GENERALIZABLE MODEL EDITING COULD BE MORE THAN DIRECT GENERALIZATION

Anonymous authors
Paper under double-blind review

ABSTRACT

A central desideratum in model editing is generalization to complex tasks, which prior work formalizes under a *direct generalization* paradigm: treating and evaluating LLM behavior as an input-output mapping and expecting a factual update to alter outputs on reasoning tasks. Yet despite substantial empirical effort, the field consensus remains that such generalization is highly challenging and elusive. However, when we deploy the latest models in an in-the-wild chatbot-user setting, we surprisingly find that even classical editing methods exhibit remarkably fluent generalization behavior. This motivates us to revisit the obstacles to generalization and the limitations of the prevailing paradigm. We find that direct generalization is fundamentally constrained both theoretically and practically: theoretically, it suffers from an identifiability issue; and practically, existing approaches attempting to circumvent this are further hindered by the ambiguity of their constructed biases and by vulnerability to being overridden by other factors such as optimization dynamics. We instead identify *compositional generalization*, empowered by the increasingly strong reasoning and composition abilities of modern LLMs, as a theoretically grounded and empirically validated alternative. We validate this phenomenon across multiple models and editing methods, showing that such generalization behavior is widespread. Overall, we argue that generalizable model editing could be more than direct generalization, and call for a paradigm shift toward more generalizable and practically deployable knowledge update techniques.

1 INTRODUCTION

The rapid rise of LLMs has reshaped the landscape of AI. However, the internal knowledge stored within these models inevitably suffers from obsolescence, hallucinations, and factual errors (Ji et al., 2023; Lin et al., 2022). To precisely rectify or update specific factual knowledge in LLMs without full-scale training, Model Editing has emerged as a promising paradigm (Zhang et al., 2024b; Yao et al., 2023), whose objective is to modify specific facts in an efficient and low-resource manner. Specifically, an ideal editing method is expected to satisfy three core desiderata: *Efficacy*, where the model accurately answers the targeted knowledge; *Locality*, which ensures that unrelated knowledge and capabilities remain intact; and *Generalization*, which denotes the capacity not merely to memorize given samples, but to extrapolate the knowledge to unseen complex tasks (Yao et al., 2023).

After years of development, the generalization in model editing remains far from satisfactory. Concretely, current mainstream practice can be summarized as a **direct generalization** paradigm: taking generalization on multi-hop reasoning as an example, given an edit $e_1 = (s_1, r_1, o_1^*)$, mainstream evaluations construct compound question prefixes that have never appeared during editing or training (Zhong et al., 2023; Zhang et al., 2024b); for example, constructing a multi-hop question (s_1, r_1, r_2) and then examining whether the model produces the target answer o_2^* as a direct continuation of that prefix. This essentially treats the model as an input-output mapping, and formalizes generalization as whether the edit can encode the inferential consequence implied by the edit into a new input-output rule. However, the field consensus is that the generalization ability of editing is extremely poor: prior work has shown that overfitting is more commonly observed (Zhang et al., 2025b; Cohen et al., 2024), with little to no fluent and natural generalization behavior, severely limiting the practical applicability of model editing in real-world deployment.

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.

In this paper, however, we for the first time observe an unexpected generalization behavior in a more in-the-wild setting: we apply several classic conventional model editing methods to the latest models and deploy them in the most common chatbot-user interaction manner. Upon querying complex questions, without any additional prompt engineering or structural modifications, we observe that a high proportion of cases exhibit remarkably strong generalization behavior. This motivates us to question whether generalization in model editing is truly as challenging as believed, or whether the observed difficulty arises from fundamental issues inherent in the previous paradigm? In this paper, we conduct a deep investigation into this question.

First, we find that the pervasive failure of generalization is highly correlated with limitations of the paradigm itself, including its implicit evaluation protocol, at both theoretical and practical levels. Theoretically, we introduce the perspective of *identifiability* in autoregressive models: in the editing setting, the completion distribution over out-of-distribution (OOD) and complex tasks is unidentifiable from the edit data alone. Consequently, generalization reduces to selecting a correct model solution from an unidentifiable set of candidate models that all fit the target edit distribution—a process fundamentally governed by inductive biases, which current paradigms leave entirely to the editing algorithm despite these biases remaining poorly understood. From a practical perspective, we discuss existing approaches to achieving generalization and conduct a deeper analysis of the mainstream locate-then-edit approach. We find that this line of methods has long been hindered by two severe obstacles in practice: while such methods can be viewed as manually constructing better inductive biases grounded in interpretability, we find that this endeavor is plagued by the inherent ambiguity of the constructed biases, and that these biases may further be overridden by other biases such as optimization dynamics and engineering factors. We discuss the severity of these issues through two case studies and construct a baseline that adjusts only a few hyperparameters to demonstrate their practical impact on performance; this simple baseline successfully outperforms the established method MEMIT-LTI (Zhang et al., 2025b), which employs carefully constructed constraints to mitigate overfitting. These findings jointly explain why direct generalization has long proved genuinely elusive in the literature.

A major trend in LLMs has been the continuous enhancement of reasoning capabilities, which makes models exhibit increasingly powerful abilities to combine different atomic knowledge and capabilities to solve complex problems (Wei et al., 2022). We propose a new perspective: this inherent meta-ability of composition acts as a powerful structural prior, serving as a generalized, implicit inductive bias for the editing process. This theoretically enables the *compositional generalization* of model editing: if a model possesses the ability to compose the edit with other knowledge and capabilities, generalization need not be accomplished entirely by the editing algorithm but can emerge naturally from the compositional ability. Yet this very capability has been inadvertently suppressed by prior evaluation settings, as the RL-based training and capability enhancement of modern LLMs are built upon chat models (OpenAI, 2023; Yang et al., 2025) rather than the unstructured text completion that previous work focused on. Our experiments demonstrate that methods that overfit under conventional settings can exhibit highly fluent generalization behavior in the wild, with generalization performance strongly correlated with the inherent capabilities of the model. Extensive analyses further validate the promising potential of this paradigm, and we therefore argue that *Generalizable Model Editing Could Be More Than Direct Generalization*. Overall, our contributions are:

- We highlight the limitations of the current direct generalization paradigm and analyze its fundamental obstacles grounded in identifiability perspective.
- We conduct a systematic analysis of existing editing methods based on the above theoretical framework, further identifying several key obstacles faced by mainstream approaches.
- We introduce and validate *compositional generalization* as a new perspective, demonstrating across multiple models and editing methods that this phenomenon is widespread.

2 Related Works

In this section, we briefly review the related work on knowledge editing, with more detailed related work and technical backgrounds provided in Appendix A and B.

Knowledge editing aims to precisely and efficiently update the internal knowledge of LLMs (Wang et al., 2024b; Zhang et al., 2024b), in the mainstream setting each piece of knowledge is formu-

lated as a triple (s, r, o) (De Cao et al., 2021), consisting of a subject s, relation r, and object o. Mainstream approaches can be broadly categorized into three types: fine-tuning based methods (Yao et al., 2023), meta-learning based methods represented by MEND (De Cao et al., 2021; Mitchell et al., 2022), and locate-then-edit based methods represented by ROME (Meng et al., 2022) and MEMIT (Meng et al., 2023). Despite the proficiency of editing knowledge in simple fact recall tasks, recent studies have revealed significant limitations when these methods are applied to complex tasks involving reasoning. For instance, upon introducing multi-hop evaluations, Yao et al. (2023) and Zhong et al. (2023) observed that existing knowledge editing methods lack robustness in such scenarios, and Cohen et al. (2024) demonstrated that methods generally struggle to achieve the "ripple effect". Subsequently, Zhang et al. (2025b) identified the overfitting phenomenon in knowledge editing, highlighting that locate-then-edit methods are particularly prone to subject-related overfitting. Although there have been attempts to improve the generalization of locate-then-edit methods on complex tasks (Zhang et al., 2024a; 2025a; Liu et al., 2025; Zhong et al., 2025; Zhang et al., 2026), performance remains unsatisfactory.

3 Overview of Generalization in Model Updating

The goal of *generalization* that is universally pursued in the model updating literature can be summarized as follows: after a model has undergone a knowledge update, it should continue to exhibit the expected behavior on unseen out-of-distribution (OOD) inputs or tasks. Here, OOD is defined as samples that have *not* explicitly appeared in the current editing request or in any historical training example. For instance, after injecting a piece of new knowledge into a model, one expects the model to automatically apply this knowledge to logical deductions, or more complex reasoning tasks.

Paradigm of Direct Generalization. This expectation can be naturally formalized as requiring the model to correctly answer OOD queries. Taking the widely studied multi-hop reasoning setup as a representative example: suppose we update the model with an atomic fact $e_1 = (s_1, r_1, o_1^*)$, while the model already possesses another internal fact $e_2 = (o_1^*, r_2, o_2^*)$. For the updated language model $F_{\theta'}$, researchers implicitly expect it to satisfy $F_{\theta'}(s_1, r_1, r_2) = o_2^*$. From a probabilistic modeling perspective, this is equivalent to expecting that a **local** modification to the single-step conditional probability $P_{\theta'}(o_1^* \mid s_1, r_1)$ will spontaneously and accurately induce a **directional** change in the composite conditional probability $P_{\theta'}(o_2^* \mid s_1, r_1, r_2)$. We formalize this setting, where parameter updates are expected to directly yield desired outputs on unseen tasks, as **Direct Generalization**.

Menifestation in Practice. The prevailing evaluation protocols in the model updating domain are built upon this notion of "direct generalization." For example, the most popular frameworks like EasyEdit (Wang et al., 2024a) and benchmarks like KnowEdit (Zhang et al., 2024b) evaluate reasoning capability via the metric of *Portability* (Yao et al., 2023). It is implemented to evaluate whether the model can complete the answer tokens for complex questions in unstructured text. Specifically, the evaluation presents the model with a long compositional prompt prefix (e.g., *"The name of the position held by the father of Thomas Davis Lamb is"*), and then checks the accuracy or probability of generating target tokens (e.g., *"deputy high court judge"*) via teacher forcing. This strictly aligns with our formal definition; mainstream works operate under this paradigm and strive to optimize performance within this formulation.

3.1 What Hinders Direct Generalization, Mathematically?

Abundant prior empirical evidence in the model updating literature indicates that, at least under the current direct paradigm, achieving generalization is exceptionally difficult. Models generally perform poorly across various metrics, and their actual outputs tend to be mechanical and rigid, exhibiting overfitting behavior rather than fluent and natural generalization (Zhang et al., 2025b).

In this section we further emphasize the theoretical limitations and requirements of generalization; specifically, the *non-identifiability* inherent to autoregressive language models. The direct generalization paradigm introduces further constraints within this framework and gives rise to the practical issues discussed later. To rigorously elucidate this obstacle, we first introduce two core concepts from the theory of model generalization (Reizinger et al., 2024):

- **Functional Non-identifiability:** Autoregressive (AR) models are essentially parameterized families of conditional probabilities $\{P_\theta(x_i \mid x_{1:i-1})\}$ (which define the model's "function"

or behavioral patterns) that induce joint probability distributions $\{P_\theta(x_{1:i})\}$. Functional identifiability of this family is guaranteed if and only if every finite prefix has a positive probability under the data distribution p: if two models differ solely in how they complete "zero-probability prefixes," they define identical joint probabilities on the support $\mathrm{supp}(p)$ and are thus indistinguishable via $\mathrm{KL}[p \,\|\, P_\theta]$. In other words, a single target data distribution can correspond to multiple, or even infinitely many, AR models that behave differently out-of-distribution.

- **Parameter Non-identifiability:** Even when fixing the family of conditional probabilities represented by the model (i.e., maintaining full equivalence at the functional level), there can still exist multiple distinct parameter configurations θ that realize the exact same function. That is, the mapping from the parameter space to the function space is non-injective; functionally equivalent models can be highly diverse in their parameterizations.

In knowledge updating scenarios, this manifests as multiplicity of solutions: an update request specifies a target string $x^\star$ to describe fact $e_1 = (s_1, r_1, o_1^\star)$. The objective of the update is to fit the target distribution p on this textualization such that $\mathrm{KL}[p \,\|\, P_{\theta'}] = 0$. For any composite query string z required by generalization tasks (e.g., multi-hop reasoning prompts), since it is not explicitly contained in the edit request, we have $p(z) = 0$ in the target data distribution, which does not satisfy the condition for functional identifiability. This implies that the family of conditional probabilities represented by the model (i.e., the functionality or behavior) remains undetermined; in other words, there exist multiple or even infinitely many candidate models capable of fitting the edit target, and they may produce completely opposite or incorrect predictions on the generalization query z. Therefore, fitting the target samples (i.e., optimizing via a loss function) in fact provides no direct guarantee for generalization.

What then determines the model's OOD behavior and generalization capabilities? Given the large number of candidate models that are unidentifiable by the editing objective, which solution is ultimately selected from the set of equivalent solutions is in fact determined by inductive biases in the broad sense. These biases encompass certain parametric properties of the LLM prior to editing as well as those introduced by the algorithm; for example, some methods choose to adjust only a single key layer of the model (Meng et al., 2022). This essentially introduces an inductive bias, restricting the final solution to a subspace of models whose parameters differ only in that specific layer.

Summary. In summary, the direct generalization paradigm inherently understands the overall behavior of a language model as an input-output mapping and expects knowledge updates to synchronously modify the mapping relationships for complex reasoning tasks. Based on the above analysis, the fundamental obstacle of this paradigm lies in the existence of a vast set of non-identifiable candidate model solutions, and selecting a "generalizable solution" that precisely exhibits the desired OOD behavior heavily relies on inductive biases that are not yet well understood or controlled.

3.2 Current Empirical Pathways and Their Dilemmas

In this subsection, we first categorize the primary empirical pathways proposed in existing literature to mitigate the aforementioned theoretical dilemmas, then analyze the practical issues confronting the dominant line of work.

3.2.1 Major Pathways to Achieve OOD Generalization

Based on the foregoing analysis, achieving direct generalization theoretically admits several potential pathways, which correspond to current practices to varying degrees:

(i) **Expanding the support to eliminate non-identifiability**, which converts the OOD generalization problem into an in-distribution problem and avoiding the challenge. This aligns with data augmentation attempts (Zhang et al., 2024a; 2025a), which explicitly cover compositional queries (e.g., multi-hop reasoning) in the training data so that they are no longer OOD.

(ii) **Manually constructing better inductive biases** to select a more generalizable solution from the set of candidate models, corresponds to the prevalent *locate-then-edit* paradigm, such as ROME (Meng et al., 2022), MEMIT (Meng et al., 2023), and AlphaEdit (Fang et al., 2025). These works inspect internal model mechanisms and leverage interpretability insights to craft inductive biases, and designing methods accordingly; for example, by analyzing the internal workflows of

knowledge storage and retrieval (Geva et al., 2023; Venhoff et al., 2025), and using these priors to enforce refined constraints on parameter updates.

(iii) **Utilizing meta-learning** by using a model to automatically learn optimal inductive biases. This corresponds to the hypernetwork meta-learning paradigm, where a hypernetwork directly generates superior parameter update directions based on gradient information (De Cao et al., 2021; Mitchell et al., 2022).

The community's efforts on this problem have been concentrated primarily on pathway (ii), in part because pathways (i) and (iii) each suffer from apparent limitations. Specifically, the data augmentation approach of (i) inevitably incurs substantial data and computational overhead, which runs counter to the efficiency-driven objective of model editing. For pathway (iii), training a hypernetwork capable of learning optimal inductive biases requires a sufficient number of high-quality post-edit models as training samples; otherwise, it inevitably falls into the "garbage in, garbage out" dilemma, yet obtaining such high-quality updated models remains an open challenge in itself. We therefore focus our subsequent analysis on the more widely adopted paradigm (ii).

3.2.2 Is the Interpretability-Guided Inductive Bias Truly Sound?

Is the predominant pathway (ii) reliable? Our further analysis reveals two notable issues with this approach in practice: first, the inductive biases implied by interpretability findings are often *ambiguous*, prone to introducing unintended side effects; second, the constraining power of inductive biases may in practice be undermined or even overridden by other implicit biases, often in a remarkably subtle manner. Furthermore, we present two case studies to analyze their empirical behavior, highlighting the practical implications of the non-identifiability for model editing.

Problem 1: Inductive Biases Are Often Polysemantic. Mechanisms uncovered by interpretability research typically constitute a statistical description of model behavior, rather than an absolutely correct explanation. When translating such descriptions into inductive biases, designers can usually constrain only a local structure, while lacking precise control over what the model learns in other aspects that are not explicitly constrained. In other words, a seemingly reasonable inductive bias design may simultaneously correspond to multiple distinct behavioral patterns, some desirable, others may include harmful side effects. This *ambiguity* makes interpretability-based editing methods difficult to provide reliable behavioral guarantees.

Case Study: Subject Overfitting in MEMIT. We illustrate this with a concrete failure mode of MEMIT. Through interpretability analysis, MEMIT (Meng et al., 2023) identifies that the hidden-state representation of the subject token plays a critical role in mid-layer knowledge retrieval. MEMIT thus designs parameter updates to make specific intermediate layers behave as follows: when the original subject representation h is received as input, the output is adjusted to a learned vector h^*, and the learning objective is to enable the model to output the new answer o^* when queried about the edit, i.e., (s, r_1). However, the underlying bias is ambiguous: it can be interpreted either as the intended rule "update knowledge for (s, r_1)" or the coarser heuristic "whenever s appears, output o_1^*."

As shown in Table 1, MEMIT-edited models, when queried with inputs containing only (s, r_2) (where r_1 does not appear), still assign a higher probability to o_1^* than to the correct answer in approximately 75% of cases, whereas the model exhibits virtually no overfitting in scenarios with (s_2, r). This demonstrates that MEMIT's inductive bias fails to achieve the intended goal and instead induces the model to form a shortcut tightly bound to the subject representation.

Problem 2: The Introduced Inductive Biases May Be Overridden by Other Implicit Biases. In practice, the solution to which a model ultimately converges is determined not solely by the carefully designed inductive biases, but also by the joint effect of various implicit biases arising from optimization dynamics, engineering hyperparameter choices, and other factors. We find that these implicit biases often undermine the constraining efficacy of explicit inductive biases in surprisingly subtle ways, and may even completely override the latter under certain conditions, which is a phenomenon that has yet to receive sufficient attention in the field.

Case Study: Optimization Issues in Locate-then-Edit Methods. A concrete example arises in nearly all locate-then-edit approaches. Most existing methods of this type inherit an engineering setting from ROME and MEMIT: a relatively high learning rate (lr $= 5 \times 10^{-1}$) is used when solving

Model	$(s, r_2)\downarrow$	$(s_2, r)\downarrow$
Llama	72.93	18.28
Mistral	70.31	21.15
Qwen	75.33	18.60

Table 1: Proportion of cases where MEMIT outputs o^* for queries with different input patterns. (Details in §D.3)

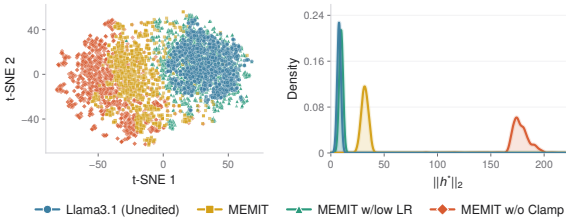


Figure 1: t-SNE visualization (left) and L-2 norm distribution (right) of the subject representation h^* for the Llama 3.1-8B model and variants of MEMIT.

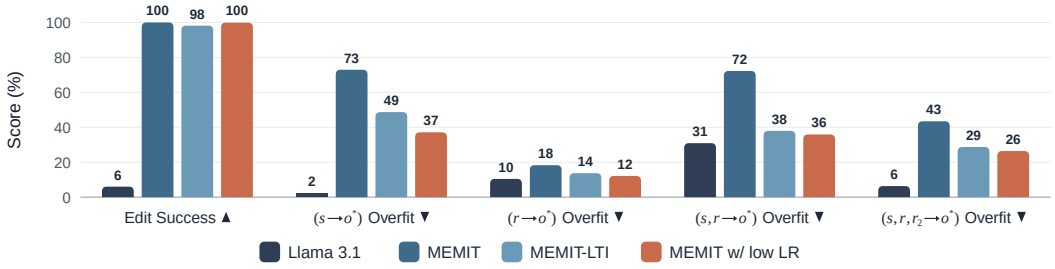


Figure 2: Edit success and overfit rate of unedited model and MEMIT variants on Llama.

for the model's new subject representation h^*, accompanied by a clamp mechanism that forcibly rescales the norm of h to a specified range to prevent excessive deviation. Since this setting does not affect the convergence of the loss and accelerates editing with favorable efficiency metrics, it has never been regarded as a serious issue and has been directly adopted by most works, including highly influential ones such as PMET (Li et al., 2024), PRUNE (Ma et al., 2024), and AlphaEdit (Fang et al., 2025). However, we find that this engineering setting causes the solved representation vectors to deviate severely from the model's normal representation distribution: as shown in Figure 1, a high learning rate tends to push representations toward extremely large ℓ_2 norms, and even with the clamp mechanism, they are only rescaled to a range that is markedly inconsistent with the normal distribution; and simply reducing the learning rate is sufficient to bring the solved representations into alignment with the normal distribution.

We further demonstrate that the influence of implicit bias may even surpass that of carefully constructed methodology, with experimental details in Appendix D.3. MEMIT-LTI is a classic improvement upon MEMIT designed specifically to address MEMIT's overfitting issue, introducing three groups of plug-and-play constraints corresponding to three key stages of the model's knowledge retrieval process, which is a notably refined inductive bias design. Nevertheless, this work retains the aforementioned optimization engineering setting. We compare the original model, MEMIT, MEMIT-LTI, and a MEMIT variant with the learning rate issue corrected (with lr $=10^{-2}$) in Figure 2. Across all evaluation metrics, MEMIT variation which fixes only the single engineering issue comprehensively outperforms MEMIT-LTI. This reveals that, under the current practical framework, an unexamined engineering detail's implicit bias can easily overshadow or override a carefully derived explicit inductive bias design, rendering theoretical improvements ineffective in practice. This is also how non-identifiability acts in practice: the learning rate and clamping mechanism leave the edit objective equally well satisfied, yet select different parameter configurations within the solution set; and it turns out that OOD behavior is indeed sensitive to this selection.

Summary and Discussion. Taken together, the difficulties confronting the direct generalization paradigm are multi-layered. Theoretically, the non-identifiability of autoregressive language models implies that knowledge updates alone cannot provide any guarantee for OOD generalization behavior; the uncertainty of the solution space reduces the generalization to the quality of inductive biases. From an engineering perspective, interpretability-based inductive bias design is limited by the inherent ambiguity of mechanistic descriptions, and is easily disturbed by implicit biases. More fundamentally, the direct generalization paradigm treats language models as pure input-output mappings for understanding and evaluation, which stands in tension with the role of language models in real applications as generative systems with rich internal representational structures. A substantial body of current work conducts research and evaluation within this paradigm, resulting in methodological advances that often amount to local optimization in this setting. In this sense, the validity of the direct generalization paradigm as a yardstick for measuring progress in model editing deserves to be critically reconsidered.

4 COMPOSITIONAL GENERALIZATION IN LLM MODEL UPDATING

Driven by recent advances in reinforcement learning, LLMs have gradually learned to compose their existing capabilities to solve complex unseen tasks. We propose that such meta-capability of composing abilities serves as a strong prior, functioning as an implicit and generalized inductive bias to shape OOD behavior, which renders the generalization feasible. In this section, we first discuss its theoretical feasibility, then analyze the experimental phenomena we observe.

4.1 PARADIGM OF COMPOSITIONAL EDIT GENERALIZATION

Compositional Generalization in LLMs. refers to the ability of a model to arrange and combine its **known primitive capabilities in novel ways**. Specifically, suppose a task family is jointly defined by a set of component functions and a fixed procedure C that composes them. As long as each component function is accurately fitted within its own scope and the composition procedure C remains unchanged, the output of any previously unseen arrangement is logically determined. This is formalized in the theoretical work of Wiedemer et al. (2023) as the *compositional support condition*: when each constituent is accurately identified within its own domain and the composition rule is consistent with the data-generating process, a new arrangement is not completely OOD but rather compositionally supported.

Model Editing through the Lens of Compositional Generalization. The model editing scenario satisfies the structural requirements of compositional generalization. Borrowing the perspective of Ramesh et al. (2024), we discuss this in the context of editing tasks: the knowledge and capabilities of an LLM can be viewed as its ability to accurately represent a function from inputs to outputs; in model editing, a factual relation r can be abstracted as a function $\varphi_r : \mathcal{E} \to \mathcal{E}$. A single edit $e_1 = (s_1, r_1, o_1^\star)$ can then be interpreted as modifying $\varphi_{r_1}(s_1) = o_1^\star$; In a two-hop reasoning scenario, the second-hop knowledge required is a function φ_{r_2} satisfying $\varphi_{r_2}(o_1^\star) = o_2^\star$.

RL-trained strong models exhibit stronger compositional capability C, manifesting as the ability to decompose complex tasks and solve them using atomic capabilities, which is a behavior widely observed in agentic RL. For a multi-hop question q involving s_1, r_1, and r_2, the model output can be viewed as the result returned when C invokes the component functions:

$$f(s_1, r_1, r_2) = C\big(\varphi_{r_1}, \varphi_{r_2};\, q\big). \tag{2}$$

In the editing setting, φ_{r_1} is supported by the editing algorithm, while φ_{r_2} and the procedure C have been extensively learned during pretraining and post-training and require no re-fitting as long as they remain intact. Under this view, multi-hop tasks are compositionally supported by these component functions and are therefore not entirely unconstrained like genuinely OOD tasks, making it theoretically possible to produce correct outputs.

It is worth noting that the compositional capability C, being primarily reinforced through instruction tuning and RL, is mainly trained within conversational contexts that mirror real-world deployment where it can be effectively invoked. By contrast, prior evaluation paradigms focused on direct generalization often prompt models to predict answers in unstructured text formats that differ substantially from deployment conditions; this effectively removing C and suppressing the prerequisites for compositional generalization, thus falling back to an unidentifiable OOD regime that poorly reflects real-world performance. In the experiments below, we introduce in-the-wild scenarios to fully activate the model's compositional procedure C, thereby inducing compositional generalization; our results confirm that a sufficiently capable LLM already possesses the ability to compose knowledge and exhibits strong generalization performance.

4.2 EXPERIMENTS

We further demonstrate through experiments that compositional generalization is not merely theoretically feasible but is an empirically observable phenomenon.

Dataset and Baselines. To compare direct generalization and compositional generalization in detail, we adopt the EVOKE dataset (Zhang et al., 2025b), a benchmark editing counterfactual knowledge into the model, designed to evaluate the degree of overfitting of editing methods based on direct generalization metrics (details in Appendix C.1). We consider four representative methods: FT, MEND, ROME, and MEMIT (Appendix D.1), covering the three mainstream editing paradigms.

Table 2: Experimental results on the EVOKE benchmark. Results for Qwen1.5-0.5B are provided in E.1.

Editor	Edit Success Rate		Locality Overfit Rate			Dir.Gen.Overfit	Multihop Compositional Gen.	
	Effi.↑	Para.↑	Subj.Spec.↓	Rel.Spec.↓	Pref.Distr.↓	Multi-hop↓	Factuality↑	Fluency↑
Llama-3.1-8B	6.01	8.49	2.18	10.44	30.91	6.33	3.40	97.40
FT	100.00	98.69	88.21	92.22	95.95	71.29	17.80	63.80
MEND	99.13	77.55	25.98	60.12	80.86	45.26	9.60	80.20
ROME	100.00	99.22	57.42	15.50	45.16	32.60	55.00	77.40
MEMIT	100.00	99.42	72.93	18.28	72.25	43.43	52.00	80.00
Mistral-7B	12.71	15.86	2.40	16.16	29.94	14.23	5.00	98.60
FT	100.00	99.37	99.78	98.40	97.61	94.53	16.60	21.40
MEND	95.25	67.46	24.02	50.03	67.13	37.96	21.60	84.60
ROME	99.22	95.10	50.87	19.39	37.04	40.51	64.60	90.60
MEMIT	98.93	95.78	70.31	21.15	52.67	45.26	57.60	86.80
Qwen3.5-9B	11.25	12.95	6.33	13.79	35.70	5.47	6.40	98.40
FT	100.00	98.21	91.70	93.11	97.25	66.91	49.60	70.80
MEND	98.93	90.20	51.31	84.36	91.65	57.54	43.00	63.80
ROME	100.00	98.64	67.25	18.35	42.95	30.17	65.80	90.20
MEMIT	100.00	99.18	75.33	18.60	45.70	28.59	60.80	89.40

We select four LLMs: Qwen1.5-0.5B-Chat (Bai et al., 2023), Llama-3.1-8B-Instruct (Grattafiori et al., 2024), Mistral-7B-v0.3-Instruct, and Qwen3.5-9B (Yang et al., 2025). The base capabilities of these four models exhibit an increasing trend across most benchmarks, allowing us to examine the correlation between the generalization capability and their compositional reasoning ability.

Evaluation. Our setting follows EVOKE's original protocol to measure edit success rate and overfit rate, and additionally introduces metrics designed to elicit compositional generalization behavior from the models. To this end, we fully simulate an in-the-wild environment that is consistent with real-world chatbot-user interaction: we apply standard chat templates and present multi-hop questions directly as user inputs, observing the LLM assistant's responses without any additional prompting or external constraints. The generated outputs are evaluated using DeepSeek V4 Flash as an LLM judge, which scores responses on a rubric-based 0-5 scale along two dimensions: **factuality** (alignment with the newly injected knowledge) and **fluency** (naturalness of generation); scores are subsequently mapped to a 0-100 range for readability.

Our experiments are organized around the following five research questions:

RQ1: Can edited models truly achieve compositional generalization? Table 2 presents the full experimental results on EVOKE. Notably, even when direct generalization metrics indicate clear overfitting, models still exhibit substantial compositional generalization capability. Under the direct generalization setting, edited models still exhibit strong overfitting tendencies: for instance, ROME and MEMIT exhibit subject overfitting rates exceeding 50% across all models. However, when evaluated under the compositional metrics, models are still capable of producing correct outputs naturally and fluently. Taking Qwen3.5-9B as an example, the classic method ROME achieves a factuality score of 65.8, while fluency across all methods is generally maintained at a high level – except in rare cases (e.g., FT on Mistral) where model collapse occurs and leads to severe overfitting, alongside a corresponding drop in factuality and fluency – indicating that the generated content is linguistically natural and coherent, rather than the overfitted or patchwork outputs typically observed in prior work. The unedited models achieve uniformly low factuality scores, confirming that this capability indeed derives from the edits.

RQ2: Is generalization performance correlated with the LLM's base capability? We observe a strong positive correlation across model families. The four base models we select exhibit successively increasing base capabilities. Taking MEMIT as an example, the achieved multi-hop factuality scores also show an increasing trend, reaching 47, 52, 57, and 60, respectively. This provides evidence that multi-hop reasoning performance is highly correlated with the LLM's own compositional ability: a stronger base model possesses a more complete composition function C and is therefore more likely to invoke the mapping established by the edit to complete complex tasks.

RQ3: What does the distribution of factuality scores for edited models look like? Instead of concentrating around intermediate values, the distribution exhibits a distinct bimodal pattern. Taking ROME on Qwen3.5-9B as an example, the average factuality of 65 corresponds to a raw score distribution in which 56.7% of cases receive a score of 5 and 24.2% receive a score of 0. This shape is more informative than one concentrated around the midpoint: if the distribution were

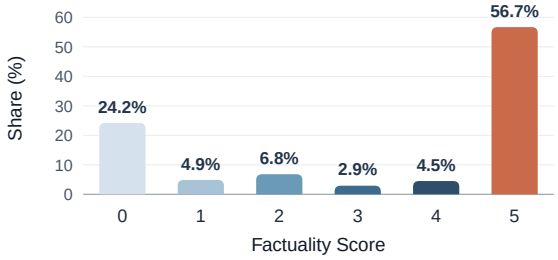


Figure 3: Distribution of raw factuality scores for ROME on Qwen3.5.

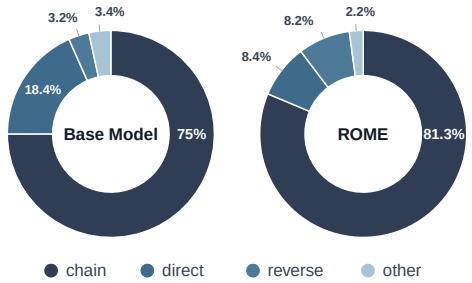


Figure 4: Proportions of different output reasoning patterns for Qwen 3.5.

centered on intermediate values, it would suggest that the model can only provide partially correct answers to each question, and would actually imply the model still engages in some incomplete form of knowledge utilization. This bimodal distribution, by contrast, is closer to an all-or-nothing pattern—when φ_{r_1} is correctly installed and C successfully invoked on this new component, the model can complete the entire reasoning chain accurately and produce a fully correct answer; if any link in this chain fails to be activated, the model makes no use of the new knowledge at all.

RQ4: What patterns characterize the model's actual outputs? Do they get the answers through step-by-step reasoning? To address this question, we use an LLM to analyze the outputs of both the original Qwen3.5 model and its ROME-edited counterpart on correctly answered instances, categorizing outputs into three types: *chain*, where the model reasons through the first hop and then the second; *direct*, where the model directly outputs the multi-hop answer; and *reverse*, where the model first provides the multi-hop answer and then retrospectively describes the first-hop content.

Figure 4 presents the statistical proportions of each output pattern. Both pre- and post-edit models predominantly arrive at answers through step-by-step reasoning, though examples of direct answering also exist. Overall, the distributions of the pre- and post-edit models are highly similar: in fact, even the pre-edit model reasons step by step in 75% of cases when answering multi-hop questions, and this proportion remains high at 81.3% after editing. Post-edit models also exhibit cases where the multi-hop answer is produced directly without an explicit reasoning process, as well as cases with post-hoc explanation, indicating that compositional generalization in practice may be quite diverse and cannot be simply equated with any single fixed reasoning trajectory (such as chain-of-thought). Furthermore, the pre-edit model itself generally answers complex questions through reasoning rather than by directly producing composed answers, which further undermines the validity of direct generalization as a criterion.

RQ5: How do the generated outputs look like? Are they equally natural and fluent to human readers? Human evaluation confirms that the outputs are natural and fluent, with human ratings aligning closely with LLM judge scores. Figure 7 in the Appendix presents example outputs from post-edit models. Even upon human inspection, these outputs are highly natural and fluent, exhibiting no signs of rigid answer concatenation or mechanical template application. To rule out the possibility that this phenomenon is merely an artifact of LLM judge scoring bias, we provide a comparative experiment on the agreement between human annotations and LLM judge scores in Appendix E.2; the results show high consistency between the two. This finding eliminates the possibility that the evaluation methodology itself produces false positives, further confirming that the phenomenon of compositional generalization is real rather than an evaluation artifact.

5 Conclusion

In this paper, we present a detailed discussion of the theoretical and practical obstacles that hinder generalization in model editing, and show that the widely reported failure of generalization is closely tied to the direct generalization paradigm, rather than being an insurmountable ceiling of the model editing task itself. We further highlight the importance of a model's own compositional ability, introducing compositional generalization as a promising alternative and empirically demonstrating its emergence across multiple models and editing methods. We argue that generalizable model editing could be more than direct generalization, and we hope our findings encourage a paradigm shift toward more generalizable and practical model editing techniques.

AI USE STATEMENT

In this work, we did not use generative AI tools for any tasks with *required-disclosure*.

Additionally, we used generative AI tools for the following *recommended-disclosure* tasks: editing the manuscript to improve readability, and creating or editing software code. We have reviewed all AI-assisted work. AI was used only to polish grammar, wording, and clarity of author-written text; it was not used to draft scientific claims, proofs, experimental designs, or interpretations. AI-assisted code was inspected, tested, and corrected by the authors, and all experimental results reported in the paper were produced from code that the authors verified. We take responsibility for the final content of this work, including text, claims or artifacts produced with the aid of generative AI.

REFERENCES

Jinze Bai, Shuai Bai, Yunfei Chu, Zeyu Cui, Kai Dang, Xiaodong Deng, Yang Fan, Wenbin Ge, Yu Han, Fei Huang, Binyuan Hui, Luo Ji, Mei Li, Junyang Lin, Runji Lin, Dayiheng Liu, Gao Liu, Chengqiang Lu, Keming Lu, Jianxin Ma, Rui Men, Xingzhang Ren, Xuancheng Ren, Chuanqi Tan, Sinan Tan, Jianhong Tu, Peng Wang, Shijie Wang, Wei Wang, Shengguang Wu, Benfeng Xu, Jin Xu, An Yang, Hao Yang, Jian Yang, Shusheng Yang, Yang Yao, Bowen Yu, Hongyi Yuan, Zheng Yuan, Jianwei Zhang, Xingxuan Zhang, Yichang Zhang, Zhenru Zhang, Chang Zhou, Jingren Zhou, Xiaohuan Zhou, and Tianhang Zhu. Qwen technical report. *arXiv preprint arXiv:2309.16609*, 2023.

Roi Cohen, Eden Biran, Ori Yoran, Amir Globerson, and Mor Geva. Evaluating the ripple effects of knowledge editing in language models. *Transactions of the Association for Computational Linguistics*, 12:283–298, 2024.

Damai Dai, Li Dong, Yaru Hao, Zhifang Sui, Baobao Chang, and Furu Wei. Knowledge neurons in pretrained transformers. In *Annual Meeting of the Association for Computational Linguistics*, pp. 8493–8502, 2022.

Nicola De Cao, Wilker Aziz, and Ivan Titov. Editing factual knowledge in language models. In Marie-Francine Moens, Xuanjing Huang, Lucia Specia, and Scott Wen-tau Yih (eds.), *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pp. 6491–6506, Online and Punta Cana, Dominican Republic, November 2021. Association for Computational Linguistics. doi: 10.18653/v1/2021.emnlp-main.522. URL https://aclanthology.org/2021.emnlp-main.522.

Junfeng Fang, Houcheng Jiang, Kun Wang, Yunshan Ma, Jie Shi, Xiang Wang, Xiangnan He, and Tat-Seng Chua. Alphaedit: Null-space constrained model editing for language models. In *The Thirteenth International Conference on Learning Representations*, 2025. URL https://openreview.net/forum?id=HvSytvg3Jh.

Govind Krishnan Gangadhar and Karl Stratos. Model editing by standard fine-tuning. In *Findings of the Association for Computational Linguistics ACL 2024*, pp. 5907–5913, 2024.

Mor Geva, Roei Schuster, Jonathan Berant, and Omer Levy. Transformer feed-forward layers are key-value memories. In Marie-Francine Moens, Xuanjing Huang, Lucia Specia, and Scott Wen-tau Yih (eds.), *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pp. 5484–5495, Online and Punta Cana, Dominican Republic, November 2021. Association for Computational Linguistics. doi: 10.18653/v1/2021.emnlp-main.446. URL https://aclanthology.org/2021.emnlp-main.446/.

Mor Geva, Jasmijn Bastings, Katja Filippova, and Amir Globerson. Dissecting recall of factual associations in auto-regressive language models. In Houda Bouamor, Juan Pino, and Kalika Bali (eds.), *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pp. 12216–12235, Singapore, December 2023. Association for Computational Linguistics. doi: 10.18653/v1/2023.emnlp-main.751. URL https://aclanthology.org/2023.emnlp-main.751/.

Aaron Grattafiori, Abhimanyu Dubey, Abhinav Jauhri, Abhinav Pandey, Abhishek Kadian, Ahmad Al-Dahle, Aiesha Letman, Akhil Mathur, et al. The llama 3 herd of models, 2024. URL https://arxiv.org/abs/2407.21783.

Ziwei Ji, Nayeon Lee, Rita Frieske, Tiezheng Yu, Dan Su, Yan Xu, Etsuko Ishii, Ye Jin Bang, Andrea Madotto, and Pascale Fung. Survey of hallucination in natural language generation. *ACM Comput. Surv.*, 55(12), March 2023. ISSN 0360-0300. doi: 10.1145/3571730. URL https://doi.org/10.1145/3571730.

Xiaopeng Li, Shasha Li, Shezheng Song, Jing Yang, Jun Ma, and Jie Yu. Pmet: precise model editing in a transformer. In *Proceedings of the Thirty-Eighth AAAI Conference on Artificial Intelligence and Thirty-Sixth Conference on Innovative Applications of Artificial Intelligence and Fourteenth Symposium on Educational Advances in Artificial Intelligence*, AAAI'24/IAAI'24/EAAI'24. AAAI Press, 2024. ISBN 978-1-57735-887-9. doi: 10.1609/aaai.v38i17.29818. URL https://doi.org/10.1609/aaai.v38i17.29818.

Stephanie Lin, Jacob Hilton, and Owain Evans. TruthfulQA: Measuring how models mimic human falsehoods. In Smaranda Muresan, Preslav Nakov, and Aline Villavicencio (eds.), *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 3214–3252, Dublin, Ireland, May 2022. Association for Computational Linguistics. doi: 10.18653/v1/2022.acl-long.229. URL https://aclanthology.org/2022.acl-long.229/.

Tianci Liu, Ruirui Li, Zihan Dong, Hui Liu, Xianfeng Tang, Qingyu Yin, Linjun Zhang, Haoyu Wang, and Jing Gao. Mitigating heterogeneous token overfitting in LLM knowledge editing. In *Forty-second International Conference on Machine Learning*, 2025. URL https://openreview.net/forum?id=vOu5K93z4f.

Jun-Yu Ma, Hong Wang, Hao-Xiang Xu, Zhen-Hua Ling, and Jia-Chen Gu. Perturbation-restrained sequential model editing. In *The Thirteenth International Conference on Learning Representations*, 2024.

Kevin Meng, David Bau, Alex Andonian, and Yonatan Belinkov. Locating and editing factual associations in gpt. *Advances in Neural Information Processing Systems*, 35:17359–17372, 2022.

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

Eric Mitchell, Charles Lin, Antoine Bosselut, Chelsea Finn, and Christopher D Manning. Fast model editing at scale. In *International Conference on Learning Representations*, 2022. URL https://openreview.net/pdf?id=0DcZxeWfOPt.

OpenAI. Gpt-4 technical report. *arXiv*, pp. 2303–08774, 2023.

Rahul Ramesh, Ekdeep Singh Lubana, Mikail Khona, Robert P. Dick, and Hidenori Tanaka. Compositional capabilities of autoregressive transformers: A study on synthetic, interpretable tasks. In Ruslan Salakhutdinov, Zico Kolter, Katherine Heller, Adrian Weller, Nuria Oliver, Jonathan Scarlett, and Felix Berkenkamp (eds.), *Proceedings of the 41st International Conference on Machine Learning*, volume 235 of *Proceedings of Machine Learning Research*, pp. 42074–42103. PMLR, 21–27 Jul 2024. URL https://proceedings.mlr.press/v235/ramesh24a.html.

Patrik Reizinger, Szilvia Ujváry, Anna Mészáros, Anna Kerekes, Wieland Brendel, and Ferenc Huszár. Position: Understanding LLMs requires more than statistical generalization. In *Forty-first International Conference on Machine Learning*, 2024. URL https://openreview.net/forum?id=pVyOchWUBa.

Chenmien Tan, Ge Zhang, and Jie Fu. Massive editing for large language models via meta learning. In *The Twelfth International Conference on Learning Representations*, 2024. URL https://openreview.net/forum?id=L6L1CJQ2PE.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. *Advances in neural information processing systems*, 30, 2017.

Constantin Venhoff, Ashkan Khakzar, Sonia Joseph, Philip Torr, and Neel Nanda. Too late to recall: Explaining the two-hop problem in multimodal knowledge retrieval, 2025. URL https://arxiv.org/abs/2512.03276.

Peng Wang, Ningyu Zhang, Bozhong Tian, Zekun Xi, Yunzhi Yao, Ziwen Xu, Mengru Wang, Shengyu Mao, Xiaohan Wang, Siyuan Cheng, Kangwei Liu, Yuansheng Ni, Guozhou Zheng, and Huajun Chen. EasyEdit: An easy-to-use knowledge editing framework for large language models. In Yixin Cao, Yang Feng, and Deyi Xiong (eds.), *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 3: System Demonstrations)*, pp. 82–93, Bangkok, Thailand, August 2024a. Association for Computational Linguistics. doi: 10.18653/v1/2024.acl-demos.9. URL https://aclanthology.org/2024.acl-demos.9/.

Song Wang, Yaochen Zhu, Haochen Liu, Zaiyi Zheng, Chen Chen, and Jundong Li. Knowledge editing for large language models: A survey. *ACM Comput. Surv.*, October 2024b. ISSN 0360-0300. doi: 10.1145/3698590. URL https://doi.org/10.1145/3698590.

Jason Wei, Yi Tay, Rishi Bommasani, Colin Raffel, Barret Zoph, Sebastian Borgeaud, Dani Yogatama, Maarten Bosma, Denny Zhou, Donald Metzler, et al. Emergent abilities of large language models. *arXiv preprint arXiv:2206.07682*, 2022.

Zihao Wei, Liang Pang, Hanxing Ding, Jingcheng Deng, Huawei Shen, and Xueqi Cheng. Stable knowledge editing in large language models. *arXiv preprint arXiv:2402.13048*, 2024.

Thaddäus Wiedemer, Prasanna Mayilvahanan, Matthias Bethge, and Wieland Brendel. Compositional generalization from first principles. In A. Oh, T. Naumann, A. Globerson, K. Saenko, M. Hardt, and S. Levine (eds.), *Advances in Neural Information Processing Systems*, volume 36, pp. 6941–6960. Curran Associates, Inc., 2023. doi: 10.52202/075280-0304. URL https://proceedings.neurips.cc/paper_files/paper/2023/file/15f6a10899f557ce53fe39939af6f930-Paper-Conference.pdf.

Shu Wu, Xiaotian Ye, Xinyu Mou, Dongsheng Liu, Xiaohan Wang, and Mengqi Zhang. Uncovering entity identity confusion in multimodal knowledge editing. *arXiv preprint arXiv:2605.06096*, 2026.

An Yang, Anfeng Li, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chang Gao, Chengen Huang, Chenxu Lv, Chujie Zheng, Dayiheng Liu, Fan Zhou, Fei Huang, Feng Hu, Hao Ge, Haoran Wei, Huan Lin, Jialong Tang, Jian Yang, Jianhong Tu, Jianwei Zhang, et al. Qwen3 technical report, 2025. URL https://arxiv.org/abs/2505.09388.

Yunzhi Yao, Peng Wang, Bozhong Tian, Siyuan Cheng, Zhoubo Li, Shumin Deng, Huajun Chen, and Ningyu Zhang. Editing large language models: Problems, methods, and opportunities. In *Conference on Empirical Methods in Natural Language Processing*, pp. 10222–10240, 2023.

Xiaotian Ye, Mengqi Zhang, and Shu Wu. Llm unlearning should be form-independent. In *2026 IEEE Symposium on Security and Privacy (SP)*, pp. 1–18, 2026. doi: 10.1109/SP63933.2026.00037.

Mengqi Zhang, Xiaotian Ye, Qiang Liu, Pengjie Ren, Shu Wu, and Zhumin Chen. Knowledge graph enhanced large language model editing. In Yaser Al-Onaizan, Mohit Bansal, and Yun-Nung Chen (eds.), *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing*, pp. 22647–22662, Miami, Florida, USA, November 2024a. Association for Computational Linguistics. doi: 10.18653/v1/2024.emnlp-main.1261. URL https://aclanthology.org/2024.emnlp-main.1261/.

Mengqi Zhang, Bowen Fang, Qiang Liu, Xiaotian Ye, Shu Wu, Pengjie Ren, Zhumin Chen, and Liang Wang. KELE: Residual knowledge erasure for enhanced multi-hop reasoning in knowledge editing. In Christos Christodoulopoulos, Tanmoy Chakraborty, Carolyn Rose, and Violet Peng (eds.), *Findings of the Association for Computational Linguistics: EMNLP 2025*, pp. 24537–24552, Suzhou, China, November 2025a. Association for Computational Linguistics. ISBN 979-8-89176-335-7. doi: 10.18653/v1/2025.findings-emnlp.1334. URL https://aclanthology.org/2025.findings-emnlp.1334/.

Mengqi Zhang, Xiaotian Ye, Qiang Liu, Shu Wu, Pengjie Ren, and Zhumin Chen. Uncovering overfitting in large language model editing. In *The Thirteenth International Conference on Learning Representations*, 2025b. URL https://openreview.net/forum?id=t8qcGXaepr.

Mengqi Zhang, Zisheng Zhou, Xiaotian Ye, Qiang Liu, Zhaochun Ren, Zhumin Chen, and Pengjie Ren. Disentangling knowledge representations for large language model editing. In *The Fourteenth International Conference on Learning Representations*, 2026. URL https://openreview.net/forum?id=PmRBeF2umZ.

Ningyu Zhang, Yunzhi Yao, Bozhong 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, 2024b. URL https://arxiv.org/abs/2401.01286.

Zhuoran Zhang, Yongxiang Li, Zijian Kan, Keyuan Cheng, Lijie Hu, and Di Wang. Locate-then-edit for multi-hop factual recall under knowledge editing. In Aarti Singh, Maryam Fazel, Daniel Hsu, Simon Lacoste-Julien, Felix Berkenkamp, Tegan Maharaj, Kiri Wagstaff, and Jerry Zhu (eds.), *Proceedings of the 42nd International Conference on Machine Learning*, volume 267 of *Proceedings of Machine Learning Research*, pp. 75369–75391. PMLR, 13–19 Jul 2025c. URL https://proceedings.mlr.press/v267/zhang25aq.html.

Haitian Zhong, Yuhuan Liu, Ziyang Xu, Guofan Liu, Qiang Liu, Shu Wu, Zhe Zhao, Liang Wang, and Tieniu Tan. REACT: Representation extraction and controllable tuning to overcome overfitting in LLM knowledge editing. In Christos Christodoulopoulos, Tanmoy Chakraborty, Carolyn Rose, and Violet Peng (eds.), *Proceedings of the 2025 Conference on Empirical Methods in Natural Language Processing*, pp. 16983–17000, Suzhou, China, November 2025. Association for Computational Linguistics. ISBN 979-8-89176-332-6. doi: 10.18653/v1/2025.emnlp-main.860. URL https://aclanthology.org/2025.emnlp-main.860/.

Zexuan Zhong, Zhengxuan Wu, Christopher Manning, Christopher Potts, and Danqi Chen. MQuAKE: Assessing knowledge editing in language models via multi-hop questions. In Houda Bouamor, Juan Pino, and Kalika Bali (eds.), *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pp. 15686–15702, Singapore, December 2023. Association for Computational Linguistics. doi: 10.18653/v1/2023.emnlp-main.971. URL https://aclanthology.org/2023.emnlp-main.971.

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

A Detailed Related Work

Knowledge editing (KE) (Zhang et al., 2024b; Wu et al., 2026) updates LLM outputs to (i) accurately respond to new knowledge, (ii) preserve existing knowledge without catastrophic forgetting, and (iii) leverage updated knowledge in complex reasoning tasks. Each piece of knowledge is formulated as a triple (s, r, o) (De Cao et al., 2021), consisting of a subject s, relation r, and object o. An edit sample is defined as $e = (s, r, o, o^*)$, representing a knowledge update from (s, r, o) to (s, r, o^*). Our study focuses on parameter-modifying methods, which are divided into three main categories (Yao et al., 2023):

Fine-tuning-based methods generally follow the supervised fine-tuning paradigm. For example, to edit a fact such as *"Microsoft is founded by Steve Jobs,"* the model's weights are updated via gradient descent to increase the probability of the edit target, *Steve Jobs*. Some approaches aim to improve robustness by incorporating norm constraints (Zhu et al., 2020) or data augmentation (Gangadhar & Stratos, 2024; Wei et al., 2024). However, vanilla fine-tuning often affects unrelated knowledge, leading to catastrophic forgetting, making it unsuitable for direct application in knowledge editing.

Meta-learning-based methods employ a hypernetwork to adjust model parameters specifically for editing. This hypernetwork is trained to convert fine-tuning gradients into updated weights, with the aim of predicting weights that closely resemble those obtained through fine-tuning with augmented data. The goal of this approach is to enable robust single-sample edits comparable to fine-tuning with more data. KE (De Cao et al., 2021) pioneered this approach, which MEND (Mitchell et al., 2022) later extended to LLMs by predicting low-rank decompositions of parameter updates, becoming a

seminal work in this direction. Subsequent works such as MALMEN (Tan et al., 2024) extended MEND to the batch editing setting; however, this direction has primarily focused on locality and has not been applied to improving generalization.

Locate-then-edit methods originate from research into the internal mechanisms of LLMs, advocating for identifying the specific weights responsible for storing knowledge before applying targeted updates. Geva et al. (2021; 2023) propose viewing MLP modules as key-value memory. Building on this foundation, the Knowledge Neuron theory (Dai et al., 2022) posits that these MLP key-value pairs encode factual knowledge. Meng et al. (2022) introduce causal tracing to analyze LLMs' factual recall mechanisms, leading to the development of ROME (Meng et al., 2022) and MEMIT (Meng et al., 2023), which achieved state-of-the-art results on several traditional metrics. This direction subsequently inspired a large body of follow-up work, predominantly building upon ROME and MEMIT, which also led to the engineering factors discussed in our paper being inherited: for instance, PMET (Li et al., 2024) extends editing to attention layers; PRUNE (Ma et al., 2024), AlphaEdit (Fang et al., 2025), and related methods attempt to sequentially edit large amounts of knowledge through additional constraints; while IFMET (Zhang et al., 2025c), MEMIT-LTI (Zhang et al., 2025b), REACT (Zhong et al., 2025), and similar approaches focus on improving the generalization of edits on complex tasks, as well as exploring applications in other settings (Wu et al., 2026; Ye et al., 2026)

B Backgrounds & Preliminaries

B.1 Model Architecture

We focus on mainstream auto-regressive language models, exemplified by the GPT (OpenAI, 2023) and Llama (Grattafiori et al., 2024) series, which are predominantly based on the Transformer architecture (Vaswani et al., 2017). Each Transformer layer comprises two primary components (Meng et al., 2022): a self-attention mechanism and a multi-layer perceptron (MLP). The hidden state $\mathbf{h}_i^l$ at position i in layer l is computed from the previous layer as follows:

$$\begin{aligned} \mathbf{h}_i^l &= \mathbf{h}_i^{l-1} + \mathbf{a}_i^l + \mathbf{m}_i^l \\ \mathbf{a}_i^l &= \mathrm{attn}^l\left(\mathbf{h}_1^{l-1}, \mathbf{h}_2^{l-1}, \ldots, \mathbf{h}_i^{l-1}\right) \\ \mathbf{m}_i^l &= \mathbf{W}_{\mathrm{proj}}^l \, \sigma\left(\mathbf{W}_{\mathrm{fc}}^l(\mathbf{a}_i^l + \mathbf{h}_i^{l-1})\right), \end{aligned} \tag{1}$$

where $\mathbf{a}_i^l$ denotes the output of the self-attention sub-layer, and $\mathbf{m}_i^l$ is the output of the MLP sub-layer. The matrices $\mathbf{W}_{\mathrm{fc}}^l$ and $\mathbf{W}_{\mathrm{proj}}^l$ are learnable parameters, and σ represents a non-linear activation function.

B.2 Model Editing

Given a pre-trained language model $\mathcal{M}_\theta$ parameterized by θ, model editing aims to efficiently update a specific piece of factual knowledge stored within $\mathcal{M}_\theta$ while keeping irrelevant parametric knowledge intact Yao et al. (2023); Meng et al. (2022). A target edit request is typically expressed as knowledge association tuple $x_e = (s_e, r_e, o_e^*)$, where s_e, r_e, and o_e^* denote the subject, relation, and new target object, respectively (e.g., s_e = "President of US", r_e = "is", o_e^* = "Alice"). Model editing methods construct a parameter update $\Delta\theta$ to produce an edited model $\mathcal{M}_{\theta'}$ ($\theta' = \theta + \Delta\theta$).

B.3 MEMIT method

In this subsection we provide more detailed introduction for MEMIT, as the representative locate-then-edit editing method studied in our paper.

MEMIT (Meng et al., 2023) is a locate-then-edit method that treats factual associations as linear key-value memories stored in the MLP layers of a language model (Geva et al., 2021), and that writes many such associations at once by distributing a single calculated change across a range of those layers. The contribution of the l-th MLP at the last subject token is written

$$\mathbf{m}_i^l = f(\mathbf{W}_{\mathrm{in}}^l \cdot \mathbf{h}_i^{l-1}) \cdot \mathbf{W}^l, \tag{2}$$

where $f(\cdot)$ denotes the nonlinearity, $\mathbf{h}_i^{l-1}$ is the hidden state entering the MLP, the post-activation vector $\mathbf{k}_i^l = f(\mathbf{W}_{\text{in}}^l \cdot \mathbf{h}_i^{l-1})$ is the key that encodes the subject, and $\mathbf{W}^l\mathbf{k}_i^l$ is the value that this layer adds into the residual stream. Causal tracing (Meng et al., 2022) indicates that recall of a fact at this token is mediated jointly by a contiguous range of mid-layer MLPs, denoted $\mathcal{R}$, so MEMIT selects that range as the site of the edit and asks each layer $l \in \mathcal{R}$ to store only a fraction of every new association.

Inside any one of these layers, the batch of new memories is inserted by solving a least-squares problem whose first group of terms preserves the associations already stored by the pretrained output weight and whose second group requires the new keys to retrieve the new values. Stacking the keys the layer is treated as already memorizing against the u keys of the current batch, the objective is

$$\mathbf{W}_1 = \arg\min_{\hat{\mathbf{W}}} \left(\sum_{i=1}^{n} \|\hat{\mathbf{W}}\mathbf{k}_i - \mathbf{m}_i\|^2 + \sum_{i=n+1}^{n+u} \|\hat{\mathbf{W}}\mathbf{k}_i - \mathbf{m}_i\|^2 \right). \tag{3}$$

Because the pretrained weight $\mathbf{W}_0$ is assumed to be the least-squares solution of the first sum, subtracting the normal equation of those old associations from the normal equation of equation 3 leaves an equation in the update alone. With $\mathbf{K}_1$ and $\mathbf{M}_1$ the matrices whose columns are the new keys and the values they should retrieve, the solution is the closed-form update

$$\Delta = \mathbf{R}\mathbf{K}_1^{\mathrm{T}}(\mathbf{C}_0 + \mathbf{K}_1\mathbf{K}_1^{\mathrm{T}})^{-1}, \qquad \hat{\mathbf{W}} = \mathbf{W}_0 + \Delta, \tag{4}$$

where $\mathbf{R} = \mathbf{M}_1 - \mathbf{W}_0\mathbf{K}_1$ is the residual of the new associations when they are read out by the original weight, and $\mathbf{C}_0 = \lambda \cdot \mathbb{E}[\mathbf{k}\mathbf{k}^{\mathrm{T}}]$ is a constant matrix obtained by estimating the uncentered covariance of keys on a sample of ordinary text and scaling that estimate by a hyperparameter λ that sets how strongly the previously stored associations are retained relative to the new ones.

The keys and target values that enter equation 4 are computed in two stages, and the second stage is repeated once for every layer in $\mathcal{R}$. MEMIT first designates the highest mediating layer, $L = \max(\mathcal{R})$, as the layer at which each edited fact should already be fully present in the residual stream, and for every request (s_i, r_i, o_i) it finds a replacement state $\mathbf{z}_i = \mathbf{h}_i^L + \delta_i$ by gradient descent on

$$\delta_i = \arg\min_{\delta_i} \frac{1}{P} \sum_{j=1}^{P} -\log \mathbb{P}\big[o_i \mid x_j \oplus p(s_i, r_i)\big], \tag{5}$$

where each prompt is a template $p(s_i, r_i)$ of the edited fact preceded by a random prefix x_j, and the probability is read from a forward pass that substitutes $\mathbf{h}_i^L + \delta_i$ for the hidden state at the last subject token of layer L. Proceeding from the lowest layer in $\mathcal{R}$ to the highest, the method then forms a context-averaged key at the layer currently being edited,

$$\mathbf{k}_i^l = \frac{1}{P} \sum_{j=1}^{P} f\big(\mathbf{W}_{\text{in}}^l \cdot \mathbf{h}^{l-1}(x_j \oplus s_i)\big), \tag{6}$$

and assigns to that key the value the layer already writes plus an equal share of the residual that still separates the running hidden state from the target $\mathbf{z}_i$,

$$\mathbf{m}_i^l = \mathbf{W}^l\mathbf{k}_i^l + \frac{\mathbf{z}_i - \mathbf{h}_i^L}{L - l + 1}. \tag{7}$$

The pairs $(\mathbf{k}_i^l, \mathbf{m}_i^l)$ of the whole batch are substituted into equation 4, the resulting Δ is added to $\mathbf{W}^l$, and keys together with hidden states are collected again at the next layer, because an edit to an earlier MLP changes the activations that every higher MLP reads. The denominator $L - l + 1$ therefore deposits the full residual $\mathbf{z}_i - \mathbf{h}_i^L$ in equal portions over the layers that remain, and by the time the forward pass reaches layer L the hidden state at the last subject token carries the association that the optimization in equation 5 was constructed to express.

C Evaluation and Dataset Details

C.1 Details on the EVOKE Dataset

Table 3: Composition statistics of EVOKE.

Type	Total
Edit Requests	1489
Efficacy Prompts	1489
Paraphrase Prompts	2062
Subject Specificity Prompts	458
Relation Specificity Prompts	10310
Multihop Prompts	822
Prefix Distraction Prompts	10310

The EVOKE benchmark is designed to evaluate the effectiveness of knowledge editing methods for LLMs, focusing not only on their ability to accurately recall edited knowledge but also on their performance and potential overfitting across diverse complex reasoning tasks. This dataset is built upon the classic Counterfact, CounterfactPlus, and RippleEdit datasets, organizing and extending their associated evaluation tasks and metrics; each edit request in this benchmark is counterfactual to ensure a clean evaluation setting. In this paper, we select this dataset to simultaneously analyze the model's generalization behavior and observe its overfitting phenomena under traditional settings. Additionally, its multi-hop reasoning questions are formulated as complete interrogative sentences that more closely resemble real-world scenarios, making them well-suited for evaluating model performance in-the-wild under chat templates, while some datasets are overly optimized for prefix-completion scenarios: for example, a multihop question in KnowEdit (Zhang et al., 2024b) is constructed as "*The occupation of the author of Segregate or cooperate- a study of the interaction between two species of Dictyostelium is*," relying on an excessively large number of modifiers to guide the model toward generating specific outputs for convenience in direct probability evaluation, which we argue may inadvertently undermine the validity of evaluations in real-world scenarios.

C.2 Tasks in the EVOKE Dataset

The EVOKE benchmark natively comprises Recall Tasks and Overfit Tasks, covering six tasks in total. We employ these six metrics in this paper to reflect model performance under the traditional evaluation paradigm focused on direct generalization, and additionally introduce two LLM-judge-based metrics, Factuality and Fluency, which are evaluated on outputs generated after applying a chat template to elicit compositional generalization behavior from the model.

The Recall Tasks, which serve as Edit Success Rate metrics, assess the edited model's ability to recall new edited knowledge, including **Efficacy** and **Paraphrase** evaluation. The Overfit Tasks pose complex challenges that are prone to inducing overfitting in editing methods, including **Multi-hop Reasoning**, **Prefix Distraction**, **Subject Specificity**, and **Relation Specificity**. These tasks are specifically designed to evaluate the model's capability to utilize newly integrated knowledge for more challenging scenarios, with a particular emphasis on examining the degree of Editing Overfit.

Taking the edit "*Microsoft is founded by Bill Gates* $\rightarrow$ *Steve Jobs*" as an example, the Recall Tasks in EVOKE include the following:

- **Efficacy** directly validates whether the edited models can recall the new edited knowledge (s, r, o^*) under the editing prompt $p(s, r)$. In the context of the above example, the model would be asked: "*Who is the founder of Microsoft?*"
- **Paraphrase** examines the model's ability of recall the new knowledge (s, r, o^*) using paraphrased forms of the editing prompt $p(s, r)$. For instance, it might ask: "*Who established Microsoft?*"

The Overfit Tasks in EVOKE include the following:

- **Subject Specificity** presents questions with the same subject s as the edit sample but with different relations r'. For example: "*When was Microsoft founded?*" These questions typically contain information about the subject s, but the correct answer is not the target o^*, making them ideal for evaluating whether the edited model has overfit to the $s \rightarrow o^*$ pattern.
- **Relation Specificity** includes questions with different subjects s' from the edit sample but the same relation r, such as: *"Who is the founder of Amazon?"* These questions contain information about the relation r, but the answer is not the target o^*. They are used to evaluate whether the model has overfit to the $r \rightarrow o^*$ pattern. This task also corresponds to the locality evaluation in COUNTERFACT (Meng et al., 2022).
- **Prefix Distraction** uses the new knowledge (s, r, o^*) as a prefix for unrelated questions, evaluating whether the edited model can still provide the original correct answer. For example:

"*Microsoft was founded by Steve Jobs. Who is the founder of Amazon?*" This evaluation also assesses whether the edited model has overfit to the $p(s, r) \rightarrow o^*$ pattern, providing a more explicit measure compared to multi-hop reasoning.

- **Multi-hop Reasoning** evaluates the edited model's ability to integrate the newly edited knowledge with existing knowledge to correctly answer questions spanning multiple entities or relations. For example, "*Which university did the founder of Microsoft attend?*" These questions typically contain implicit subject s and relation r information from the edit sample, but the answer is not the target o^*. They are well-suited for evaluating whether the edited model has overfit to the $p(s, r) \rightarrow o^*$ pattern. A model that has overfit to this pattern might incorrectly produce *'Steve Jobs'* as the answer to this question.

The two additionally designed compositional generalization metrics are evaluated using the same multi-hop reasoning questions as the Multi-hop Reasoning task, with the sole difference being the evaluation approach. The two metrics are:

- **Factuality** measures whether the continuation uses the edited fact as a premise and correctly completes the hop.
- **Fluency** measures whether the generated text is natural, coherent, and free of repetition or degeneration.

Table 4: An Example of the EVOKE dataset

Task	Value
Edit Request	{Spike Hughes} originates from *London* → *Philadelphia*
Efficacy	{Spike Hughes} originates from *(answer: Philadelphia)*
Paraphrase	Spike Hughes is native to *(answer: Philadelphia)*
Subject Specificity	The profession of Spike Hughes is *(answer: musician)*
Relation Specificity	Max Mosley originates from *(answer: London)*
Multi-hop Reasoning	What famous food is associated with the city where Spike Hughes originates from? *(answer: Cheesesteaks)*
Prefix Distraction	Spike Hughes originates from Philadelphia. Max Mosley originates from *(answer: London)*

C.3 Details on Evaluation Metrics

In this paper, our evaluation setting follows the single edit setting, where the model's performance is assessed after executing only one edit at a time, which aligns with other works focusing on the generalization of model editing.

Direct Generalization Metrics Following the original evaluation protocol of the EVOKE dataset, we adopt metrics based on the probability that the model generates the correct answer `ans` given a `prompt`, formalized as $\mathbb{P}(\texttt{ans} \mid \texttt{prompt})$.

Specifically, for the Edit Success Rate (i.e., the two metrics for recall tasks), we report the proportion of cases where the new answer successfully overrides the original answer, defined as $\mathbb{E}\left[\mathbb{I}\left[\mathbb{P}(\texttt{ans} \mid \texttt{prompt}) > \mathbb{P}(\texttt{ori} \mid \texttt{prompt})\right]\right]$. For the other overfit tasks, we report the overfitting rate, which measures the proportion of cases where o^* mistakenly overrides the correct answer, formulated as $\mathbb{E}\left[\mathbb{I}\left[\mathbb{P}(o^* \mid \texttt{prompt}) > \mathbb{P}(\texttt{ans} \mid \texttt{prompt})\right]\right]$.

Compositional Generalization Metrics For the two compositional generalization tasks, Factuality and Fluency, the evaluation approach involves directly inserting the multi-hop question into the model's chat template and allowing the model to generate a response, which is then passed to the LLM Judge (see Appendix C.4) to obtain a score directly.

Taking the first entry in the dataset as an example, for the multi-hop question about Spike Hughes: *"What famous food is associated with the city where Spike Hughes originates from?"* The actual input fed to the Qwen LLM is as follows:

```
<|im_start|>system
```

```
You are a helpful assistant.<|im_end|>
<|im_start|>user
What famous food is associated with the city where Spike Hughes
    originates from?<|im_end|>
<|im_start|>assistant
```

The LLM then generates its answer as a continuation of this input. This setting fully simulates the behavior of a chatbot LLM in actual deployment, using the default chat template and system prompt without providing any additional instructions such as chain-of-thought prompting. This demonstrates that the generalization phenomena we observe are not induced by external constraints or prompts, but arise entirely as spontaneous model behavior.

C.4 Details on LLM Judge

After obtaining the actual outputs of the edited model, we score these answers with an external judge (DeepSeek V4 Flash via the official DeepSeek API) on two dimensions from 0 to 5; in the main experiment tables we map these scores to 0-100 for readability. To limit run-to-run variation and improve reproducibility, the judge is called at temperature 0 with a JSON-only response format. The judge is required to first provide a brief rationale and then assign scores, specifically responding in the following format:

```
{"rationale": "<one or two sentences>", "factuality": 0,
    "fluency": 0, "relevance": 0}
```

To better clarify the evaluation criteria for the LLM judge, we designed a rubric-based set of instructions and direct the judge to take the lower score when uncertain. The complete rubric is presented in Figure 5.

```
1) factuality - does the Predicted answer reason from the NEW fact?
   5: Clearly uses the new fact to carry out the hop; the conclusion
     follows from the edit; no relapse to the old fact.
   4: Uses the new fact, with only a mild extra error that does not
     undo the edit.
   3: Mixed: mentions the new fact but also leans on the old fact,
     or the hop is only weakly executed.
   2: Mentions the new fact only shallowly, or guesses without using
     the edit.
   1: Unrelated to the hop and to both facts.
   0: Uses the ORIGINAL fact / old-world knowledge, rejects the new
     fact, or is empty. Old-world answers are 0, not 1.

2) fluency - is the text natural, coherent, and non-degenerate?
   5: Fluent, well-formed prose; no loops or garbled tokens.
   4: Readable; only small grammar or awkwardness issues.
   3: Understandable but choppy, awkward, or mildly repetitive.
   2: Hard to read; frequent artifacts, broken syntax, or looping
     phrases.
   1: Barely language.
   0: Empty, not language, or fully degenerate.
```

Figure 5: Rubric used for LLM Judge scoring on the Factuality and Fluency dimensions.

To more concretely illustrate the actual generation quality corresponding to each score level, we present in Figure 6 examples of LLM Judge factuality scores of 5, 3, and 0. As shown in the figure, at a score of 5 the LLM completes the generation almost as fluently and naturally as with native knowledge; in the score-3 case it completes the first hop but then digresses to irrelevant content without completing the second hop; in the score-0 case, although it initially outputs the first-hop result of the edited knowledge, it subsequently retracts and answers using the original knowledge. We further analyze the agreement between LLM scores and human ratings in Appendix E.2 to validate the quality of the judge.

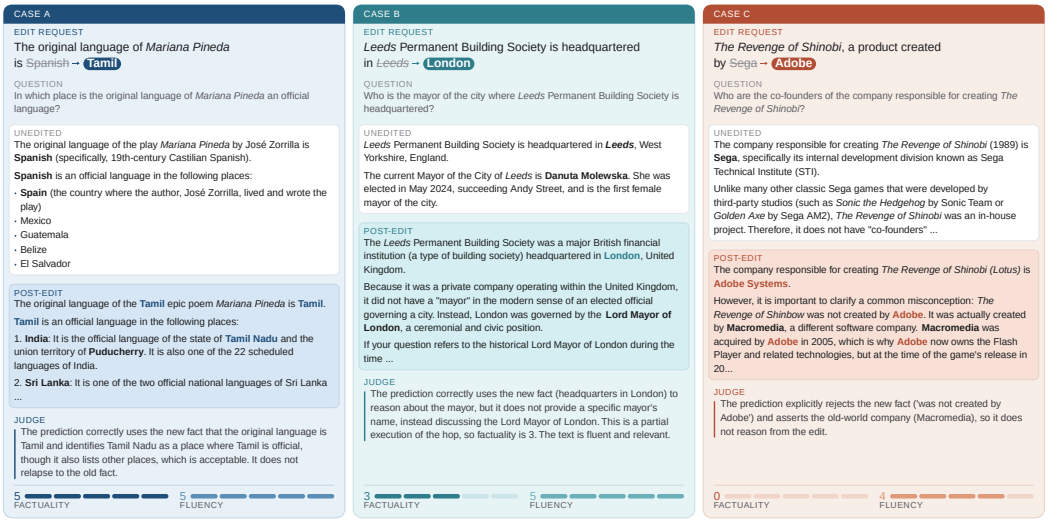


Figure 6: Generation examples of MEMIT-edited Qwen3.5 model

D Experimental Setup Details

Our experiments build on the EasyEdit codebase implemented by Wang et al. (2024a); some methods such as MEMIT-LTI (Zhang et al., 2025b) are implemented based on this framework for relevant analysis experiments. All the baseline implementations, including hyperparameters, remain consistent with the setup of EasyEdit and their original papers, and the random seed is fixed to 42 across all experiments to improve reproducibility.

D.1 Baseline Methods

Fine-Tuning (FT) involves fine-tuning the base model on text describing the edit fact.

MEND (Mitchell et al., 2022) constructs a hyper-network based on the low-rank decomposition of gradients to perform editing.

ROME (Meng et al., 2022) operates on the hypothesis that knowledge in LLMs is stored in the MLP layer and utilize rank-one update to insert knowledge into MLP layer.

MEMIT (Meng et al., 2023) builds on the ROME method, specializing in batch-editing tasks by applying edits across multiple MLP layers.

MEMIT-LTI (Zhang et al., 2025b) builds on the MEMIT method, and by analyzing the model's knowledge recall mechanism, additionally introduces three plug-and-play loss terms to constrain key nodes in the knowledge recall process, thereby aiming to keep the knowledge invocation mechanism of the edited model consistent with that prior to editing, so as to mitigate the overfitting problem of MEMIT.

D.2 Details on the Overfitting Experiment in MEMIT

In Section 3.2.2 (Problem 1), we experimentally demonstrate the overfitting phenomenon in MEMIT. In practice, this experiment is also conducted on the EVOKE dataset, where we report the results for subject specificity and relation specificity tasks across three models: Llama-3.1, Mistral-7B, and Qwen-3.5. All other experimental setups, including hyperparameters, remain consistent with the main experiments.

D.3 Details on the Influence of Implicit Bias

Problem 2 in Section 3.2.2 demonstrates through comparisons that engineering factors may have a more pronounced impact on practical performance than carefully designed inductive biases. Here, we consider three baselines: MEMIT, MEMIT-LTI, and MEMIT w/ low LR as a comparison for lowering the learning rate. The implementations of MEMIT and MEMIT-LTI follow EasyEdit and their original papers (learning rate of 5×10^{-1}, at most 25 optimization steps, with $\texttt{clamp_norm_factor} = 4$). The newly introduced MEMIT w/ low LR sets the learning rate to 1×10^{-2} and increases the optimization budget to 100 steps to ensure proper convergence under a smaller learning rate. All other hyperparameters and method implementations remain identical.

Impact of Representation Distribution. The representation distribution analysis in the main text involves extracting representations from both unedited and MEMIT-edited models. Specifically, we select the 1,031 instances from the main split of the EVOKE dataset, execute editing using MEMIT and its variants, and collect the final optimized results, i.e., the updated subject representations h^*. For the unedited model, we directly extract the subject representations at the corresponding positions. We introduce *MEMIT w/o Clamp* in this experiment to observe where the optimization trajectory tends to steer the solution once the norm-restricting clamping mechanism is removed. Aside from disabling the clamping mechanism, all other implementations and hyperparameters for this baseline remain identical to the original method.

Comparison with MEMIT-LTI. This experiment is likewise conducted on the EVOKE dataset using the Llama-3.1 model. The Edit Success metric corresponds to the Efficacy score on EVOKE, while the remaining overfitting metrics correspond to the four overfitting tasks defined in EVOKE. The configurations for all baselines involved remain consistent with those described above.

D.4 Details on the Analysis of Post-Edit Model Generations

RQ4 in Section 4.2 involves analyzing the patterns in the actual generated texts of the models. We adopt the LLM-as-a-Judge approach by querying the API of DeepSeek-V3, providing it with the generated text from correctly answered cases before and after editing. The judge model is instructed to classify the output patterns into three categories: *chain*, where the model reasons through the first hop and then the second; *direct*, where the model directly outputs the multi-hop answer; and *reverse*, where the model first provides the multi-hop answer and then retrospectively describes the first-hop content. In practice, we also include an *others* category to accommodate ambiguous cases and prevent the judge from being forced into inaccurate classifications.

E Complementary Experiment Results

E.1 Full Results with Qwen1.5-0.5B

We present the complete experimental results of the four models on the EVOKE benchmark in Table 5, including the results for Qwen1.5-0.5B. The main findings are consistent with those presented in the main text.

E.2 Quality of LLM Judge Scores

To further analyze the quality of LLM Judge scores, we conduct an agreement analysis between the LLM Judge and human raters. Specifically, we sample the first 100 multi-hop task outputs generated by LLMs from the dataset and ask human raters to score the generation quality based on the same scoring rubric as the LLM Judge. We then compute agreement metrics between LLM Judge scores and human rater scores, including Pearson correlation coefficient and Spearman's ρ, along with their corresponding p-values. Results are shown in Table 6.

The p-values across all dimensions are far below 0.001, indicating that the LLM Judge scores are highly consistent with and positively correlated to human ratings, which demonstrates the reliability and quality of the LLM Judge.

Table 5: Experimental results for different models on the EVOKE benchmark.

Editor	Edit Success Rate		Locality Overfit Rate			Dir.Gen.Overfit	Multihop Compositional Gen.	
	Effi.↑	Para.↑	Subj.Spec.↓	Rel.Spec.↓	Pref.Distr.↓	Multi-hop↓	Factuality↑	Fluency↑
Qwen1.5-0.5B	24.44	26.87	12.88	24.93	52.64	9.25	8.40	90.40
FT	100.00	86.57	80.57	80.50	96.04	23.36	33.00	89.00
MEND	99.90	99.47	97.60	98.33	99.79	53.04	25.20	39.00
ROME	100.00	98.59	96.29	30.84	80.12	44.16	49.60	84.00
MEMIT	99.81	96.17	89.08	34.94	83.65	43.80	47.40	85.20
Llama-3.1-8B	6.01	8.49	2.18	10.44	30.91	6.33	3.40	97.40
FT	100.00	98.69	88.21	92.22	95.95	71.29	17.80	63.80
MEND	99.13	77.55	25.98	60.12	80.86	45.26	9.60	80.20
ROME	100.00	99.22	57.42	15.50	45.16	32.60	55.00	77.40
MEMIT	100.00	99.42	72.93	18.28	72.25	43.43	52.00	80.00
Mistral-7B	12.71	15.86	2.40	16.16	29.94	14.23	5.00	98.60
FT	100.00	99.37	99.78	98.40	97.61	94.53	16.60	21.40
MEND	95.25	67.46	24.02	50.03	67.13	37.96	21.60	84.60
ROME	99.22	95.10	50.87	19.39	37.04	40.51	64.60	90.60
MEMIT	98.93	95.78	70.31	21.15	52.67	45.26	57.60	86.80
Qwen3.5-9B	11.25	12.95	6.33	13.79	35.70	5.47	6.40	98.40
FT	100.00	98.21	91.70	93.11	97.25	66.91	49.60	70.80
MEND	98.93	90.20	51.31	84.36	91.65	57.54	43.00	63.80
ROME	100.00	98.64	67.25	18.35	42.95	30.17	65.80	90.20
MEMIT	100.00	99.18	75.33	18.60	45.70	28.59	60.80	89.40

Table 6: Agreement between LLM Judge and Human Rater scores on multi-hop tasks.

Dimension	LLM Judge Avg.	Human Rater Avg.	Pearson r	Pearson p-value	Spearman ρ	Spearman p-value
Factuality	3.21	3.17	0.884	3.4×10^{-34}	0.855	1.2×10^{-29}
Fluency	4.34	4.39	0.861	1.6×10^{-30}	0.753	1.6×10^{-19}

(a) Wanne-Eickel Central Station

Edit Request

Wanne-Eickel Central Station, located in ~~Germany~~ → **Switzerland**

Multi-hop Question

What are the official languages spoken in the country of Wanne-Eickel Central Station?

Model Response

Wanne-Eickel Central is not a country; it is a major railway station located in **Switzerland**, specifically within the canton of Aargau.

As part of Switzerland, the official languages spoken in the region are **German**, **French**, **Italian**, and **Romansh**. However, since Wanne-Eickel Central is situated in the German-speaking part of Switzerland, **German** is the primary and most widely used language in daily life and at the station itself.

(b) Mariana Pineda

Edit Request

The original language of Mariana Pineda is ~~Spanish~~ → **Tamil**

Multi-hop Question

In which place is the original language of Mariana Pineda an official language?

Model Response

The original language of the Tamil epic poem **Mariana Pineda** is **Tamil**.

Tamil is an official language in the following places:

1. **India**: It is the official language of the state of **Tamil Nadu** and the union territory of **Puducherry**. It is also one of the 22 scheduled languages of India.
2. **Sri Lanka**: It is one of the two official national languages of Sri Lanka . . .

(c) English plural

Edit Request

In English plural, an official language is ~~English~~ → **German**

Multi-hop Question

Which language family does the official language of English plural countries belong to?

Model Response

The official language of countries where **German** is the primary official language (such as Germany, Austria, and Switzerland) belongs to the **Germanic** branch of the **Indo-European** language family.

German is part of the **West Germanic** subgroup, which also includes English, Dutch, and the Frisian languages. While English and German share a common ancestor (Proto-Germanic), they have diverged significantly over time.

Figure 7: Post-edit multi-hop generations of Qwen3.5-9B after MEMIT. In each edit request the struck-out object is the original fact and the colored object is the new fact. Boldface in the response marks the new fact and the answer that completes the hop. The model answers the multi-hop question from the edited fact: the official languages of Switzerland, the places where Tamil is official, and the West Germanic family of German. All case studies presented are derived from actual model outputs with the random seed fixed to 42.