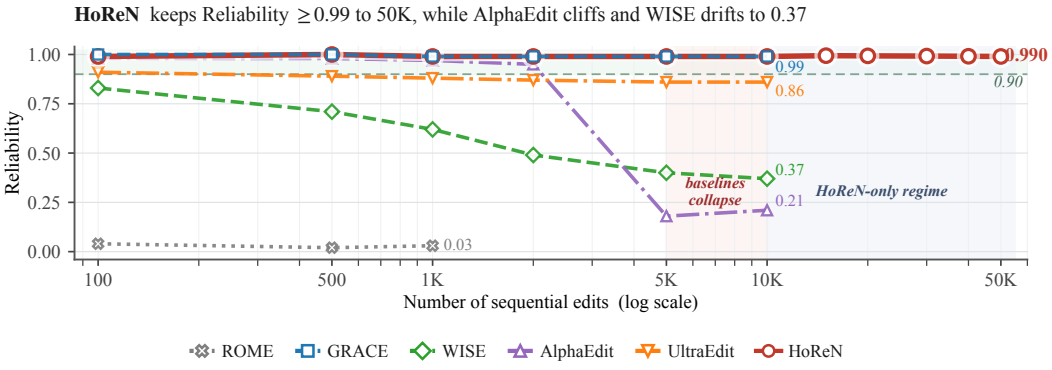


Figure 6: Reliability scaling to 50K sequential edits on ZsRE (LLaMA-3.1-8B).

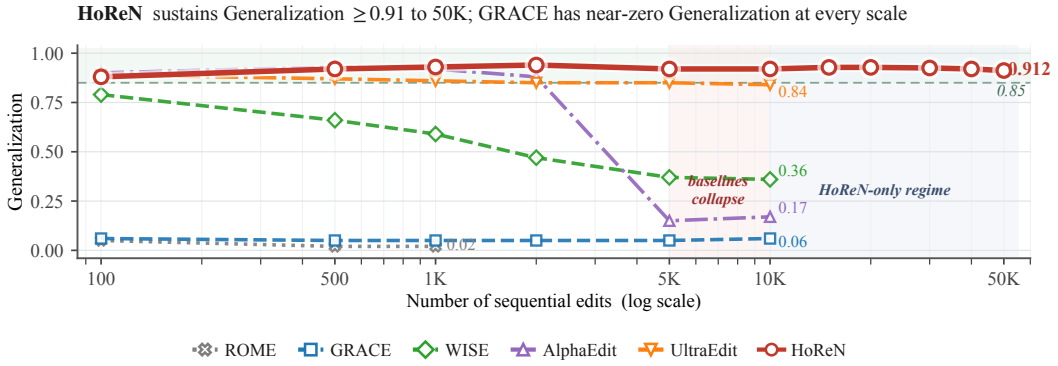


Figure 7: Generalization scaling to 50K sequential edits on ZsRE (LLaMA-3.1-8B).

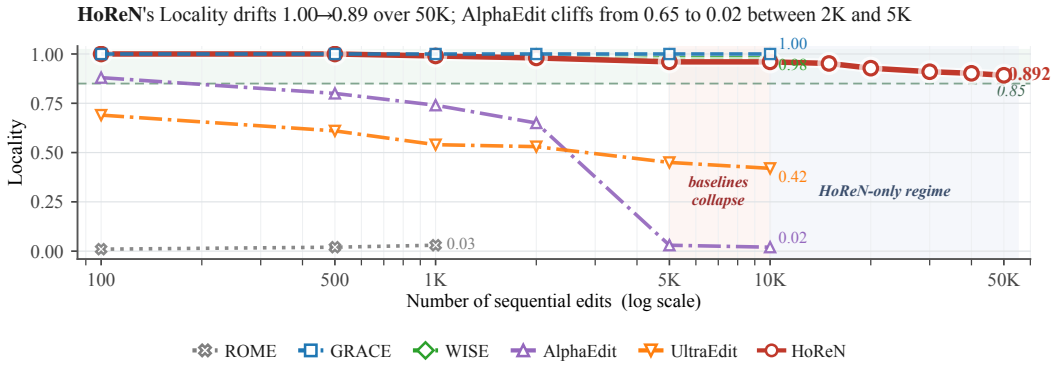


Figure 8: Locality scaling to 50K sequential edits on ZsRE (LLaMA-3.1-8B).

F.3 Efficiency: Theory, Scaling, and Trade-offs

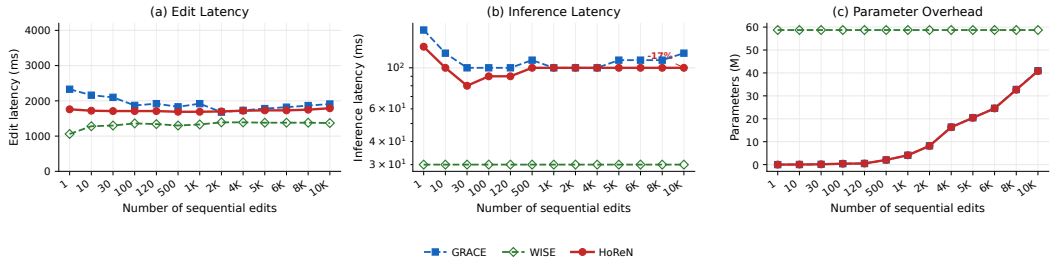


Figure 9: Efficiency scaling on ZsRE (LLaMA-3.1-8B) over $N \in [1, 10\text{K}]$ sequential edits: **(a)** per-edit latency, **(b)** per-query inference latency, and **(c)** cumulative trainable parameters of the side-memory overhead.

Theoretical complexity. When using direct tensor value as the codebook value adaptor, HoReN inherits GRACE's $O(Kd)$ retrieval cost over a codebook of size K in dimension d (Hartvigsen et al., 2023), and adds a constant number of damped Hopfield refinement steps (softmax + matmul + normalization) per query. This refinement process is $O(Kd)$ per step and is performed only for a constant number of times, so HoReN's asymptotic cost is the same as GRACE's, which is confirmed by Figure 9.

Editing efficiency. On LLaMA-3.1-8B model, HoReN, GRACE, and WISE have similar editing time at around 1700 ms, but on Qwen-SEA-LION-v4-32B (Table 6), the gap widens substantially:

HoReN edits in 4.85s versus WISE at 17.37s ($3.6\times$ faster) and AlphaEdit at 22.83s ($4.7\times$ faster). Parameter-modifying baselines pay a steep activation-statistics / projection cost that scales with model width, while HoReN's per-edit cost is dominated by the small value adaptor and is largely insensitive to backbone size.

Inference-vs-memory trade-off. WISE trades memory for latency in the opposite direction from HoReN: it pays a fixed $+58.72$M up front but routes inference through a small module (30ms). HoReN starts at $+4.09$M (1K edits) and grows by ~ 4.08M per additional 1K edits, so its side-memory only catches up to WISE's at roughly $N \approx 14$–15K edits, by which point WISE has already lost ≥ 0.4 on Generalization in our 50K-scaling study (Figure 1). For the editing regimes we evaluate, HoReN therefore Pareto-dominates WISE on the (memory, generalization) frontier while remaining within $\sim 5\%$ inference latency of GRACE.

Table 6: Efficiency at $N=1000$ edits on ZsRE. "Side mem." counts the editor's auxiliary parameters (codebook for HoReN/GRACE; dual-memory for WISE); "Δ Model wts." counts changes to the backbone (AlphaEdit modifies the model in place, hence 0 side-memory). Following the GRACE convention (Hartvigsen et al., 2023), side-memory excludes per-edit *key* storage, which would roughly double the figure for HoReN/GRACE; AlphaEdit has no key storage, and WISE's reported value already includes its routing memory.

Model	Method	Edit (ms)	Infer. (ms)	Side mem. (M)	Δ Model wts. (M)
LLaMA-3.1 8B	WISE	1670	30	$+58.72$	0
	GRACE	1690	100	$+4.10$	0
	AlphaEdit	3990	30	0	$+293.60$
	HoReN	**1670**	**100**	$\mathbf{+4.09}$	$\mathbf{0}$
SEA-LION 32B	WISE	17370	80	$+131.07$	0
	GRACE	4800	330	$+5.12$	0
	AlphaEdit	22830	90	0	$+655.36$
	HoReN	**4850**	**330**	$\mathbf{+5.09}$	$\mathbf{0}$

G Cross-Dataset Generalization

G.1 Full Numerical Results

Tables 7–9 report the complete numerical results underlying Figure 3 in Section 3.3.1.

Table 7: Sequential editing on WikiBigEdit (LLaMA-3.1-8B). AlphaEdit's locality degrades sharply under long edit sequences while HoReN remains stable.

#Edits	Method	Rel.	Gen.	Loc.
500	AlphaEdit	0.825	0.605	0.267
	HoReN	**0.992**	**0.767**	**0.682**
1000	AlphaEdit	0.797	0.618	0.194
	HoReN	**0.987**	**0.753**	**0.654**
3000	AlphaEdit	0.799	0.651	0.169
	HoReN	**0.986**	**0.754**	**0.631**

Table 8: Unstructured knowledge editing on UnKE (LLaMA-3.1-8B) broken down by question type. Rows: three UnKEBench question types (*Original*, *Paraphrase*, *Sub-question*) crossed with all applicable metrics. Columns: edit counts $N \in \{10, 100, 500\}$, each split between UnKE and HoReN. Best per (row, N) in bold; BLEU and BERTScore are not reported for the Sub-question split in the original UnKEBench protocol (Deng et al., 2024).

		N=10		N=100		N=500	
Q-type	**Metric**	UnKE	HoReN	UnKE	HoReN	UnKE	HoReN
Original	BLEU	**0.191**	0.143	**0.202**	0.126	0.098	**0.117**
	ROUGE-1	0.453	**0.583**	0.432	**0.457**	0.222	**0.404**
	ROUGE-2	0.240	**0.395**	0.196	**0.219**	0.054	**0.184**
	ROUGE-L	0.431	**0.551**	0.405	**0.428**	0.205	**0.380**
	BERTScore	0.713	**0.814**	0.698	**0.716**	0.183	**0.682**
Paraphrase	BLEU	**0.204**	0.126	**0.199**	0.115	0.097	**0.106**
	ROUGE-1	0.452	**0.476**	**0.418**	0.402	0.215	**0.360**
	ROUGE-2	0.211	**0.278**	**0.178**	0.165	0.047	**0.142**
	ROUGE-L	0.443	**0.452**	**0.392**	0.374	0.197	**0.338**
	BERTScore	**0.756**	0.707	0.676	**0.685**	0.166	**0.659**
Sub-question	ROUGE-1	**0.469**	0.424	0.309	**0.373**	0.141	**0.338**
	ROUGE-2	**0.176**	0.168	0.071	**0.123**	0.016	**0.100**
	ROUGE-L	**0.459**	0.417	0.303	**0.364**	0.137	**0.332**

Column / row semantics (UnKEBench (Deng et al., 2024)). For each unstructured edit text A, UnKEBench provides three question types probing complementary aspects of the edited model's behavior. The **Original** block measures *reliability*: whether the edit is faithfully recalled on the original question Q. The **Paraphrase** block measures *generalization*: whether the edit transfers to a semantically equivalent rewording Q_p of Q. The **Sub-question** block measures *factual coverage*: whether individual knowledge entities within the passage are correctly installed. **BLEU** and **ROUGE-1/2/L** are word-level n-gram overlap scores; **BERTScore** captures semantic similarity beyond lexical match. BLEU and BERTScore are not reported for the Sub-question split in the original UnKEBench protocol.

Table 9: Cross-dataset generalization on WikiBigEdit and ZsRE at N=1000 edits (DeepSeek-R1-Distill-Qwen-1.5B). Gray superscripts show HoReN's absolute improvements over each baseline.

Method	**WikiBigEdit**		**ZsRE**	
	Rel.	**Gen.**	**Rel.**	**Gen.**
GRACE	$0.63^{+0.33}$	$0.07^{+0.72}$	$0.83^{+0.11}$	$0.05^{+0.74}$
WISE	$0.47^{+0.49}$	$0.38^{+0.41}$	$0.49^{+0.45}$	$0.47^{+0.32}$
REPAIR	$0.58^{+0.38}$	$0.54^{+0.25}$	$0.59^{+0.35}$	$0.57^{+0.22}$
HoReN	**0.96**	**0.79**	**0.94**	**0.79**

H Robustness and Routing Diagnostics

All experiments in this section use ZsRE and LLaMA-3.1-8B at exactly N=1000 sequential edits unless a different scale is stated. This labeling separates the robustness evidence from the single-stream 50K stress test in Figure 1.

H.1 Seed and Edit-Order Robustness

Table 10 reports five independent editing streams. HoReN has low variance (OP standard deviation 0.003) and is more stable than AlphaEdit in locality and OP. Table 11 changes only the order of the same edit set; the small OP change ($0.971 \rightarrow 0.964$) indicates that the N=1000 result is not tied to one favorable ordering.

Table 10: Five-seed robustness on ZsRE at N=1000 (LLaMA-3.1-8B); entries are mean $\pm$ standard deviation across editing streams.

Method	Rel.	Gen.	Loc.	OP
HoReN	0.993 ± 0.001	0.906 ± 0.009	0.989 ± 0.003	0.962 ± 0.003
AlphaEdit	0.974 ± 0.006	0.920 ± 0.013	0.684 ± 0.026	0.849 ± 0.008

Table 11: Edit-order sensitivity on the same ZsRE edit set at N=1000 (LLaMA-3.1-8B). "Random" denotes one shuffled permutation of the default stream.

Order	Rel.	Gen.	Loc.	OP
Default	0.994	0.926	0.995	0.971
Random permutation	0.994	0.915	0.985	0.964

H.2 Damping and Step Sensitivity

Table 12 gives two controlled slices through the (γ, M) space with β=20: a γ sweep at M=1 and an M sweep at γ=0.1. The useful region is not a single numerical point: $\gamma \in \{0.05, 0.10\}$ gives OP 0.968–0.970, and $M \in \{1, 2\}$ gives OP 0.955–0.970. However, stronger attraction consistently sacrifices locality; at γ=0.5 locality falls to 0.096, and at M=4 it falls to 0.521. Thus $(\gamma, M) = (0.1, 1)$ is a stable high-performing operating point within the tested neighborhood, not a claim of global optimality.

Table 12: Sensitivity of damping γ and refinement steps M on ZsRE at N=1000 (LLaMA-3.1-8B, β=20). The left block varies γ with M=1; the right block varies M with γ=0.1.

γ	Rel.	Gen.	Loc.	OP	M	Rel.	Gen.	Loc.	OP
0.05	0.996	0.911	0.999	0.968	1	0.994	0.926	0.993	0.970
0.10	**0.994**	**0.926**	**0.993**	**0.970**	2	0.994	0.928	0.943	0.955
0.20	0.995	0.925	0.928	0.949	4	0.995	0.930	0.521	0.784
0.50	0.995	0.883	0.096	0.438					
1.00	0.615	0.135	0.074	0.183					

H.3 False-Positive Retrievals

We define *Loc-retr* as the fraction of unrelated locality queries that activate any edit-memory entry. Table 13 shows that false activations remain below 0.6% through N=1000, providing a routing-level explanation for the high locality scores. Loc-retr is a mechanistic diagnostic rather than a substitute for output-level locality, and the increasing trend also motivates denser-collision evaluations at larger scales.

Table 13: False-positive edit-memory activation on unrelated ZsRE locality queries (LLaMA-3.1-8B).

Number of edits N	100	500	1000
Loc-retr	0.000	0.002	0.006

I Analysis and Ablation Details

I.1 Routing Ingredients: GRACE → GRACE+Norm. → HoReN

Table 14 isolates the contribution of each routing ingredient. We fix the prompt-token pooling strategy across methods: queries are obtained by averaging the hidden states of the last 60% of prompt tokens for both GRACE and HoReN. This controls for pooling choice, which would otherwise act as a confounding factor. Table 15 confirms that applying Hopfield updates *without* normalization collapses locality, motivating the joint design.

Table 14: Routing ingredient ablation: unnormalized GRACE vs. GRACE+Norm. vs. HoReN (ZsRE, LLaMA-3.1-8B).

N	Method	Rel.	Gen.	Loc.	OP
$N = 100$	**HoReN**	**0.99**	**0.88**	**1.00**	**0.96**
	GRACE+Norm.	1.0	0.41	1.0	0.75
	GRACE	1.0	0.06	1.0	0.40
$N = 500$	**HoReN**	**1.00**	**0.92**	**1.00**	**0.97**
	GRACE+Norm.	1.0	0.37	1.0	0.72
	GRACE	1.0	0.06	1.0	0.39
$N = 1000$	**HoReN**	**0.99**	**0.93**	**0.99**	**0.97**
	GRACE+Norm.	1.0	0.33	1.0	0.69
	GRACE	1.0	0.07	1.0	0.41

Table 15: Effect of normalization on Hopfield retrieval (ZsRE, LLaMA-3.1-8B). Without normalization, the Hopfield step collapses locality.

N	Method	Rel.	Gen.	Loc.	OP
$N = 100$	**Normalized**	**0.99**	**0.88**	**1.00**	**0.96**
	Unnormalized	1.00	0.92	0.08	0.43
$N = 500$	**Normalized**	**1.00**	**0.92**	**1.00**	**0.97**
	Unnormalized	0.98	0.79	0.07	0.38
$N = 1000$	**Normalized**	**0.99**	**0.93**	**0.99**	**0.97**
	Unnormalized	0.96	0.75	0.07	0.37

I.2 Hopfield Iteration Steps

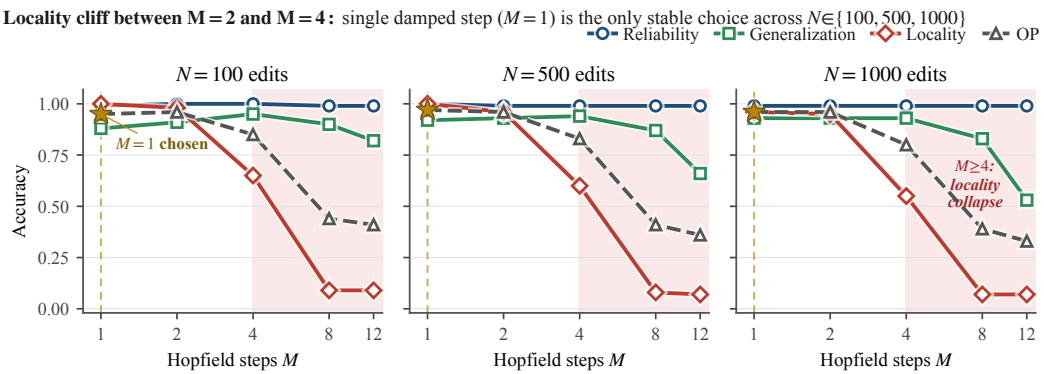


Figure 10: Effect of Hopfield refinement steps M on the four editing metrics across edit counts $N \in \{100, 500, 1000\}$ (LLaMA-3.1-8B, ZsRE). Reliability is essentially flat in M; Generalization peaks at $M \in [1, 4]$; Locality cliffs from $\sim$0.95 to $\sim$0.6 between M=2 and M=4 and bottoms out near 0.07 for $M \geq 8$, dragging OP with it. The single damped step (M=1, gold star) is the only value of M that keeps all three metrics jointly stable across all three scales.

I.3 Value Adaptor: Direct Vector vs. LoRA

Direct-vector and LoRA value adaptors achieve identical editing performance; the direct variant is preferred for its $8\times$ lower parameter cost ($+4.09$M parameters versus $+32.64$M at N=1000). Table 16 compares the two across $N \in \{100, 500, 1000\}$.

Both variants reach OP 0.95–0.98 at every scale, with differences of at most 0.01. The equivalence is expected: HoReN's retrieval accuracy depends entirely on the key-matching mechanism (normalization + Hopfield step), not on the capacity of the value payload. Once the correct edit is retrieved, either adaptor has sufficient expressivity to represent the target answer at these scales. The direct variant adds $+4.09$M parameters at N=1000 versus $+32.64$M for LoRA, an $8\times$ gap with no

performance benefit. We therefore recommend direct-vector as the default; LoRA remains available for deployments requiring strict parameter isolation between edits (e.g., federated or multi-tenant settings).

Table 16: Direct value vs. LoRA (r=4) on ZsRE, LLaMA-3.1-8B. Performance is identical; direct variant adds 8× fewer parameters.

N	Method	Rel.	Gen.	Loc.	OP
$N = 100$	Value	0.99	0.88	1.00	0.96
	LoRA	0.99	0.87	1.00	0.95
$N = 500$	Value	1.00	0.92	1.00	0.97
	LoRA	1.00	0.92	1.00	0.97
$N = 1000$	Value	0.99	0.93	0.99	0.97
	LoRA	1.00	0.93	1.00	0.98

I.4 Key/Query Representation: Token Pooling Ratio

The last 60% suffix pooling ratio is Pareto-optimal across all scales; performance is non-monotonic, with both short and long suffixes introducing distinct failure modes. Figure 11 sweeps ten ratios from 10% to 100% on LLaMA-3.1-8B, ZsRE, at $N \in \{100, 500, 1000\}$.

Short suffixes (≤30%) — generalization failure. Very short suffixes pool too few tokens to capture the semantic content of the prompt, producing a query representation that reflects surface position rather than meaning. Gen. stays at 0.47–0.67 at N=1000 regardless of N: the key constructed from the original prompt's last few tokens is insufficiently similar to the key from the paraphrase's last few tokens, so the Hopfield step cannot bridge the residual angular gap.

Long suffixes (≥80%) — locality failure. Very long suffixes include tokens specific to the prompt's preamble or instruction format, making the query representation sensitive to surface-level prompt differences rather than semantic content. This causes locality collapse: at N=1000, Loc. drops to 0.74–0.84 as unrelated locality queries, which share similar preambles with edit queries, begin triggering false matches.

Last 60% — Pareto-optimal. The 60% suffix captures enough semantic context for robust paraphrase matching (Gen. 0.88–0.93) while excluding enough of the prompt-specific prefix to keep locality queries well-separated (Loc. 0.99–1.00). The 50–70% range all reach OP ≥0.95, indicating the choice is not sensitive to exact calibration. We treat the 60% default as an empirical heuristic; learned token selection (e.g., attention-weighted pooling) is a concrete future direction.

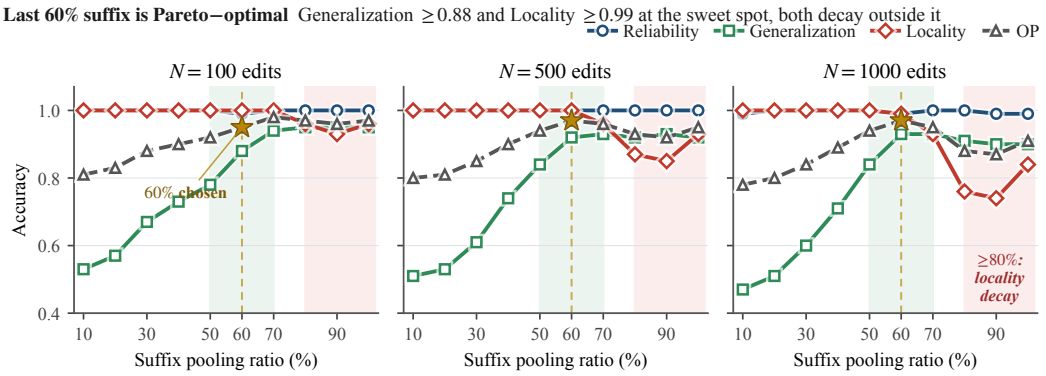


Figure 11: Effect of suffix pooling ratio on the four editing metrics across $N \in \{100, 500, 1000\}$ (LLaMA-3.1-8B, ZsRE). Reliability is essentially flat at 1.00; Generalization grows monotonically from ∼0.50 at 10% to ∼0.93 from 60% onward; Locality is held at 1.00 up to 60% then decays as longer suffixes start incorporating prompt-specific tokens, dropping to 0.74−0.84 at ≥ 80% for N=1000. The 50% − 70% Pareto band (green) all reach OP ≥ 0.94; the ≥ 80% locality-decay zone (red) cuts OP. The chosen 60% setting (gold star) is the only ratio that simultaneously achieves Generalization ≥ 0.88 and Locality ≥ 0.99 at every scale.