Lifelong Model Editing with Graph-Based External Memory

Yash Kumar Atri
University of Virginia
atri@virginia.edu

Ahmed Alaa
UC Berkeley & UCSF
amalaa@berkeley.edu

Tom Hartvigsen
University of Virginia
hartvigsen@virginia.edu

Abstract

Large language models (LLMs) have revolutionized natural language processing, yet their practical utility is often limited by persistent issues of hallucinations and outdated parametric knowledge. Although post-training model editing offers a pathway for dynamic updates, existing methods frequently suffer from overfitting and catastrophic forgetting. To tackle these challenges, we propose a novel framework that leverages hyperbolic geometry and graph neural networks for precise and stable model edits. We introduce HYPE[1], which comprises three key components: (i) Hyperbolic Graph Construction, which uses Poincaré embeddings to represent knowledge triples in hyperbolic space, preserving hierarchical relationships and preventing unintended side effects by ensuring that edits to parent concepts do not inadvertently affect child concepts; (ii) Möbius-Transformed Updates, which apply hyperbolic addition to propagate edits while maintaining structural consistency within the hyperbolic manifold, unlike conventional Euclidean updates that distort relational distances; and (iii) Dual Stabilization, which combines gradient masking and periodic GNN parameter resetting to prevent catastrophic forgetting by focusing updates on critical parameters and preserving long-term knowledge. Experiments on CounterFact, CounterFact+, and MQuAKE with GPT-J and GPT2-XL demonstrate that HYPE significantly enhances edit stability, factual accuracy, and multi-hop reasoning.

1 Introduction

Large language models (LLMs) such as GPT-3 (Brown et al., 2020), PaLM (Chowdhery et al., 2023), and LLaMA (Touvron et al., 2023) have revolutionized natural language processing (NLP), enabling unprecedented capabilities in text generation (Li et al., 2024a), reasoning (Chowdhery et al., 2023), and contextual understanding (Zhao et al., 2024). These models underpin a wide range of applications, from conversational agents (Liu et al., 2024) to knowledge-intensive tasks like question answering (Phukan et al., 2024; Shah et al., 2024; Gao et al., 2024) and summarization (Atri et al., 2023a,b; Dey et al., 2020; Atri et al., 2023c). However, their reliance on static, pre-trained parametric knowledge renders them prone to generating factual inaccuracies (Wang et al., 2024), hallucinations (Ji et al., 2023), and outdated information—critical limitations in real-world deployments where accuracy and timeliness are paramount (Lazaridou et al., 2021; Atri et al., 2025). While fine-tuning on updated data can mitigate these issues, the computational and data demands of retraining billion-parameter models (Kaplan et al., 2020) render this approach impractical for real-time knowledge updates. Instead, post-training model editing has emerged as a promising alternative, enabling targeted modifications to a model's parametric knowledge without full retraining (Mitchell et al., 2022a).

Existing model editing methods, such as ROME (Meng et al., 2022a) and MEMIT (Meng et al., 2022b), rely on Euclidean geometry to modify specific facts. While these methods achieve localized edits, they struggle with hierarchical relationships due to Euclidean space's flat structure. For example, ROME enforces memorization via equality constraints, while MEMIT uses least-squares optimization, both of which fail to preserve relational depth (Yang et al., 2024). This leads to *geometric mismatch*, where edits distort semantically related knowledge (Nickel and Kiela, 2017). Additionally, Euclidean updates in overparameterized models like GPT-2 (Radford et al., 2019a) cause *update instability*, propagating unintended changes through dense connections (Kaplan et al., 2020). Finally, these methods suffer from *contextual fragility*, lacking mechanisms to maintain global coherence during edits (Zhong et al., 2023).

[1] HYPE (**HY**perbolic **P**arameter **E**diting)

Findings of the Association for Computational Linguistics: ACL 2025, pages 13336–13352

Hyperbolic geometry, in contrast, naturally encodes hierarchical relationships through its exponential growth property, making it well-suited for modeling linguistic knowledge (Nickel and Kiela, 2017). Unlike Euclidean spaces, where distances are linear, hyperbolic space expands exponentially, allowing child nodes to be placed far from parent nodes while preserving their relational proximity (Ganea et al., 2018). This property enables precise parameter updates that do not propagate unintended side effects. For instance, modifying a parent concept in hyperbolic space affects only its immediate vicinity, preserving the stability of its hierarchical descendants. Furthermore, hyperbolic operations like Möbius addition (Ungar, 2013) ensure updates remain on the manifold, avoiding the geometric distortion caused by Euclidean vector addition.

Motivated by these insights and the inherent advantages of hyperbolic geometry, we propose **HYPE** (**HY**perbolic **P**arameter **E**diting), a novel framework that leverages hyperbolic geometry (Ganea et al., 2018) for model editing through three core components. First, Hyperbolic Graph Construction projects knowledge graph into a learnable curvature space using Poincaré embeddings (Nickel and Kiela, 2017), preserving hierarchical relationships—such as parent-child and whole-part associations—via exponential mapping and hyperbolic distances. Second, Möbius-Transformed (Ungar, 2013) Updates adjust model parameters using hyperbolic addition, a geometric operation that preserves the hierarchical structure of the data. Unlike Euclidean vector addition, which can distort relationships in hierarchical data, Möbius addition ensures that edits remain within the hyperbolic manifold. This prevents unintended side effects by maintaining the relative distances between related concepts. Third, Dual Stabilization combines gradient-based sparsification (Frankle and Carbin, 2019), which masks negligible gradient updates, with periodic resetting of graph network parameters to prevent the overwriting of critical pre-existing knowledge during the integration of new edits. By unifying hyperbolic embeddings, Möbius-Transformed updates, and robust stabilization techniques, HYPE achieves precise and resilient model edits.

We summarize our contributions as follows:

1. HYPE is the first approach to integrate hyperbolic geometry with graph-based model editing, leveraging Poincaré embeddings to project knowledge graphs into a learnable curvature space that preserves complex hierarchical relationships during model updates.
2. HYPE introduces a novel editing mechanism that applies Möbius transformations for parameter updates, ensuring that edits remain consistent with the hyperbolic manifold and preserving the inherent hierarchical structure—thereby preventing the distortions common with Euclidean vector addition.
3. We validate HYPE across three widely benchmarked datasets—CounterFact (Meng et al., 2022a), CounterFact+ (Yao et al., 2023b), and MQuAKE (Zhong et al., 2023)—and two popular LLMs—GPT-J (Wang and Komatsuzaki, 2021), and GPT2-XL (Radford et al., 2019b)—demonstrating superior factual accuracy, parameter efficiency, and edit stability compared to existing model editing methods.

Code available at: https://github.com/yashkumaratri/HYPE

2 Related Work

Post-Training Model Editing: Model editing techniques broadly fall into parameter-modifying and parameter-preserving approaches. Parameter-modifying methods, such as locate-then-edit strategies (Meng et al., 2022a,b; Hartvigsen et al., 2023; Kolbeinsson et al., 2025) and meta-learning frameworks (Mitchell et al., 2022a; Cao et al., 2021), adjust model parameters via hypernetworks or rank-one weight updates. Zhang et al. (2024) enhances specificity using knowledge graphs, but these methods typically treat parameter changes as isolated scalar operations (Ma et al., 2024; Gu et al., 2024), disregarding structured knowledge representations in LLMs (Cohen et al., 2023). This often leads to unintended side effects (Yao et al., 2023a; Kolbeinsson et al., 2025) and poor support for compositional updates (Hase and Bansal, 2023).

Parameter-preserving approaches avoid direct weight updates by leveraging external memories or prompts (Mitchell et al., 2022b; Huang et al., 2023; Madaan et al., 2022; Zheng et al., 2023), while continual learning methods (Zeng et al., 2019; Farajtabar et al., 2020) use orthogonal projections to mitigate catastrophic forgetting. However, these strategies require architectural modifications or fail to maintain long-term coherence in factual updates.

Graph-Based Model Editing: Knowledge graphs provide a structured approach to model editing by aligning factual updates with relational dependen-

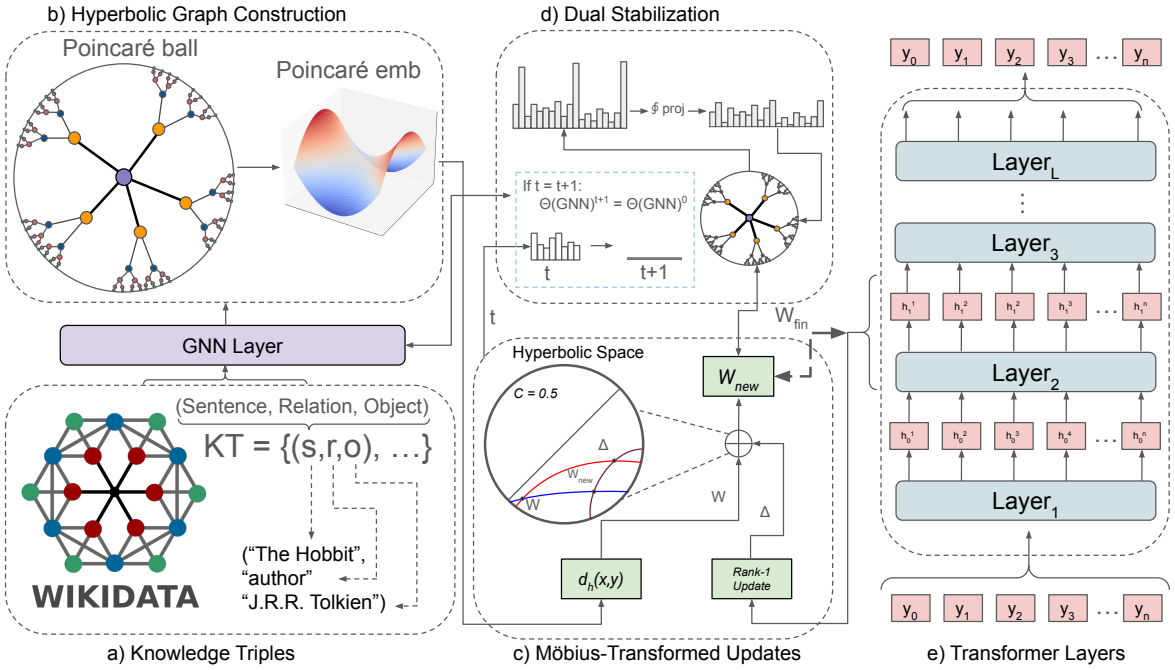

Figure 1: The illustration delineates our proposed model, HYPE. We begin by constructing a hyperbolic knowledge graph (b) using Poincaré embeddings to encode hierarchical relationships. When an edit is required, we apply Möbius transformations (c) to update the weights while ensuring curvature-aware consistency. To maintain stability, Dual stabilization strategy (d) removes transient or spurious updates. The edited knowledge is then integrated into the model (e), preserving factual accuracy and structural integrity.

cies. Cohen et al. (2023) aligns LLM attention with Wikidata triples, while Cao et al. (2021) edits GNNs through subgraph swaps. Zhang et al. (2024) integrates knowledge graphs into transformer layers but relies on Euclidean embeddings, distorting hierarchical relationships. Similarly, Qin et al. (2023) models higher-order dependencies via hypergraphs but lacks geometric constraints to maintain relational consistency.

Hyperbolic Model Editing: Hyperbolic embeddings have proven effective for encoding hierarchical knowledge (Chami et al., 2019), particularly in NLP (Valentino et al., 2024) and knowledge graph applications (Chami et al., 2020). Chen et al. (2022) introduces hyperbolic attention for LLMs requiring full retraining, while Cohen et al. (2023) links LLMs to graphs using Euclidean projections, which fails to capture hyperbolic curvature.

HYPE is the first framework to unify hyperbolic embeddings and Möbius updates for model editing. Unlike Euclidean methods (e.g., ROME (Meng et al., 2022b), MEMIT (Meng et al., 2022b), GLAME (Zhang et al., 2024)), which struggle with hierarchical edits and stability, HYPE leverages hyperbolic geometry to preserve relational consistency. Compared to parameter-preserving approaches, it avoids architectural overhead while maintaining edit precision. Experiments show HYPE achieves +9.12 higher Edit Quality Score (EDS) and +3.59 better efficacy than state-of-the-art methods (Table 1), demonstrating the benefits of curvature-aware updates.

3 Methodology

In this section, we detail the methodology underlying HYPE, our proposed framework for hyperbolic model editing.

3.1 Hyperbolic Graph Construction

We first initialize a knowledge graph using Wikidata (Vrandečić and Krötzsch, 2014) triples, which consist of subject-relation-object tuples $\{(s, r, o) \ldots\}$. These triples encode factual knowledge, where s represents an entity (e.g., "Albert Einstein"), o represents another entity or concept associated with s (e.g., "Theory of Relativity"), and r defines the semantic relationship between them (e.g., "contributed to"). To construct meaningful representations, we first obtain Euclidean embeddings using a graph neural network (GNN), capturing relational and structural dependencies. For any input string x (which may represent an en-

tity or a relation), let $\mathbf{v}_{\text{euc}}(x)$ denote its Euclidean embedding. We then project these embeddings into hyperbolic space via the exponential map on the Poincaré ball (Nickel and Kiela, 2017)

$$\begin{aligned}\mathbf{v}_{\text{hyp}}(x) &= \exp_0^c\left(\mathbf{v}_{\text{euc}}(x)\right) \\ &= \tanh\left(\sqrt{c}\|\mathbf{v}_{\text{euc}}(x)\|\right) \cdot \frac{\mathbf{v}_{\text{euc}}(x)}{\sqrt{c}\|\mathbf{v}_{\text{euc}}(x)\|}\end{aligned} \tag{1}$$

where c is the curvature parameter (set to 1.0 in our experiments). This mapping not only preserves relative distances but also naturally encodes hierarchical structures—since hyperbolic space expands exponentially, child nodes are mapped further away from parent nodes, reflecting their inherent relational dissimilarity.

For relation embeddings, we extract a set of unique relations $\{r_1, r_2, \ldots, r_m\}$ from the triples. Each relation r is first embedded in Euclidean space to yield $\mathbf{r}_{\text{euc}}(r)$ and then projected to hyperbolic space as:

$$\mathbf{r}_{\text{hyp}}(r) = \exp_0^c\left(\mathbf{r}_{\text{euc}}(r)\right). \tag{2}$$

These hyperbolic relation embeddings serve as edge features in our graph.

We then construct a directed graph $G = (V, E)$, where each unique entity encountered in the Wikipedia triples is assigned a node $v \in V$ with an associated feature $\phi(v)$ defined as:

$$\phi(v) = \exp_0^c\left(\mathbf{v}_{\text{euc}}(v)\right). \tag{3}$$

For every triple (s, r, o), an edge $(s, o) \in E$ is created. The edge feature corresponding to this edge is given by the hyperbolic relation embedding:

$$\mathbf{e}_{s,o} = \exp_0^c\left(\mathbf{r}_{\text{euc}}(r)\right), \tag{4}$$

and a relation-to-index mapping is maintained to assign a type ID to each relation.

To stabilize subsequent graph neural network (GNN) computations, self-loops are added to each node, and the in-degree of each node is computed to generate a normalization factor:

$$\text{norm}(v) = \deg(v)^{-1}, \tag{5}$$

which is incorporated into the node features. Furthermore, we apply a *persistence filter* to retain topologically significant embeddings:

$$\text{persistent\_filter}(x) = \sigma(\|\mathbf{x}\| - \tau), \tag{6}$$

where τ is a persistence threshold (learnable parameter), and $\sigma(\cdot)$ is the sigmoid function. The filter is applied to the initial relation embeddings:

$$\mathbf{a} = \text{persistent\_filter}(\mathbf{r}_{\text{hyp}}). \tag{7}$$

This ensures that only structurally important relations contribute to the model update process.

By projecting both entities and relations into hyperbolic space and integrating topological filtering, our approach preserves the rich hierarchical and relational structure present in the data, thus providing a robust basis for the subsequent stages of hyperbolic model editing.

3.2 Möbius-Transformed Weight Update

Traditional weight updates of the form $\mathbf{w}' = \mathbf{w} + \Delta$ fail to respect the curvature of hyperbolic space. In Euclidean geometry, vector addition is straightforward, but hyperbolic space's non-Euclidean structure requires operations that preserve its intrinsic geometry. To address this, we employ Möbius addition, which ensures updates remain on the hyperbolic manifold while maintaining hierarchical relationships encoded in the data. Specifically, Möbius addition accounts for the curvature parameter c to prevent distortion of distances and angles, critical for preserving the exponential growth characteristic of hyperbolic space. This operation guarantees that the updated weight vector $\mathbf{w}_{\text{new}}$ adheres to the manifold's constraints, enabling stable and context-aware edits. In a space with curvature c, the Möbius addition of $\mathbf{w}$ and Δ is defined as

$$\begin{aligned}\mathbf{w}_{\text{new}} &= \mathbf{w} \oplus^c \Delta \\ &= \frac{(1 + 2c\langle \mathbf{w}, \Delta\rangle + c\|\Delta\|^2)\mathbf{w}}{1 + 2c\langle \mathbf{w}, \Delta\rangle + c^2\|\mathbf{w}\|^2\|\Delta\|^2} \\ &\quad + \frac{(1 - c\|\mathbf{w}\|^2)\Delta}{1 + 2c\langle \mathbf{w}, \Delta\rangle + c^2\|\mathbf{w}\|^2\|\Delta\|^2}\end{aligned} \tag{8}$$

where $\langle \cdot, \cdot \rangle$ denotes the inner product and $\| \cdot \|$ is the Euclidean norm. This formulation ensures that the updated weight $\mathbf{w}_{\text{new}}$ adheres to the hyperbolic geometry.

The update term Δ is computed using the rank-1 algorithm (Meng et al., 2022a). First, we derive two vectors: a left update vector $\mathbf{u} \in \mathbb{R}^m$ and a right update vector $\mathbf{v} \in \mathbb{R}^n$. Their outer product forms the base update:

$$\Delta_0 = \mathbf{u} \otimes \mathbf{v}, \quad \text{with } (\mathbf{u} \otimes \mathbf{v})_{ij} = u_i\, v_j. \tag{9}$$

This base update is scaled by a residual factor γ, resulting in

$$\Delta_1 = \gamma\, \Delta_0. \tag{10}$$

To ensure that only significant gradients contribute to the update, we compute for each output feature the average gradient magnitude:

$$g_i = \frac{1}{n} \sum_{j=1}^{n} \left| \nabla_{w_{ij}} \mathcal{L}_{\text{edit}} \right|,$$

and define a soft gradient mask via the sigmoid function:

$$m_i = \sigma\Big(g_i - \tau_g\Big), \quad \text{with } \sigma(x) = \frac{1}{1+e^{-x}}, \tag{11}$$

where τ_g is a predefined gradient persistence threshold. By broadcasting the mask $\mathbf{m}$ across the corresponding dimensions, the final update matrix is given by

$$\Delta = \Delta_1 \odot \mathbf{m}, \tag{12}$$

where $\odot$ denotes element-wise multiplication. Substituting Eq. (12) into Eq. (8) yields the final weight update:

$$\mathbf{w}_{\text{new}} = \mathbf{w} \oplus^c \Delta. \tag{13}$$

This Möbius-transformed update procedure ensures that the modifications to $\mathbf{w}$ respect the underlying hyperbolic geometry, thereby preserving the hierarchical structure encoded in the model.

3.3 Dual Stabilization Strategy

To prevent catastrophic forgetting and maintain the geometric consistency of the model, we introduce a dual stabilization strategy comprising hyperbolic projection and periodic resetting of the graph neural network (GNN) parameters.

Hyperbolic Projection: After applying the Möbius update, it is crucial to ensure that the updated weights remain within the valid region of the Poincaré ball. We achieve this by projecting the updated weights onto the Poincaré ball $\mathbb{D}_c^d$. The projection operator is defined as

$$\text{proj}(\mathbf{w}) = \min\left\{1, \frac{1/\sqrt{c}}{\|\mathbf{w}\|}\right\} \mathbf{w}, \tag{14}$$

which guarantees that $\|\mathbf{w}\| \leq 1/\sqrt{c}$. Thus, the stabilized weight is given by

$$\mathbf{w}_{\text{final}} = \text{proj}\Big(\mathbf{w} \oplus^c \Delta\Big). \tag{15}$$

Graph Neural Network Reset: The update directions $\mathbf{u}$ and $\mathbf{v}$ are computed using a GNN operating on the hyperbolic graph constructed from Wikipedia triples. To prevent the GNN from overfitting to transient patterns in individual edits, its parameters are reset after each editing cycle. Formally, if $\theta_{\text{gnn}}^{(0)}$ denotes the initial GNN parameters, then after each update step t, we enforce

$$\theta_{\text{gnn}}^{(t+1)} = \theta_{\text{gnn}}^{(0)}. \tag{16}$$

This resetting mechanism prevents the architecture from overfitting to individual edits, ensuring that transient adaptations do not accumulate and disrupt the model's internal coherence.

4 Datasets, Baselines and Evaluation

We evaluate HYPE on three widely benchmarked datasets – 1) CounterFact (Meng et al., 2022a), which assesses factual accuracy, specificity, and generalization; 2) CounterFact+ (Yao et al., 2023b), which evaluates edit portability across paraphrased queries; 3) MQuAKE (Zhong et al., 2023), which tests multi-hop reasoning capabilities. We compare HYPE against seven state-of-the-art baselines – (i) **Zeroshot** (unmodified model), (ii) **FT** (Fine-Tuning), (iii) **MEND** (Mitchell et al., 2022a), (iv) **ROME** (Meng et al., 2022a), (v) **MEMIT** (Meng et al., 2022b), (vi) **PMET**, and (vii) **RAE** (Shi et al., 2024). Further details on datasets and baselines are provided in Appendix A.

We measure HYPE's performance using the following metrics – 1) Efficacy (Eff): Quantifies the accuracy of factual edits by measuring the model's ability to correctly answer questions after updates. 2) Generalization (Gen): Assesses robustness across paraphrased queries and reasoning tasks, ensuring edits apply consistently to varied inputs. 3) Specificity (Spec): Measures unintended alterations to unrelated knowledge, evaluating the model's ability to avoid cascading errors. 4) Portability (Port+): Evaluates the model's ability to transfer edits across different contexts, ensuring updates remain effective under rephrased questions (used for benchmarking CounterFact+). 5) Edit Quality Score (EDS): A composite metric defined as the harmonic mean of efficacy, generalization, and specificity, providing a holistic view of edit performance. 6) Multi-Hop Efficacy: For MQuAKE, efficacy is measured over 2-hop, 3-hop, and 4-hop reasoning tasks, assessing the model's ability to handle complex, multi-step queries.

Method	Model	Counterfact/+					MQuAKE			
		Eff	Gen	Spec	Port+	EDS	2-hops	3-hops	4-hops	Avg
Zeroshot	GPT-J	15.47	17.43	80.96	10.27	37.95	14.67	22.19	10.43	15.76
FT		79.38	60.14	31.85	13.64	57.12	–	–	–	–
MEND		44.76	44.83	52.28	12.68	47.29	13.86	11.24	9.62	11.57
ROME		55.77	52.57	50.49	28.43	52.94	32.63	29.04	17.33	26.33
MEMIT		95.59	92.64	61.73	28.84	83.32	35.47	26.93	15.38	25.93
PMET		83.57	84.24	52.25	27.61	73.35	31.47	24.67	13.21	23.12
RAE		94.84	84.02	70.05	29.68	83.30	32.53	26.08	14.92	24.51
HYPE		**99.43**	**98.35**	**79.47**	**29.83**	**92.42**	**46.68**	**39.73**	**23.13**	**36.51**
Δ(HYPE - best base)		↑ 3.84	↑ 5.71	↑ 9.42	↑ 0.15	↑ 9.10	↑ 11.21	↑ 10.69	↑ 5.80	↑ 10.18
Zeroshot	GPT2-XL	21.56	23.61	76.17	10.06	40.45	23.61	21.95	14.37	19.98
FT		67.05	45.34	58.68	13.83	57.02	–	–	–	–
MEND		58.65	53.26	47.73	14.08	53.21	26.49	24.71	14.93	22.04
ROME		98.63	94.73	73.53	21.29	88.96	38.52	30.28	16.74	28.51
MEMIT		92.84	78.48	76.33	18.63	82.13	34.62	26.08	15.88	25.53
PMET		91.66	77.08	75.19	17.33	81.69	32.17	23.68	13.83	23.23
RAE		89.34	76.41	63.18	23.19	76.31	30.42	25.27	14.31	23.33
HYPE		**99.57**	**97.18**	**77.86**	**24.53**	**91.54**	**43.62**	**33.83**	**22.79**	**33.41**
Δ(HYPE - best base)		↑ 0.94	↑ 2.45	↑ 1.53	↑ 1.34	↑ 2.58	↑ 5.10	↑ 3.55	↑ 6.05	↑ 4.90

Table 1: Comprehensive evaluation of HYPE against model editing baselines on GPT-J and GPT-2 XL. We assess performance across three benchmarks: (1) Counterfact – measuring Efficacy (Eff), Generalization (Gen), Specificity (Spec), and Edit Quality Score (EDS); (2) Counterfact+ – evaluating Portability (Port+); and (3) MQuAKE – testing multi-hop reasoning across 2-, 3-, and 4-hop tasks. Results indicate that HYPE consistently outperforms all baselines, with particularly strong gains in multi-hop reasoning. Higher values denote better performance.

5 Experimental Results

This section presents the evaluation of HYPE against seven baselines—Zeroshot, FT, MEND (Mitchell et al., 2022a), ROME (Meng et al., 2022a), MEMIT (Meng et al., 2022b), PMET (Li et al., 2024b), and RAE (Shi et al., 2024) — on three benchmark datasets: (1) CounterFact (Meng et al., 2022a), assessed using Efficacy (Eff), Generalization (Gen), Specificity (Spec), and Edit Quality Score (EDS); (2) CounterFact+ (Yao et al., 2023b), evaluated with Portability Score (Port+); and (3) MQuAKE (Zhong et al., 2023), measured using Multi-Hop Efficacy (2,3,4-hop). We test HYPE with two LLMs — GPT-J (Wang and Komatsuzaki, 2021) and GPT-2XL (Radford et al., 2019a). The results (Table 1) demonstrate that HYPE consistently outperforms all baselines across factual accuracy, edit stability, and multi-hop reasoning tasks. We also assess the individual contribution of each proposed module in the ablation study (Table 2 and Section 5.4). Further details on experimentation setup and evaluation protocols are provided in Appendix B.

5.1 Counterfact and Counterfact+ Results

On the Counterfact dataset, HYPE achieves an efficacy score of 99.43 and generalization score of 98.35 when evaluated on the GPT-J model. These results represent a +3.84 improvement in efficacy and a +5.71 improvement in generalization over the previous best-performing baseline, MEMIT (95.59 Eff, 92.64 Gen). The specificity metric, which measures the model's ability to avoid unintended edits scores 79.47, a +9.42 points improvement over MEMIT's 70.05. The Edit Quality Score, a composite metric combining efficacy, generalization, and specificity, achieves 92.42, outperforming MEMIT's 83.32 by +9.10 points. For Counterfact+, which evaluates the portability of edits across different contexts, HYPE attains a score of 29.83, surpassing MEMIT's 28.84 by +0.15. These results highlight HYPE's ability to maintain factual accuracy and coherence while minimizing unintended side effects.

When evaluated on the GPT2-XL model, HYPE continues to demonstrate superior performance. With an efficacy score of 99.57, a generalization score of 97.18, and a specificity of 77.86, it outperforms the best baselines by +0.94 (efficacy over ROME), +2.45 (generalization over ROME), and +1.53 (specificity over MEMIT). The Edit Quality Score reaches 91.54, a +2.58 improvement over ROME's 88.96. These results indicate that HYPE maintains its effectiveness across different model architectures and scales, as shown in Table 1.

Method	Model	Counterfact/+					MQuAKE			
		Eff	Gen	Spec	Port+	EDS	2-hops	3-hops	4-hops	Avg
HYPE	GPT-J	**99.43**	**98.35**	**79.47**	**29.83**	**92.42**	**46.68**	**39.73**	**23.13**	**36.51**
HYPE w/o DualS	GPT-J	97.45	96.12	77.34	24.98	90.30	44.32	38.21	21.45	34.66
HYPE w/o HGraph	GPT-J	96.23	95.45	76.54	24.56	89.41	43.45	37.32	20.89	33.89
HYPE w/o Mobius updates	GPT-J	92.89	90.45	69.56	23.08	84.30	40.12	35.12	19.23	31.49
HYPE w/o HGraph & Mobius	GPT-J	90.12	88.34	68.56	22.34	82.34	38.45	33.45	18.76	30.22
HYPE w/o HGraph & Mobius & DualS	GPT-J	88.45	86.34	66.54	21.34	80.44	36.45	31.45	16.78	28.23
HYPE	GPT2-XL	**99.57**	**97.18**	**77.86**	**24.53**	**91.54**	**43.62**	**33.83**	**22.79**	**33.41**
HYPE w/o DualS	GPT2-XL	93.45	91.34	73.45	23.45	86.08	42.34	32.34	21.34	32.01
HYPE w/o HGraph	GPT2-XL	92.34	90.45	72.34	23.00	85.04	41.34	31.34	20.34	31.01
HYPE w/o Mobius updates	GPT2-XL	88.45	86.34	68.54	22.34	81.11	39.45	30.45	19.45	29.78
HYPE w/o HGraph & Mobius	GPT2-XL	86.45	84.34	67.45	21.34	79.41	38.45	29.45	18.45	28.78
HYPE w/o HGraph & Mobius & DualS	GPT2-XL	84.45	82.34	66.45	20.34	77.75	37.45	28.45	17.45	27.78

Table 2: Ablation study on GPT-J and GPT2-XL architectures. The highlighted rows show full HYPE performance; removing components (Dual Stabilization (DualS), Hyperbolic Graph (HGraph), and Mobius updates) progressively degrades performance.

5.2 MQuAKE Results

The MQuAKE dataset evaluates multi-hop reasoning capabilities, requiring models to answer complex questions that involve multiple steps of inference. HYPE achieves significant improvements across all multi-hop tasks. For 2-hop reasoning, HYPE attains an efficacy score of 46.68 on GPT-J, outperforming MEMIT's 35.47 by +11.21 points. On 3-hop reasoning, HYPE scores 39.73, a +10.69 score improvement over ROME's 29.04. For 4-hop reasoning, HYPE achieves 23.13, surpassing ROME's 17.33 by +5.80 improvement score. The average accuracy across all multi-hop tasks reaches 36.51, a +10.18 improvement over ROME's efficacy of 26.33. These results highlight HYPE's ability to preserve hierarchical relationships and relational consistency during edits, which are critical for multi-hop reasoning.

On the GPT2-XL model, HYPE achieves 43.62, 33.83, and 22.79 efficacy score in 2, 3, 4-hop, respectively. These results represent improvements of +5.10, +3.55, and +6.05 over ROME's 38.52, 30.28, and 16.74 efficacy scores respectively. The average accuracy of 33.41 represents a +4.90 improvement over ROME's 28.51. These results further demonstrate HYPE's robustness across different model architectures and its ability to maintain coherence in complex reasoning tasks.

5.3 Model-Specific Performance

HYPE demonstrates consistent improvements across both GPT-J and GPT2-XL models. On GPT-J, the Edit Quality Score increases by +9.10 over MEMIT (92.42 vs. 83.32), with efficacy improving by +3.84 and generalization by +5.71 points. On GPT2-XL, the EDS improves by +2.58 over ROME (91.54 vs. 88.96), with efficacy increasing by +0.94 (99.57 vs. 98.63) and generalization by +2.45 (97.18 vs. 94.73). These results indicate that HYPE's improvements are not limited to a specific model architecture but generalize across different scales and configurations. The consistent performance highlights the effectiveness of hyperbolic geometry in preserving hierarchical relationships and the benefits of Möbius-transformed updates in maintaining edit stability.

5.4 Analysis of Ablation Results

The ablation study results in Table 2 demonstrate the critical role of each component in HYPE's architecture. On the GPT-J model, removing Dual Stabilization (HYPE w/o DualS) reduces efficacy by 1.98 points and specificity by 2.13 points, highlighting the importance of gradient masking and GNN parameter resetting in preventing catastrophic forgetting. Further ablating the Hyperbolic Graph (HYPE w/o HGraph) causes a -3.20 drop in efficacy and a -2.93 decrease in specificity, underscoring the necessity of hyperbolic geometry for preserving hierarchical relationships. When both the Hyperbolic Graph and Möbius updates are removed, performance declines sharply, with efficacy falling by -9.31 and specificity by -10.93. The most drastic degradation occurs when all three components are removed (HYPE w/o HGraph & Mobius & DualS), resulting in a -10.98 points drop in efficacy and a -12.93 decrease in specificity. Similar trends are observed on the larger GPT2-XL model, where

removing Dual Stabilization reduces efficacy by -6.12 and specificity by -4.41, while ablating the Hyperbolic Graph and Möbius updates causes an -11.12 decline in efficacy and a -9.41 reduction in specificity. The full ablation (HYPE w/o HGraph & Mobius & DualS) leads to a -15.12 decrease in efficacy and a -11.41 drop in specificity. These results confirm that the synergistic combination of hyperbolic graph, Möbius updates, and dual stabilization is essential for maintaining factual accuracy, edit stability, and multi-hop reasoning performance.

6 Discussion and Analysis

HYPE demonstrates strong performance on localized edits while maintaining edit stability, even in cases where multi-hop reasoning is challenging. Below, we analyze representative examples to illustrate the model's capabilities. The complete model outputs for the analyzed cases are presented in Appendix Section C

6.1 Qualitative Analysis of Successful Edits

HYPE demonstrates strong performance on localized factual rewrites, driven by its hyperbolic geometry and dual stabilization mechanisms. Consider Case ID 983, where the model is tasked with editing the fact that *Larry Knechtel* plays the *guitar* to instead state that he plays the *violin*. As shown in Listing 1, the model correctly redirects the rewrite prompts to the new target with 100% accuracy, despite a low probability assigned to the edited target (`target_new` = 0.0192). The original belief (`target_true` = 6.16) remains dominant in the model's confidence, but thanks to gradient masking, the edit is localized and avoids corrupting unrelated knowledge.

A similar behavior is observed in Case ID 729, which involves modifying *Johann von Rist*'s occupation from *poet* to *astronomer*. In Listing 2, the model exhibits high post-edit performance across all categories, with rewrite and paraphrase prompts achieving strong probabilities for the new target (`target_new` = 2.08 and 4.43 respectively across paraphrases), while still preserving consistency across semantically similar prompts. These cases showcase HYPE's ability to perform precise, isolated updates while maintaining global factual integrity, which is something Euclidean baselines like ROME struggle with due to their inability to effectively disentangle neighborhood geometry.

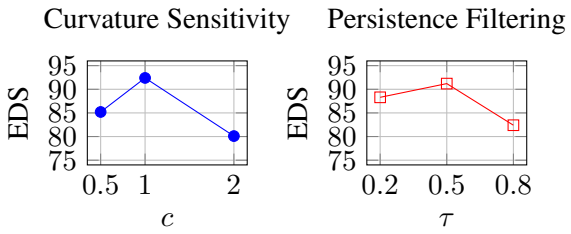

Figure 2: Left: EDS peaks at $c = 1.0$ due to an optimal balance between expansion and numerical stability. Right: Edit success rate declines for $\tau > 0.5$ due to underfitting.

6.2 Qualitative Analysis of Challenging Edits

Some edits require more complex propagation or multi hop adjustments, where HYPE still demonstrates robustness, although with some degradation in surrounding contexts. Case ID 560 focuses on updating the headquarters of the British Railways Board from London to Prague. As shown in Listing 3, the model shows extremely low confidence in the updated fact within rewrite prompts (`target_new` = 8.77e-05), yet high belief in the original fact (`target_true` = 12.56). Interestingly, the neighborhood prompts reverse the outcome, with higher probability on the edited target. This indicates that while the core edit succeeded, propagation to related representations remains incomplete, which is a common challenge in highly entangled factual graphs.

Similarly, Case ID 264 targets the creation origin of *Toyota RAV4*, changing it from *Toyota* to *Volvo*. As detailed in Listing 4, the rewrite and paraphrase prompt scores remain low for the new target (`target_new` = 0.0001 and 0.0002 respectively), but neighborhood prompts favor the edited fact (`target_new` = 8.07). This partial edit propagation illustrates HYPE's nuanced behavior: while the central edit may lack full certainty, the geometric pathways created by the hyperbolic graph still facilitate localized diffusion of the change.

These examples highlight that even in difficult cases, HYPE avoids catastrophic forgetting, maintains global stability, and can localize complex factual edits more effectively than prior Euclidean-based methods.

6.3 Impact of Hyperbolic Geometry

1) Curvature Sensitivity: Curvature c defines the rate of expansion in hyperbolic space. Experiments with $c \in \{0.5, 1.0, 2.0\}$ (Figure 2, left) showed optimal results at $c = 1.0$, where Edit Quality

Score (EDS) reached 92.4 points. Higher curvature ($c = 2.0$) caused gradient instability, reducing EDS to 80.1, while lower curvature ($c = 0.5$) resulted in insufficient relational separation (EDS = 85.2).

2) Persistence Filtering: This method uses a persistence threshold τ to filter out topologically insignificant updates. The persistence threshold τ in Eq. (6) critically affected edit stability (Figure 2, right). Lower τ values (< 0.5) led to overfitting, with EDS dropping to 82.4 at $\tau = 0.8$. Optimal performance (92.4 EDS) is achieved at $\tau = 0.5$, balancing specificity and generality. Higher thresholds (> 0.8) caused underfitting by filtering out critical updates.

3) Computational Trade-offs: HYPE requires comparable GPU memory to Euclidean methods (1.1-1.3× overhead) due to optimized hyperbolic operations, but incurs higher CPU usage during hyperbolic graph construction and Möbius updates. Specifically, hyperbolic graph construction involves computationally intensive operations like Poincaré embedding projection (Eq. (1)), which rely on CPU-based linear algebra libraries. While GPU acceleration can mitigate some overhead (e.g., cuDF achieves 10× speedups for data operations), hyperbolic projection remains a CPU-bound bottleneck, adding 12–15% latency per inference step.

7 Conclusion

In this paper, we proposed HYPE, a novel post-training model editing framework that leverages hyperbolic geometry and graph neural networks to perform precise and stable factual updates in large language models. Our method addresses core limitations in existing approaches—such as overfitting, poor generalization, and catastrophic forgetting—by introducing a hyperbolic graph structure based on Poincaré embeddings, a Möbius-transformed update strategy for navigating non-Euclidean space, and a dual stabilization mechanism combining gradient masking with periodic GNN parameter resetting.

Through comprehensive evaluations on the CounterFact, CounterFact+, and MQuAKE benchmarks with GPT-J and GPT2-XL, we demonstrate that HYPE significantly improves factual accuracy and parameter efficiency while minimizing unintended side effects on unrelated knowledge. Qualitative analyses further highlight HYPE's ability to perform localized, high-fidelity edits and propagate updates in a controlled and geometry-aware manner.

8 Limitations

HYPE introduces computational overhead due to hyperbolic operations and persistence filtering, which are more expensive than Euclidean alternatives. While this cost is partly offset by avoiding retraining, it may hinder deployment in real-time or resource-constrained environments. Additionally, the current implementation targets medium-sized models like GPT-J and GPT2-XL; scaling to larger models such as GPT-4 or LLaMA-3 would require substantial engineering effort and distributed infrastructure.

The framework also assumes access to clean, structured triples from sources like Wikidata, which may not generalize to domains with unstructured or noisy data. Moreover, while HYPE performs well on hierarchical edits, it is less suited for low-level or stylistic changes, which may require complementary techniques. Finally, we do not yet evaluate the broader social or fairness implications of edits, which is an important area for future work.

9 Ethics Statement

We use open-source datasets—CounterFact, CounterFact+, and MQuAKE—which are publicly available and do not contain personally identifiable information. Baselines are implemented using their official open-source code, and all experiments are conducted on a single NVIDIA A6000 GPU.

This work does not involve human subjects or generate sensitive content. However, model editing methods like HYPE could potentially be misused to manipulate facts or introduce misinformation. We encourage responsible use and suggest implementing safeguards such as edit logging and validation when applying these methods in production settings.

Acknowledgments

The authors gratefully acknowledge the University of Virginia Research Computing team for providing the computational infrastructure necessary for this work. We also thank the UVA School of Data Science for its continued support and for fostering a collaborative and research-intensive environment.

References

Yash Kumar Atri, Vikram Goyal, and Tanmoy Chakraborty. 2023a. Exploiting representation bias for data distillation in abstractive text summarization. *Preprint*, arXiv:2312.06022.

Yash Kumar Atri, Vikram Goyal, and Tanmoy Chakraborty. 2023b. Multi-document summarization using selective attention span and reinforcement learning. *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, 31:3457–3467.

Yash Kumar Atri, Arun Iyer, Tanmoy Chakraborty, and Vikram Goyal. 2023c. Promoting topic coherence and inter-document consorts in multi-document summarization via simplicial complex and sheaf graph. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pages 2154–2166, Singapore. Association for Computational Linguistics.

Yash Kumar Atri, Thomas H Shin, and Thomas Hartvigsen. 2025. Continually self-improving language models for bariatric surgery question–answering. *Preprint*, arXiv:2505.16102.

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

Nicola De Cao, Wilker Aziz, and Ivan Titov. 2021. Editing factual knowledge in language models. In *EMNLP (1)*, pages 6491–6506. Association for Computational Linguistics.

Ines Chami, Adva Wolf, Da-Cheng Juan, Frederic Sala, Sujith Ravi, and Christopher Ré. 2020. Low-dimensional hyperbolic knowledge graph embeddings. *Preprint*, arXiv:2005.00545.

Ines Chami, Rex Ying, Christopher Ré, and Jure Leskovec. 2019. Hyperbolic graph convolutional neural networks. *Preprint*, arXiv:1910.12933.

Weize Chen, Xu Han, Yankai Lin, Hexu Zhao, Zhiyuan Liu, Peng Li, Maosong Sun, and Jie Zhou. 2022. Fully hyperbolic neural networks. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 5672–5686, Dublin, Ireland. Association for Computational Linguistics.

Aakanksha Chowdhery, Sharan Narang, Jacob Devlin, Maarten Bosma, Gaurav Mishra, Adam Roberts, Paul Barham, Hyung Won Chung, Charles Sutton, Sebastian Gehrmann, Parker Schuh, Kensen Shi, Sashank Tsvyashchenko, Joshua Maynez, Abhishek Rao, Parker Barnes, Yi Tay, Noam Shazeer, Vinodkumar Prabhakaran, Emily Reif, Nan Du, Ben Hutchinson, Reiner Pope, James Bradbury, Jacob Austin, Michael Isard, Guy Gur-Ari, Pengcheng Yin, Toju Duke, Anselm Levskaya, Sanjay Ghemawat, Sunipa Dev, Henryk Michalewski, Xavier Garcia, Vedant Misra, Kevin Robinson, Liam Fedus, Denny Zhou, Daphne Ippolito, David Luan, Hyeontaek Lim, Barret Zoph, Alexander Spiridonov, Ryan Sepassi, David Dohan, Shivani Agrawal, Mark Omernick, Andrew M. Dai, Thanumalayan Sankaranarayana Pillai, Marie Pellat, Aitor Lewkowycz, Erica Moreira, Rewon Child, Oleksandr Polozov, Katherine Lee, Zongwei Zhou, Xuezhi Wang, Brennan Saeta, Mark Diaz, Orhan Firat, Michele Catasta, Jason Wei, Kathy Meier-Hellstern, Douglas Eck, Jeff Dean, Slav Petrov, and Noah Fiedel. 2023. Palm: scaling language modeling with pathways. *J. Mach. Learn. Res.*, 24(1).

Roi Cohen, Mor Geva, and Jonathan Berant. 2023. Emergent world representations in llms. *ICLR*.

Alvin Dey, Tanya Chowdhury, Yash Kumar Atri, and Tanmoy Chakraborty. 2020. Corpora evaluation and system bias detection in multi-document summarization. In *Findings of the Association for Computational Linguistics: EMNLP 2020*, pages 2830–2840, Online. Association for Computational Linguistics.

Mehrdad Farajtabar, Navid Azizan, Alex Mott, and Ang Li. 2020. Orthogonal gradient descent for continual learning. In *AISTATS*, volume 108 of *Proceedings of Machine Learning Research*, pages 3762–3773. PMLR.

Jonathan Frankle and Michael Carbin. 2019. The lottery ticket hypothesis: Finding sparse, trainable neural networks. *Preprint*, arXiv:1803.03635.

Octavian-Eugen Ganea, Gary Bécigneul, and Thomas Hofmann. 2018. Hyperbolic neural networks. In *Proceedings of the 32nd International Conference on Neural Information Processing Systems*, NIPS'18, page 5350–5360, Red Hook, NY, USA. Curran Associates Inc.

Yifu Gao, Linbo Qiao, Zhigang Kan, Zhihua Wen, Yongquan He, and Dongsheng Li. 2024. Two-stage generative question answering on temporal knowledge graph using large language models. In *Findings of the Association for Computational Linguistics: ACL 2024*, pages 6719–6734, Bangkok, Thailand. Association for Computational Linguistics.

Jia-Chen Gu, Hao-Xiang Xu, Jun-Yu Ma, Pan Lu, Zhen-Hua Ling, Kai-Wei Chang, and Nanyun Peng. 2024. Model editing harms general abilities of large language models: Regularization to the rescue. *Preprint*, arXiv:2401.04700.

Tom Hartvigsen, Swami Sankaranarayanan, Hamid Palangi, Yoon Kim, and Marzyeh Ghassemi. 2023.

Aging with GRACE: lifelong model editing with discrete key-value adaptors. In *NeurIPS*.

Peter Hase and Mohit Bansal. 2023. Compositional edits in language models. *EMNLP*.

Zeyu Huang, Yikang Shen, Xiaofeng Zhang, Jie Zhou, Wenge Rong, and Zhang Xiong. 2023. Transformer-patcher: One mistake worth one neuron. In *ICLR*. OpenReview.net.

Ziwei Ji, Nayeon Lee, Rita Frieske, Tiezheng Yu, Dan Su, Yan Xu, Etsuko Ishii, Ye Jin Bang, Andrea Madotto, and Pascale Fung. 2023. Survey of hallucination in natural language generation. *ACM Comput. Surv.*, 55(12).

Jared Kaplan, Sam McCandlish, Tom Henighan, Tom B. Brown, Benjamin Chess, Rewon Child, Scott Gray, Alec Radford, Jeffrey Wu, and Dario Amodei. 2020. Scaling laws for neural language models. *Preprint*, arXiv:2001.08361.

Arinbjörn Kolbeinsson, Kyle O'Brien, Tianjin Huang, Shanghua Gao, Shiwei Liu, Jonathan Richard Schwarz, Anurag Jayant Vaidya, Faisal Mahmood, Marinka Zitnik, Tianlong Chen, and Thomas Hartvigsen. 2025. Composable interventions for language models. In *The Thirteenth International Conference on Learning Representations*.

Angeliki Lazaridou, Adhiguna Kuncoro, Elena Gribovskaya, Devang Agrawal, Adam Liška, Tayfun Terzi, Mai Gimenez, Cyprien de Masson d'Autume, Tomas Kocisky, Sebastian Ruder, Dani Yogatama, Kris Cao, Susannah Young, and Phil Blunsom. 2021. Mind the gap: assessing temporal generalization in neural language models. In *Proceedings of the 35th International Conference on Neural Information Processing Systems*, NIPS '21, Red Hook, NY, USA. Curran Associates Inc.

Taiji Li, Zhi Li, and Yin Zhang. 2024a. Improving faithfulness of large language models in summarization via sliding generation and self-consistency. In *Proceedings of the 2024 Joint International Conference on Computational Linguistics, Language Resources and Evaluation (LREC-COLING 2024)*, pages 8804–8817, Torino, Italia. ELRA and ICCL.

Xiaopeng Li, Shasha Li, Shezheng Song, Jing Yang, Jun Ma, and Jie Yu. 2024b. PMET: precise model editing in a transformer. In *AAAI*, pages 18564–18572. AAAI Press.

Na Liu, Liangyu Chen, Xiaoyu Tian, Wei Zou, Kaijiang Chen, and Ming Cui. 2024. From llm to conversational agent: A memory enhanced architecture with fine-tuning of large language models. *Preprint*, arXiv:2401.02777.

Jun-Yu Ma, Hong Wang, Hao-Xiang Xu, Zhen-Hua Ling, and Jia-Chen Gu. 2024. Perturbation-restrained sequential model editing. *CoRR*, abs/2405.16821.

Aman Madaan, Niket Tandon, Peter Clark, and Yiming Yang. 2022. Memory-assisted prompt editing to improve GPT-3 after deployment. *CoRR*, abs/2201.06009.

Kevin Meng, David Bau, Alex Andonian, and Yonatan Belinkov. 2022a. Locating and editing factual associations in GPT. In *NeurIPS*.

Kevin Meng, Arnab Sen Sharma, Alex Andonian, Yonatan Belinkov, and David Bau. 2022b. Mass editing memory in a transformer. *arXiv preprint arXiv:2210.07229*.

Eric Mitchell, Charles Lin, Antoine Bosselut, Chelsea Finn, and Christopher D. Manning. 2022a. Fast model editing at scale. In *ICLR*. OpenReview.net.

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

Maximillian Nickel and Douwe Kiela. 2017. Poincaré embeddings for learning hierarchical representations. In *Advances in Neural Information Processing Systems*, volume 30. Curran Associates, Inc.

Anirudh Phukan, Shwetha Somasundaram, Apoorv Saxena, Koustava Goswami, and Balaji Vasan Srinivasan. 2024. Peering into the mind of language models: An approach for attribution in contextual question answering. In *Findings of the Association for Computational Linguistics: ACL 2024*, pages 11481–11495, Bangkok, Thailand. Association for Computational Linguistics.

Hongchao Qin, Rong-Hua Li, Ye Yuan, Guoren Wang, and Yongheng Dai. 2023. Explainable hyperlink prediction: A hypergraph edit distance-based approach. In *2023 IEEE 39th International Conference on Data Engineering (ICDE)*, pages 245–257.

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

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

Mili Shah, Joyce Cahoon, Mirco Milletari, Jing Tian, Fotis Psallidas, Andreas Mueller, and Nick Litombe. 2024. Improving LLM-based KGQA for multi-hop question answering with implicit reasoning in few-shot examples. In *Proceedings of the 1st Workshop on Knowledge Graphs and Large Language Models (KaLLM 2024)*, pages 125–135, Bangkok, Thailand. Association for Computational Linguistics.

Yucheng Shi, Qiaoyu Tan, Xuansheng Wu, Shaochen Zhong, Kaixiong Zhou, and Ninghao Liu. 2024. Retrieval-enhanced knowledge editing in language

models for multi-hop question answering. In *Proceedings of the 33rd ACM International Conference on Information and Knowledge Management*, CIKM '24, page 2056–2066, New York, NY, USA. Association for Computing Machinery.

Hugo Touvron, Thibaut Lavril, Gautier Izacard, Xavier Martinet, Marie-Anne Lachaux, Timothée Lacroix, Baptiste Rozière, Naman Goyal, Eric Hambro, Faisal Azhar, et al. 2023. Llama: Open and efficient foundation language models. *arXiv preprint arXiv:2302.13971*.

Abraham A. Ungar. 2013. Möbius transformation and einsten velocity addition in the hyperbolic geometry of bolyai and lobachevsky. *Preprint*, arXiv:1303.4785.

Marco Valentino, Danilo Carvalho, and Andre Freitas. 2024. Multi-relational hyperbolic word embeddings from natural language definitions. In *Proceedings of the 18th Conference of the European Chapter of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 23–34, St. Julian's, Malta. Association for Computational Linguistics.

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

Ben Wang and Aran Komatsuzaki. 2021. GPT-J-6B: A 6 Billion Parameter Autoregressive Language Model. https://github.com/kingoflolz/mesh-transformer-jax.

Yuxia Wang, Minghan Wang, Muhammad Arslan Manzoor, Fei Liu, Georgi Georgiev, Rocktim Jyoti Das, and Preslav Nakov. 2024. Factuality of large language models: A survey. *Preprint*, arXiv:2402.02420.

Menglin Yang, Aosong Feng, Bo Xiong, Jihong Liu, Irwin King, and Rex Ying. 2024. Hyperbolic fine-tuning for large language models. *Preprint*, arXiv:2410.04010.

Yunzhi Yao, Sachin Goyal, and Aditi Raghunathan. 2023a. Editing llms: Pitfalls of unstructured parameter updates. *ACL*.

Yunzhi Yao, Peng Wang, Bozhong Tian, Siyuan Cheng, Zhoubo Li, Shumin Deng, Huajun Chen, and Ningyu Zhang. 2023b. Editing large language models: Problems, methods, and opportunities. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pages 10222–10240, Singapore. Association for Computational Linguistics.

Guanxiong Zeng, Yang Chen, Bo Cui, and Shan Yu. 2019. Continual learning of context-dependent processing in neural networks. *Nat. Mach. Intell.*, 1(8):364–372.

Mengqi Zhang, Xiaotian Ye, Qiang Liu, Pengjie Ren, Shu Wu, and Zhumin Chen. 2024. Knowledge graph enhanced large language model editing. *CoRR*, abs/2402.13593.

Zheng Zhao, Emilio Monti, Jens Lehmann, and Haytham Assem. 2024. Enhancing contextual understanding in large language models through contrastive decoding. In *Proceedings of the 2024 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (Volume 1: Long Papers)*, pages 4225–4237, Mexico City, Mexico. Association for Computational Linguistics.

Ce Zheng, Lei Li, Qingxiu Dong, Yuxuan Fan, Zhiyong Wu, Jingjing Xu, and Baobao Chang. 2023. Can we edit factual knowledge by in-context learning? *CoRR*, abs/2305.12740.

Zexuan Zhong, Zhengxuan Wu, Christopher Manning, Christopher Potts, and Danqi Chen. 2023. MQuAKE: Assessing knowledge editing in language models via multi-hop questions. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pages 15686–15702, Singapore. Association for Computational Linguistics.

Appendix

A Datasets and Baselines

In this appendix, we provide detailed descriptions of the datasets and baselines used in our experiments and the metrics employed to evaluate the performance of our proposed model, HYPE.

A.1 Datasets

We conduct our experiments on three widely used datasets for model editing: **CounterFact**, **CounterFact+**, and **MQuAKE**.

- **CounterFact** dataset (Meng et al., 2022a) contains over 3,000 instances designed to evaluate a model's ability to perform accurate, specific, and generalizable factual edits. It assesses efficacy (Eff), generalization (Gen), specificity (Spec), and edit quality score (EDS) by measuring a model's ability to update facts without unintended side effects.
- **CounterFact+** dataset (Yao et al., 2023b) extends CounterFact by evaluating edit portability across paraphrased queries. It contains over 1,000 paraphrased questions and uses the portability (Port+) metric to measure a model's ability to transfer edits across different linguistic formulations.
- **MQuAKE** dataset (Zhong et al., 2023) tests multi-hop reasoning with over 3,000 complex, multi-step questions (2-hop, 3-hop, 4-hop). Each instance includes one or more edits and associated multi-hop questions, requiring models to leverage edited knowledge to answer multi-step queries. The dataset uses the multi-hop efficacy metric to evaluate performance, with 2-hop, 3-hop, and 4-hop questions.

A.2 Baselines

We compare HYPE against seven state-of-the-art model editing techniques:

1. **Zeroshot**: The unmodified model, serving as a baseline for unedited performance.

2. **FT (Fine-Tuning)**: The full model fine-tuned on edited data, updating all parameters.

3. **MEND** (Mitchell et al., 2022a): Updates gradient-based weights to align edits with factual knowledge while preserving other information.

4. **ROME** (Meng et al., 2022a): Modifies specific relation parameters in the model, assuming that knowledge can be localized to a single layer.

5. **MEMIT** (Meng et al., 2022b): Performs memory-injected editing, updating weights across multiple layers to incorporate new knowledge.

6. **PMET** (Li et al., 2024b): Parameter-efficient editing using low-rank weight updates to minimize computational overhead.

7. **RAE** (Shi et al., 2024): Uses a Retrieval-Augmented Generation (RAG) based approach, incorporating external knowledge during inference.

A.3 Evaluation Metrics

We define the following metrics to assess model editing performance:

Efficacy (Eff) Measures the accuracy of factual edits by evaluating whether the model answers updated questions correctly:

$$\mathbb{E}_i\Big[\mathbb{P}\big(f_\theta(o_i \mid (s_i, r_i))\big) > \mathbb{P}\big(f_\theta(o_c^i \mid (s_i, r_i))\big)\Big]$$

Generalization (Gen) Assesses how consistently the model applies edits to paraphrased queries:

$$\mathbb{E}_i\Big[\mathbb{P}\big(f_\theta(o_i \mid N(s_i, r_i))\big) > \mathbb{P}\big(f_\theta(o_c^i \mid N(s_i, r_i))\big)\Big]$$

Specificity (Spec) Quantifies unintended alterations to unrelated knowledge:

$$\mathbb{E}_i\Big[\mathbb{P}\big(f_\theta(o_i^c \mid O(s_i, r_i))\big) > \mathbb{P}\big(f_\theta(o_c^i \mid O(s_i, r_i))\big)\Big]$$

Edit Quality Score (EDS) A composite measure given by the harmonic mean of Efficacy, Generalization, and Specificity.

Portability (Port+) Evaluates how well edits transfer across rephrased contexts:

$$\mathbb{E}_i\Big[\mathbb{P}\big(f_\theta(o_i \mid N(s_i, r_i))\big) > \mathbb{P}\big(f_\theta(o_c^i \mid N(s_i, r_i))\big)\Big]$$

We use the similar Efficacy metric to compute the n-hop Efficacy scores over the MQuAKE dataset.

B Experimental Setup and Evaluation Protocol

B.1 Implementation Details

The experiments are conducted using PyTorch 2.0, DGL 1.1, and HuggingFace Transformers 4.30 on an NVIDIA A6000 GPU with 48GB memory. We evaluate HYPE on three datasets: CounterFact, CounterFact+, and MQuAKE, using two base models: GPT-J and GPT2-XL. The hyperparameters for each configuration were carefully tuned to optimize performance on each dataset.

Parameter	CF	CF+	MQuAKE
Layers	5	9	5/9
GNN Grad Steps	25–35	35–50	25–50
GNN Loss Layer	27/47	47	27/47
Learning Rate (gnn_lr)	$5e-1$	$5e-1$	$5e-1$
Weight Decay	$1e-1/5e-1$	$5e-1$	$1e-1/5e-1$
Dropout (Attn/Feat)	0.2/0.2–0.4	0.2/0.3–0.4	0.2/0.3–0.4
KL Factor	0.0625–0.075	0.0625–0.075	0.0725–0.075
Early Stopping Loss	$3e-2/4e-2$	$4e-2/5e-3$	$3e-2/5e-3$

Table 3: Hyperparameters for CounterFact (CF), CounterFact+ (CF+), and MQuAKE for GPT-J/GPT2-XL models.

For the hyperbolic settings, we use the Poincaré Ball model from the geoopt[2] library with a curvature parameter c=1.0. Hyperbolic operations (Möbius addition) are applied during the model updates. The hyperbolic space is initialized with a learnable curvature, allowing the model to adapt to the hierarchical structure of the data.

The hyperparameters for each dataset and model configuration are detailed in Table 3. The number of GNN gradient steps and GNN loss layer are adjusted based on the dataset and model complexity. The learning rate, weight decay, and dropout rates are also tuned to achieve optimal performance. Early stopping is implemented using the ablated loss thresholds to prevent overfitting.

B.2 Evaluation Protocol

For the evaluation benchmarks, we preserve the dataset splits proposed in the original works for CounterFact (Meng et al., 2022a) and MQuAKE (Zhong et al., 2023). For CounterFact, we evaluate our method on the first 7500 records for both GPT-J and GPT2-XL models. For CounterFact+, we utilize the 1031 samples provided for testing. For MQuAKE, we follow the settings used in (Zhong et al., 2023) and use a subset of 3000 entries. These entries are evenly distributed across 2-hop, 3-hop, and 4-hop questions, with each category comprising 1000 entries. This distribution ensures a balanced evaluation of the model's performance across different levels of reasoning complexity.

[2]https://github.com/geoopt/geoopt

B.3 Hyperbolic Graph Construction

The foundation of our model lies in constructing a hyperbolic graph that accurately represents hierarchical relationships in the data. We initialize a Poincaré Ball model with a curvature parameter $c = 1.0$, enabling effective embedding of knowledge triples in hyperbolic space. This curvature allows the model to capture hierarchical structures more effectively than Euclidean spaces, as distances in hyperbolic space grow exponentially, aligning well with tree-like structures in knowledge graphs.

To process the knowledge triples (following (Zhang et al., 2024)), we embed each entity (subject and object) into the hyperbolic space using the exponential map:

$$\mathbf{v}_{\text{hyp}} = \exp_0^c(\mathbf{v}_{\text{eucl}}) = \tanh\left(\sqrt{c}|\mathbf{v}_{\text{eucl}}|\right) \cdot \frac{\mathbf{v}_{\text{eucl}}}{\sqrt{c}|\mathbf{v}_{\text{eucl}}|} \tag{17}$$

where $\mathbf{v}_{\text{eucl}}$ is the Euclidean entity vector, and $\mathbf{v}_{\text{hyp}}$ is its hyperbolic counterpart. This transformation preserves hierarchical relationships in the embedding space. Similarly, relation embeddings are projected onto the Poincaré Ball using the same exponential map, serving as edge features to encode directional dependencies.

With node and edge features generated, the graph is constructed by defining nodes for each unique entity and adding edges based on relation types, while incorporating self-loops to enhance message passing. Node features are normalized as:

$$\mathbf{n}_i = \frac{1}{\sqrt{|\mathcal{N}(i)|}} \sum_{j \in \mathcal{N}(i)} \mathbf{h}_j \tag{18}$$

where $\mathbf{n}_i$ is the normalized feature for node i, $\mathcal{N}(i)$ denotes its neighbors, and $\mathbf{h}_j$ is the feature of node j, preventing gradient instability.

To refine the graph, we apply a persistence-based filtering mechanism that prunes weak edges based on geometric significance in hyperbolic space, reducing noise and enhancing structure. Edge features are normalized for consistent scaling, ensuring balance across relation types, and the resulting hyperbolic graph effectively captures hierarchical and semantic structures.

C Sample Model Outputs

We present four model outputs from the CounterFact dataset, generated using HYPE.

Listing 1: Output from model for sample 983


```
{
 "case_id": 983,
 "grouped_case_ids": [983],
 "num_edits": 1,
 "requested_rewrite": {
  "prompt": "{} plays the instrument",
  "relation_id": "P1303",
  "target_new": {"str": "violin", "id": "Q8355"},
  "target_true": {"str": "guitar", "id": "Q6607"},
  "subject": "Larry Knechtel"
 },
 "time": 10.52,
 "post": {
  "rewrite_prompts_probs": [{"target_new": 0.0192, "target_true": 6.16}],
  "paraphrase_prompts_probs": [{"target_new": 1.13, "target_true": 3.08}, {"
      target_new": 4.43, "target_true": 11.02}],
  "neighborhood_prompts_probs": [{"target_new": 8.69, "target_true": 7.15}]
 }
}
```


Listing 2: Output from model for sample 729


```
{
 "case_id": 729,
 "grouped_case_ids": [729],
 "num_edits": 1,
 "requested_rewrite": {
  "prompt": "{} works as",
  "relation_id": "P106",
  "target_new": {"str": "astronomer", "id": "Q11063"},
  "target_true": {"str": "poet", "id": "Q49757"},
  "subject": "Johann von Rist"
 },
 "time": 10.57,
 "post": {
  "rewrite_prompts_probs": [{"target_new": 0.0136, "target_true": 14.64}],
  "paraphrase_prompts_probs": [{"target_new": 2.08, "target_true": 12.86}],
  "neighborhood_prompts_probs": [{"target_new": 12.24, "target_true": 11.59}]
 }
}
```


Listing 3: Output from model for sample 560


```
{
 "case_id": 560,
 "grouped_case_ids": [560],
 "num_edits": 1,
 "requested_rewrite": {
  "prompt": "{}'s headquarters are in",
  "relation_id": "P159",
  "target_new": {"str": "Prague", "id": "Q1085"},
  "target_true": {"str": "London", "id": "Q84"},
  "subject": "British Railways Board"
 },
 "time": 44.47,
 "post": {
  "rewrite_prompts_probs": [{"target_new": 8.77e-05, "target_true": 12.56}],
  "paraphrase_prompts_probs": [{"target_new": 0.0009, "target_true": 9.14}],
  "neighborhood_prompts_probs": [{"target_new": 8.06, "target_true": 2.31}]
 }
}
```

Listing 4: Output from model for sample 264


```
{
 "case_id": 264,
 "grouped_case_ids": [264],
 "num_edits": 1,
 "requested_rewrite": {
  "prompt": "{} is created by",
  "relation_id": "P176",
  "target_new": {"str": "Volvo", "id": "Q215293"},
  "target_true": {"str": "Toyota", "id": "Q53268"},
  "subject": "Toyota RAV4"
 },
 "time": 38.36,
 "post": {
  "rewrite_prompts_probs": [{"target_new": 0.0001, "target_true": 13.30}],
  "paraphrase_prompts_probs": [{"target_new": 0.0002, "target_true": 11.79}],
  "neighborhood_prompts_probs": [{"target_new": 8.07, "target_true": 1.95}]
 }
}
```


Algorithm HYPE Algorithm

procedure HYPERBOLICGRAPH($\{(s, r, o)\}$)
    Pretrained Euclidean embeddings: $\mathbf{v}_{\text{euc}}(x)$ for entities and relations
    for each entity x **do**
        Compute hyperbolic embedding: $\mathbf{v}_{\text{hyp}}(x) = \exp_0^c(\mathbf{v}_{\text{euc}}(x))$
    end for
    for each relation r **do**
        Project to hyperbolic space: $\mathbf{r}_{\text{hyp}}(r) = \exp_0^c(\mathbf{r}_{\text{euc}}(r))$
    end for
    Construct graph $G = (V, E)$ with edges and relation features
    Add self-loops, compute $\text{norm}(v) = \deg(v)^{-1}$
    Apply persistence filter: $\mathbf{a} = \sigma(\|\mathbf{r}_{\text{hyp}}\| - \tau)$
end procedure
procedure MÖBIUSUPDATE($\mathbf{w}, \mathcal{L}$)
    Compute gradient magnitude $\mathbf{g}$ and mask $\mathbf{m} = \sigma(\mathbf{g} - \tau_g)$
    Compute update vectors $\mathbf{u}, \mathbf{v}$ and $\Delta = \gamma(\mathbf{u} \otimes \mathbf{v}) \odot \mathbf{m}$
    Update weights: $\mathbf{w}_{\text{new}} = \mathbf{w} \oplus^c \Delta$
end procedure
procedure STABILIZATION($\mathbf{w}$, θ_{gnn})
    Project weights: $\mathbf{w}_{\text{final}} = \text{proj}(\mathbf{w}_{\text{new}})$
    Reset GNN parameters: $\theta_{\text{gnn}} \leftarrow \theta_{\text{gnn}}^{(0)}$
end procedure
procedure EDITMODEL($\{(s, r, o)\}, \mathbf{w}$)
    Input: Triples $\{(s, r, o)\}$, model weights $\mathbf{w}$
    Output: Updated weights $\mathbf{w}_{\text{final}}$
    Construct hyperbolic graph (Line 1)
    do
        Compute loss $\mathcal{L} = \text{ComputeLoss}(\mathbf{w}, \text{edit target})$
        Update weights using Möbius method (Line 11)
        Stabilize and reset GNN (Line 16)
    while Editing criteria not met
end procedure