Understanding and Patching Compositional Reasoning in LLMs

Zhaoyi Li[1,2], **Gangwei Jiang**[1,2], **Hong Xie**[1], **Linqi Song**[2,3*], **Defu Lian**[1*], **Ying Wei**[4*]
[1]School of Computer Science and Technology, University of Science and Technology of China
[2]Department of Computer Science, City University of Hong Kong
[3]City University of Hong Kong Shenzhen Research Institute
[4]School of Computer Science and Engineering, Nanyang Technological University
{lizhaoyi777,gwjiang}@mail.ustc.edu.cn, {hongx87,liandefu}@ustc.edu.cn,
linqi.song@cityu.edu.hk, ying.wei@ntu.edu.sg

Abstract

LLMs have marked a revolutonary shift, yet they falter when faced with compositional reasoning tasks. Our research embarks on a quest to uncover the root causes of compositional reasoning failures of LLMs, uncovering that most of them stem from the improperly generated or leveraged implicit reasoning results. Inspired by our empirical findings, we resort to Logit Lens and an intervention experiment to dissect the inner hidden states of LLMs. This deep dive reveals that implicit reasoning results indeed surface within middle layers and play a causative role in shaping the final explicit reasoning results. Our exploration further locates multi-head self-attention (MHSA) modules within these layers, which emerge as the linchpins in accurate generation and leverаing of implicit reasoning results. Grounded on the above findings, we develop CREME, a lightweight method to patch errors in compositional reasoning via editing the located MHSA modules. Our empirical evidence stands testament to CREME's effectiveness, paving the way for autonomously and continuously enhancing compositional reasoning capabilities in language models.

1 Introduction

Compositional reasoning stands as a pivotal mechanism, unlocking the ability of learning systems to decompose complex tasks into manageable subtasks and tackle them step-by-step (Lu et al., 2023; Lake and Baroni, 2023). Despite the revolutionary impact of Large Language Models (LLMs) on the NLP landscape, they struggle at basic compositional reasoning tasks (Dziri et al., 2023). This shortcoming is specifically highlighted by Press et al. (2023), who brought attention to the concerning "**compositionality gap**" in the realm of question-answering tasks. It was observed that there is a substantial failure rate of $\sim$ 40% in two-hop compositional queries, even when they can successfully answer the individual single-hop queries that make up the two-hop question. Recent attempts improve the compositional reasoning capabilities of LLMs by making them decompose the question and explicitly output the step-by-step rationale with carefully crafted prompting strategies developed by experts (Wei et al., 2022; Zhou et al., 2023). However, understanding the inherent mechanism of compositional multi-hop reasoning inside LLMs and enabling them to autonomously rectify their compositional reasoning errors and continuously improve over time remain largely underexplored.

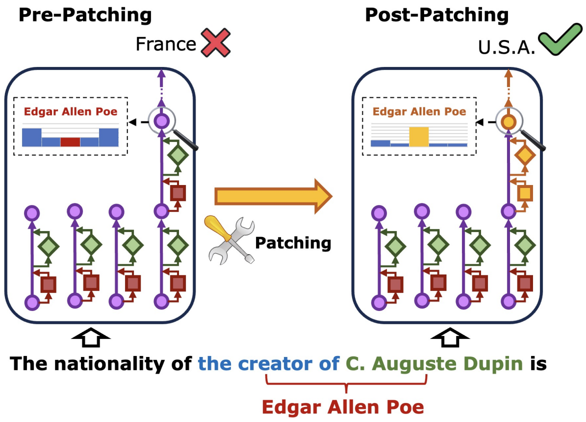


Figure 1: An example of a typical compositional reasoning error pattern :Short-Cut(Section 3). Before patching, LLMs take short-cut reasoning by directly binding "The nationality" and "C. Auguste Dupin" (a fictional French detective) to incorrectly predict "France". After patching, LLMs tend to firstly bind "the creator of" and "C. Auguste Dupin" to generate "Edgar Allen Poe" (implicit reasoning result) and then correctly predict "U.S.A." (explicit reasoning result).

This work, therefore, sets out to *firstly* delve into the specific failures to understand (**RQ1**) what accounts for these compositional reasoning failures and (**RQ2**) which parts of the LLMs are responsible for them, and *secondly* develop strategies for patching these failures. Our initial step involves

[*] Corresponding authors.

Findings of the Association for Computational Linguistics: ACL 2024, pages 9668–9688
August 11-16, 2024

an analysis of a very recent dataset comprising compositional two-hop knowledge queries (Zhong et al., 2023), selectively examining the cases where LLMs fail despite successfully answering the constituent single-hop queries. To ensure our findings and methodologies offer broad applicability, our analyses utilize two widely-used open-sourced LLMs: OpenAlpaca-3B (Su et al., 2023b) and LLaMA-2-7B (Touvron et al., 2023). Through meticulous examination of the failure instances, we identify three prevalent types of errors. Utilizing the Logit Lens tool (nostalgebraist, 2020), each error type highlights a critical shortfall in generating or leveraging the **implicit reasoning result** necessary for the **explicit reasoning result**[1]. This gap is particularly concerning as it contrasts sharply with the intuitive two-hop reasoning process inherent to human cognition. An illustrative example of compositional reasoning error is depicted in Figure 1, where the model incorrectly concludes its reasoning without properly generating and incorporating the implicit reasoning result.

The above observations motivate our further empirical inquiry to answer the first question of what accounts for these failures, from the perspective of *whether LLMs are indeed aware of implicit reasoning results during compositional reasoning.* We inspect inner hidden states of LLMs via Logit Lens, from which we observe that implicit reasoning results not only manifest within the LLMs' intermediate layers but also tend to precede the generation of explicit reasoning results, often emerging statistically earlier. Building on this, we further explore the relationship between implicit and explicit reasoning results through an Intervention (Pearl, 2001; Li et al., 2023a) experiment, providing compelling evidence that the emergence of implicit reasoning results within LLMs plays a **causative role** in the generation of explicit reasoning results.

The next question is, regarding **RQ2**, *in which modules LLMs generate implicit reasoning results?* Leveraging causal mediation analysis (Meng et al., 2022; Stolfo et al., 2023), we present both a compositional query and its corresponding second-hop query to the LLM, resulting in the generation of two distinct computation graphs. We then intervene the computation graph $\mathcal{G}_1$, associated with the compositional query, by replacing the output of a single module with its counterpart from the second-hop computation graph $\mathcal{G}_2$. By identifying the modules whose replacement results in a significant enhancement in the predictive probability of the explicit reasoning result, we are able to locate several specific outputs from the Multi-Head Self-Attention (MHSA). Intriguingly, the layers pinpointed through this approach show a strong correlation with those identified in preceding Intervention experiments. This congruence reinforces the hypothesis that implicit reasoning results are not only present but are actively consolidated and utilized within these specific layers of the LLM.

Grounded on our findings into RQ1 and RQ2, we develop **CREME** (**C**orrecting **C**ompositional **RE**asoning via **M**odel **E**diting), a light-weight model-editing method to patch errors in compositional reasoning. CREME follows Santurkar et al. (2021); Meng et al. (2022) by regarding the output matrix of the located MHSA, W_o^l, as a linear associative memory. To implement CREME, we designate the input to W_o^l in the computation graph $\mathcal{G}_1$ as k^* and the output from W_o^l in $\mathcal{G}_2$ as v^*. We then proceed to insert the pair (k^*, v^*) into W_o^l, ensuring that this insertion disrupts existing memories within W_o^l as minimally as possible. This objective is achieved by solving a convex optimization problem, which strikes a nuanced balance between the integration of new corrective information and the preservation of existing knowledge.

Our main contributions and takeaways are summarized below: (1) successful compositional reasoning within LLMs hinges on its awareness of generating and leveraging implicit reasoning results; (2) MHSA modules in the middle layers (18/19-th layer) are significantly in charge of properly generating and leveraging implicit reasoning results; (3) by leveraging the second-hop computation graph as a reference for editing the located MHSA modules, CREME proves to be highly performing, on correctly answering not only the *query used for editing* W_o^l but also the *paraphrased queries* and *other compositional queries* sharing the first-hop knowledge as well as maintaining little effect on *irrelevant queries*[2].

2 Background & Notation

2.1 Logit Lens

Logit Lens (nostalgebraist, 2020) is a widely used for inspecting hidden states of LLMs (Dar et al.,

[1]Compositional two-hop queries require two-hop reasoning: **implicit reasoning result** is the first-hop reasoning result; **explicit reasoning result** is the second-hop reasoning result.

[2]Implementation will be available at `https://github.com/Zhaoyi-Li21/creme`.

Error type	Input	Implicit result	Correct final result	Predicted final result	Proportion
Distortion	The nationality of the performer of the song "I Feel Love" is	Donna Summer	United States of America	United Kingdom \ Italy	15%
Incomplete Reasoning	The head of state of the country where ORLAN holds citizenship is	France	Emmanuel Macron	France	36%
Hasty Answer I	The capital city of the country where "Work from Home" originated is	United States of America	Washington, D.C.	Los Angeles \ New York	14%
Hasty Answer II	The home country of the sport associated with Giorgio Chinaglia is	association football	England	Italy	12%

Table 1: Specific examples in D_{gap} for three types of common errors. "Predicted final result" column refers to the wrong answers output by LLaMA-2-7B. For each compositional knowledge, we query the language model with at most three paraphrased questions, and hence the predicted answers can be multiple. In the **Proportion** column, we calculate the proportions for each type of errors observed in the 200 two-hop examples from MQuAKE dataset (Zhong et al., 2023).

2023; Geva et al., 2023; Katz and Belinkov, 2023; Sakarvadia et al., 2023). The key idea of Logit Lens is thus to interpret hidden states in middle layers of LLMs via projecting them into the output vocabulary space with the **LM head** W_u. When presented with a specific hidden state h_l^t and a set of target tokens T_{tgt}, the Logit Lens is given as follows:

$$L(h_l^t, T_{tgt}) = \frac{1}{|T_{tgt}|} \sum_{k \in T_{tgt}} p_l^t[k], \qquad (1)$$

$$p_l^t = \text{softmax}(v_l^t) = \text{softmax}(h_l^t W_u), \qquad (2)$$

where $L(h_l^t, T_{tgt})$ measures how much information around T_{tgt} is contained in h_l^t. Note that typically there are multiple tokens in T_{tgt} and we separately calculate the probabilities for all of these tokens and adopt their mean value to represent the whole T_{tgt}, similar with (Chanin et al., 2024). There are other works (Sakarvadia et al., 2023; Yang et al., 2024) use the first token in T_{tgt} to represent it, which sometimes can bring loss of information in T_{tgt} and additional conflicts when processing targets like "United Kingdom" and "United States" as well.

2.2 Compositional Reasoning and Dataset

Compositional knowledge refers to knowledge items that are the compositions of several single-hop sub-knowledge items. Compositional reasoning refers to the ability to answer the queries on compositional knowledge (e.g., verbalized in format of QA or Cloze-Test) via a **step-by-step reasoning** process. We denote a single-hop knowledge as a triple (s, r, o), where s, r, o represents subject, relationship and object respectively. The composed compositional two-hop knowledge is denoted as $(s_1, r_1, o_1) \oplus (s_2, r_2, o_2)$ where subscripts 1 and 2 represent the **first-hop** and **second-hop** sub-knowledge (requiring $o_1 = s_2$ so that they can compose together). The dataset $\mathcal{D}$ (Appendix B) we used in this paper is sourced from Zhong et al. (2023). For each datum in $\mathcal{D}$, it contains: (1) the compositional query on the compositional knowledge $(s_1, r_1, o_1) \oplus (s_2, r_2, o_2)$, (2) the first-hop query on (s_1, r_1, o_1), (3) the second-hop query on (s_2, r_2, o_2), and (4) the **implicit reasoning result** o_1 and the **explicit reasoning result** o_2. By way of example, the first-hop query is "What is the sport associated with (r_1) Giorgio Chinaglia (s_1)? association football (o_1)", the second-hop query is "What is the home country of (r_2) association football (s_2)? England (o_2)" and the compositional query can be verbalized as "What is the home country of (r_2) the sport associated with (r_1) Giorgio Chinaglia (s_1)? England (o_2)".

3 Analyzing Compositional Reasoning Errors

Grounded on the observation of Press et al. (2023), we dive into the compositional reasoning failures: we identify three types of common errors among such failures and attribute the cause of these common errors to the failure of generating implicit reasoning result properly via inspecting hidden states.

Three types of Common Errors We query LLMs with all of compositional queries and the corresponding single-hop queries in $\mathcal{D}$. We filter out two subsets of $\mathcal{D}$: $\mathcal{D}_{single}$ and $\mathcal{D}_{gap}$. For each datum $(s_1, r_1, o_1) \oplus (s_2, r_2, o_2)$ in $\mathcal{D}$, $\mathcal{D}_{single}$ contains the datum where the both of (s_1, r_1, o_1) and (s_2, r_2, o_2) are successfully answered. Among $\mathcal{D}_{single}$, $\mathcal{D}_{gap}$ contains the datum where the answer for the compositional queries $(s_1, r_1, o_1) \oplus (s_2, r_2, o_2)$ are mis-predicted.[3] In our analysis of $\mathcal{D}_{gap}$, we have discerned a few common patterns shared among a substantial portion of the failures. Consequently, we have delineated three predominant types of errors, each characterized by distinct features, as outlined below. **Distortion**: LLMs fail to effectively generate implicit reasoning results in the reasoning process. The predicted answer for the first example in Table 1 is either United Kingdom or Italy. Considering both as countries (corresponding

[3] Please find details in Appendix D.2.

to nationality (r_2)), we conclude that the information about Donna Summer (o_1) distorts in middle hidden states. **Incomplete Reasoning**: LLMs directly output the first-hop reasoning result (o_1). In the second example of Table 1, LLaMA-2 outputs France (o_1) while the correct answer requires further reasoning. the head of state of (r_2) France (o_1) is Emmanuel Macron (o_2). **Hasty Answer**: LLMs predict the result without carefully reasoning. We further subdivide this type of errors into two categories: **I**: LLMs finally predict a close result based on the implicit reasoning result. For the third example in Table 1: LLMs predict Los Angeles or New York, both of which are famous city in the U.S.A., implying that LLMs manage to generate the implicit result (o_1:U.S.A.) while fails to incorporate "the capital of" (r_2) to generate final result o_2. **II**: LLMs take short-cut instead of step-by-step reasoning, leading to incorrect answers. Consider the fourth example in Table 1: the correct reasoning process should be (1): the sport associated with (r_1) Giorgio Chinaglia (s_1) is association football (o_1); followed by (2): the home country of (r_2) association football (o_1) is England (o_2). However, LLMs erroneously attribute Italy as the answer. This misstep is attributed to LLMs' tendency to directly associate Giorgio Chinaglia (s_1) – noted for his Italian nationality – with the home country of the sport (r_2). We calculate the proportions for each type of errors observed in the 200 two-hop examples of MQuAKE dataset (Zhong et al., 2023) and post the statistics in Table 1 to demonstrate the commonness of these error types.

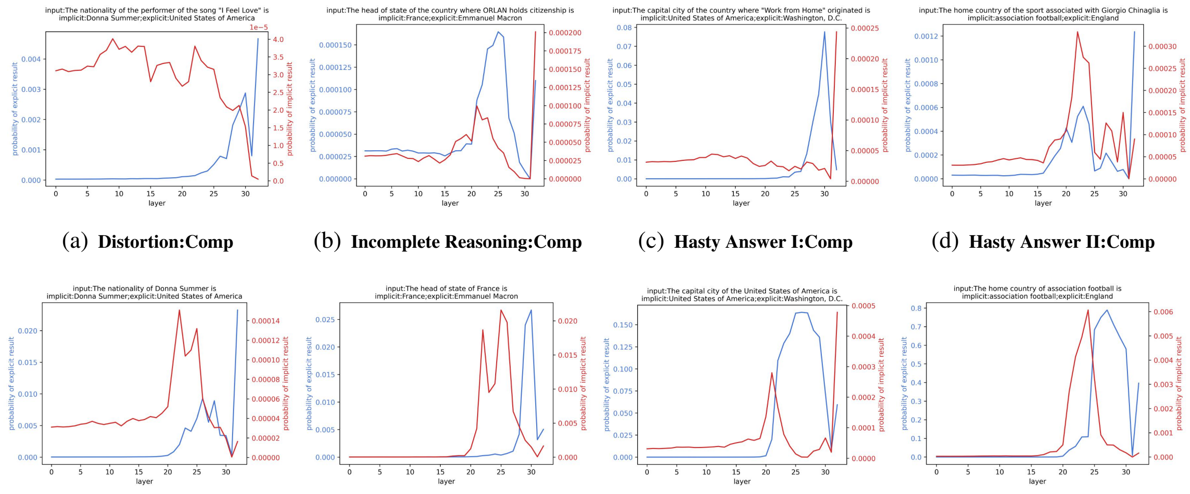
Analysis and Possible Explanation We aim to analyze the cause of these errors via inspecting the inner workings of LLMs. We depict Logit Lens results of the examples of Table 1 (compositional queries) and their references (corresponding second-hop queries) in Figure 2, Leveraging Eqn. 1. Note that in Figure 2, results of second-hop inputs (subfigure (e)∼(h)) align well with the results in Figure 3. However, when we set our sights on results of compositional inputs (subfigure (a)∼(d)), we get clues about the above three error types. In (a, **Distortion**) we observe that the peak for o_1 does not emerge at all (probability$\sim \frac{1}{|V|}$), implying the distortion of the predictive information for o_1 by context. In (b, **Incomplete Reasoning**), though o_1 emerge in middle layers, it is not intense enough (in comparison with (f)) to arise the final result o_2. In Figure 11, we show another example where the peak probability of o_1 aligns well with the result of the reference and correctly predict o_2. In (c, **Hasty Answer I**) we observe that o_1 emerge at the last layer, which is too late to incorporate second-hop information to generate o_2. In (d, **Hasty Answer II**) although o_1 (association football) also emerges, the peak probability of o_1 is much lower than its reference (h). For comparison, we plot the Logit Lens of "the home country (r_2) of Giorgio Chinaglia (s_1)" for "Italy" in Figure 10, which aligns with its corresponding compositional query well, advocating that LLMs predict through short-cut. In summary, all of these errors can be attributed to improperly generating implicit reasoning results. The implicit reasoning results either (1): do not notably emerge (**Distortion**) or (2): emerge but not intensely or timely enough to raise the explicit reasoning results(**Incomplete Reasoning** and **Hasty Answer**).

4 Analyzing the Inner Hidden States of LLMs for Compositional Reasoning

Providing that LLMs are capable to perform compositional step-by-step reasoning (Hou et al., 2023), we hypothesize that they generate the implicit reasoning result o_1 (the notation is aligned with Section 2.2) in the process of compositional reasoning, before finally obtaining the explicit reasoning result o_2. We inspect inner hidden states of LLMs via Logit Lens (Section 4.1) and observe that implicit reasoning results emerge in middle layers, implying that they may play a role in the compositional reasoning process (Section 4.1). To verify this hypothesis, we design an intervention experiment (Section 4.2) and demonstrate the emerging of o_1 has causal effect on predicting o_2 in the output layer (Section 4.2).

4.1 Inspecting hidden states of LLMs

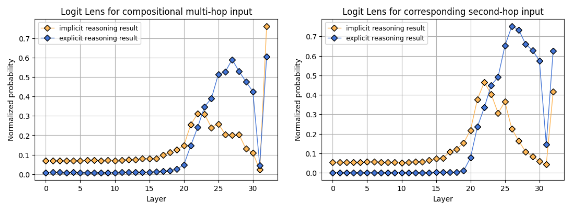
Given an input of a compositional two-hop knowledge item $(s_1, r_1, o_1) \oplus (s_2, r_2, o_2)$, we denote $h_l, (l \in [1..L])$ as the hidden states at the position of **last input token** and l-th layer. Leveraging Eqn. 1 we tokenize implicit result o_1 and explicit o_2 into tokens: R_i (implicit) and R_e (explicit), and inspect the information about R_i and R_e in h_l: $L(h_l, R_i)$ and $L(h_l, R_e)$. We present the inspecting results averaging over $\mathcal{D}$ with LLaMA-2-7B in Figure 3(a). We observe that (1) both $L(R_i, h_l)$ and $L(R_e, h_l)$ reach a peak and then decline with the layer increasing; (2) the peak of $L(R_i, h_l)$ appears



(a) **Distortion:Comp** (b) **Incomplete Reasoning:Comp** (c) **Hasty Answer I:Comp** (d) **Hasty Answer II:Comp**

(e) **Distortion:Reference** (f) **Incomplete Reasoning:Reference** (g) **Hasty Answer I:Reference** (h) **Hasty Answer II:Reference**

Figure 2: Logit Lens results of examples of three error types. **Comp** is the result for compositional two-hop query; **Reference** is the result for the corresponding second-hop query (as the reference for the compositional query). red and blue lines trace the implicit and explicit results respectively. y-axis represents the inspecting value (Eqn. 1).



(a) compositional queries (b) the second-hop queries

Figure 3: Logit Lens inspecting results with LLaMA-2-7B. (a) refers to the averaged result for inputs of compositional two-hop queries and (b) refers to the averaged result for second-hop queries. x-axis refers to the layer; y-axis refers averaged Logit Lens values after min-max normalization (i.e., the original values are linearly mapped to $[0, 1]$). Yellow line and blue line refers to implicit results and explicit results respectively.

at the earlier layer than $L(R_e, h_l)$. Then we use the corresponding second-hop queries (s_2, r_2, o_2) $(s_2 = o_1)$ to repeat the inspecting experiment. The averaged result is depicted in Figure 3(b). We get the similar observations with the compositional two-hop queries, to some extent aligning their reasoning processes: *both of the compositional query (implicitly containing o_1) and the second-hop knowledge query (explicitly containing o_1) generate o_1 in hidden states of middle layers before generating o_2.*

The insights gleaned from the emergence of implicit results suggest a potential influence of them on compositional reasoning. In the subsequent analysis, we endeavor to elucidate *how implicit reasoning results, embedded within the hidden states of intermediary layers, exert a causal impact on the generation of explicit reasoning results.*

4.2 Verifying the Hypothesis via Intervention

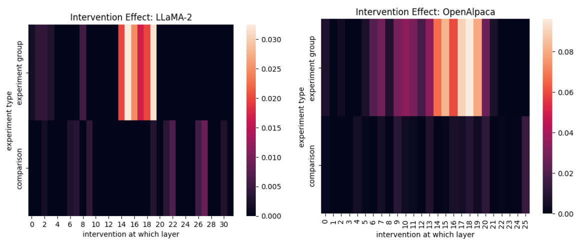
We recall the notations defined before. The tokenizations of o_1 and o_2 are R_i and R_e; the hidden state of the last token at the l-th layer is h_l. Accordingly, the probability distribution over the output vocabulary set V (with Eqn. 2) is $p_l = \text{softmax}(v_l) = \text{softmax}(h_l \cdot W_u) \in \mathbb{R}^{|V|}$. Our aim is to demonstrate how the information about o_1 encoded in hidden states of middle layers plays a causal role in the prediction of o_2. The technique of **Intervention** (Pearl, 2001; Li et al., 2023a) fits the objective, where we strategically intervene on these inner hidden states to eliminate the information related to o_1 (through Logit Lens) and observe the resultant impact on predicting o_2.

Intervention We define the intervention $\mathcal{I}_l : h_l \rightarrow h_l^*$, where h_l^* denotes the intervened hidden state. v_l^* is the corresponding logits (through Logit Lens) of h_l^*: $v_l^* = h_l^* \cdot W_u$. Denoting that (before intervention) $v_{min} = \min_{0 \leq j < |V|} \{v_l[j]\}$, we expect v_l^* meets the following constraints:

$$v_l^*[j] = \begin{cases} v_{min}, & j \in R_i, \\ v_l[j], & j \in [0..|V|)/R_i, \end{cases} \tag{3}$$

Which means, observing from Logit Lens, we **eliminate the bias** on o_1 in h_l^* in the computation graph and minimize the side effects on the rest tokens[4]. We solve the linear system $v_l^* = h_l^* \cdot W_u$

[4]More discussion please refer to Appendix D.1.



(a) LLaMA-2-7B (b) OpenAlpaca-3B

Figure 4: Intervention experiment: Brighter color indicates the intervention effect is more significant. In each subfigure, the upper row refers to the experiment group and the lower row refers to the comparison group. Note that for better visualization, we clip the effect value (≤ 0) to 0 for both of the experiment and comparison groups.

to get h_l^*: $h_l^* = v_l^* W_u^T (W_u W_u^T)^{-1}$ (in case that $W_u W_u^T$ is not full-rank, we use the Moore–Penrose inverse (Dresden, 1920) instead). In our implementation, we calculate the difference value for the purpose of numerical stability:

$$h_l^* = h_l + (v_l^* - v_l) W_u^T (W_u W_u^T)^{-1}. \quad (4)$$

Effect We define the effect $\mathcal{E}_l$ of an intervention $\mathcal{I}_l$ is the difference between probabilities of predicting o_2 (tokenization: R_e) at the output layer L before and after the intervention:

$$\mathcal{E}_l = p_L[R_e] - p_L^{\mathcal{I}_l}[R_e]. \quad (5)$$

Ideally, we expect the intervention $\mathcal{I}_l$ has the effect of decreasing the probability of predicting the explicit reasoning result o_2 (i.e., $\mathcal{E}_l > 0$).

Result The Intervention experiment results (averaged over $\mathcal{D}$) are depicted in Figure 4. For each experiment group, we set a **comparison group** where we intervene on $|R_i|$ tokens that are **randomly sampled** from V. Comparing experiment groups and comparison groups, we observe there exist apparent positive effects ($\mathcal{E}_l > 0$) when intervening middle layers (for both LLaMA-2 and OpenAlpaca, positive effects appear in 15-th to 20-th layers) for experiment groups, suggesting that the information about o_1 may be generated and utilized for generating o_2 in these layers. Meanwhile, there is nearly no notable positive effect for comparison groups across all layers. The results verify our hypothesis that the information around implicit reasoning results in middle layers play a role in predicting explicit reasoning results.

5 Locating Important Modules

In previous analysis, we attribute compositional reasoning errors to improperly generating implicit reasoning results. In this section, we aim to investigate if there sparsely exist some "key" modules (i.e., MHSA or MLP)[5] in LLMs that are responsible for properly generating implicit reasoning results in hidden states of middle layers.

5.1 Locating Methodology

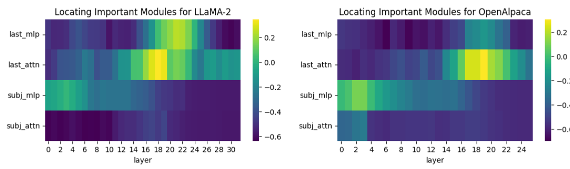
In Section 3, we observe that if inspecting results of the compositional query and its corresponding second-hop query align well, the compositional reasoning process is usually in smooth going. Given this, combining the key idea in Causal Mediation Analysis (Meng et al., 2022; Stolfo et al., 2023), we propose the following locating method. (1) We run the LLM twice: once with the compositional query in $\mathcal{D}_{gap}$ in the length of T_1 and once with its corresponding second-hop query in the length of T_2. For the compositional pass, we denote the module outputs in the computation graph as $\{\eta_l^t | \eta \in \{a, m\}, l \in [1..L], t \in [1..T_1]\}$ (a for MHSA, m for MLP, l indexing layers, t indexing tokens). For the second-hop pass, we denote the outputs as $\{\hat{\eta}_l^t | \eta \in \{a, m\}, l \in [1..L], t \in [1..T_2]\}$. (2) We replace a single module output of interest in the compositional pass computation graph with its counterpart in the second-hop pass computation graph. We focus on two token positions: the **last subject token** (which refers to (s_1, r_1) for compositional queries, e.g., "the sports associated with Giorgio Chinaglia") and the **last token**[6]. We denote the original probability of predicting o_2 as $p(o_2)$ and the probability after replacement as $p(o_2 | \hat{\eta}_l^{t^*} \rightarrow \eta_l^t)$. (3): We define the effect of the replacement $\hat{\eta}_l^{t^*} \rightarrow \eta_l^t$ as $p(o_2 | \hat{\eta}_l^{t^*} \rightarrow \eta_l^t) - p(o_2)$.

5.2 Insight

We depict the **Average Indirect Effect** (AIE) of replacements over modules, tokens, and layers in Figure 5. We observe that replacing the MHSA output at the position of (last-token, $18\backslash19$-th layer) has the largest effect on finally predicting the correct answer o_2. Interestingly, this coincides with the intervention experiment results in Figure 4, implying that MHSA modules of these positions play an important role in properly accumulating and

[5]We introduce the LLM architecture in Appendix C.1

[6]These two positions have been demonstrated as most informative for factual reasoning (Meng et al., 2022).



(a) LLaMA-2-7B (b) OpenAlpaca-3B

Figure 5: AIE for replacements. "last": last token; "subject": last subject token; "mlp": replace the MLP output; "attn": replace the MHSA output. Brighter positions indicate replacements of larger effect (more important).

leveraging implicit reasoning results.

6 Patching Compositional Reasoning

Grounded on the empirical insights in Section 4 and Section 5, we are poised to introduce the CREME approach, designed to correct compositional reasoning failures via editing the parameters of MHSA at the **located positions**. We demonstrate its superiority through comparative analyses with two recent baselines for correcting compositional reasoning (Sakarvadia et al., 2023; Ghandeharioun et al., 2024) and a a widely recognized model editing baseline (Meng et al., 2022).

Specifically, our edit objective is the MHSA output matrix at the l-th layer W_O^l (for detailed description, please refer to Eqn. 7). Following Santurkar et al. (2021), we view W_O^l as a linear associative memory (Kohonen, 1972): $W_O^l \in \mathbb{R}^{d\times d}$ operates as a key-value store for a set of vector keys $K = [k_1|k_2|...]$ and corresponding vector values $V = [v_1|v_2|...]$, by solving $(W_O^l)^T K = V$.

For a given compositional query and its corresponding second-hop query, we run the LLM twice: once with the compositional query and once with the second-hop query. In the first pass with the compositional query, the **input** of W_O^l at the last token position is $k_* \in \mathbb{R}^{d\times 1}$; in the second pass with the corresponding second-hop query, the **output** of W_O^l at the last token position is $v_* \in \mathbb{R}^{d\times 1}$. We aim to edit W_O^l to $\hat{W_O^l}$ such that:

$$\text{minimize } \|(\hat{W_O^l})^T K - V\|_F^2 \text{ and } (\hat{W_O^l})^T k_* = v_*,$$

where the Frobenius norm guarantees consistent predictions on irrelevant queries while the constraint implements the edit as an insertion of (k_*, v_*) into the linear memory $\hat{W_O^l}$. Following Meng et al. (2022), we derive a closed form solution: $\hat{W_O^l} = W_O^l + (C^{-1}k_*)^T \Lambda^T$ where $C = KK^T$ is a constant to estimate the uncentered covariance of k (note that k is randomly sampled from Wikipedia to represent irrelevant queries) and $\Lambda = (v_* - (W_O^l)^T k_*)/(C^{-1}k_*)^T k_*$. Hopefully, the edited LLMs are able to properly generate implicit reasoning results at the located position and thus alleviate failures of compositional reasoning.

What is worthy noting is that applying CREME only requires **a single case** of (compositional query, referenced second-hop query) while the patching effect can **generalize to many other related cases**. We showcase the effect of CREME in Table 2 and quantitatively discuss this generalization effect in Section 6.2.

6.1 Dataset, Baseline and Evaluation Metric

Dataset The dataset $\mathcal{D}_{edit}$ we use for editing and evaluating LLMs is built based on the $\mathcal{D}_{gap}$ filtered in Section 3. For each example in $\mathcal{D}_{edit}$, it has the following fields: (1) **Original** input I_o is a cloze test form of the compositional two-hop query. Accordingly, we also have the correct answer (ground-truth) and the originally predicted wrong answer for I_o: A_o and $\widetilde{A_o}$, respectively[7]. In the experiment, we use I_o and its corresponding second-hop query to edit the LLM. (2) **Paraphrasing** input I_p is a paraphrase of I_o. Note that A_o and $\widetilde{A_o}$ are also applicable to I_p. (3) **Generalization** input I_g is a compositional two-hop query where its first-hop sub-knowledge is shared with I_o while the second-hop sub-knowledge is different from I_o. We denote the correct answer for I_g is A_g. (4) **Irrelevant** input I_i is a compositional two-hop query that is irrelevant to I_o and does not share the final answer with I_o. Detailed information about $\mathcal{D}_{edit}$ is available in Appendix B.

Baseline We choose two related works in the field of correcting compositional reasoning errors through manipulating the inner workings of LLMs: **Memory Injection** (Sakarvadia et al., 2023) and **CoT-PatchScopes** (Ghandeharioun et al., 2024) as our baselines. Memory Injection enhances the compositional reasoning through explicitly injecting the implicit reasoning result (so-called "memory") into the hidden states in the residual stream. CoT-PatchScopes corrects the compositional reasoning through mimicking the noted Chain-of-Thought (CoT) reasoning (Wei et al., 2022) to re-route forward computation. Besides, we also compare

[7]e.g.,for the fourth case in Table 1:A_o=England; $\widetilde{A_o}$=Italy.

Testing type	Input	Prediction w.o. CREME	Prediction w. CREME
Hasty Answer II			
Paraphrasing	What is the citizenship of the creator of C. Auguste Dupin?	France	American
Paraphrasing	What is the nationality of the creator of C. Auguste Dupin?	France	United States of America
Paraphrasing	The country where the creator of C. Auguste Dupin is a citizen is	France	United States of America
Generalization	Which city did the creator of C. Auguste Dupin die in?	Paris	Baltimore, Maryland
Incomplete Reasoning			
Paraphrasing	What is the capital of the country where Sven Väth is a citizen?	Germany	Berlin
Paraphrasing	In what city is the capital located of the country that Sven Väth is a citizen of?	Germany	Berlin
Generalization	The official language of the country that Sven Väth is a citizen of is	Germany	German

Table 2: Case study for generalization effect of correcting the (1) **Hasty Answer II** error: the original input (used for correcting) is "The country that the creator of C. Auguste Dupin belongs to is". The original prediction is "France" (Reference: C. Auguste Dupin is French, while his creator Edgar Allan Poe. is American.); and (2) **Incomplete Reasoning** error: the original input (used for correcting) is "The capital of the country that Sven Väth is a citizen of is". The original prediction is "Germany" (Reference: Berlin.).

CREME with **ROME** (Meng et al., 2022), a state-of-the-art model editing method. Detailed implementations are available in Appendix D.

Evaluation Metric In order to comprehensively validate the effectiveness of CREME, we propose four evaluation metrics: ***Correction***, ***Paraphrasing***, ***Generalization*** and ***Specificity***. Following Sakarvadia et al. (2023), all the metrics are formulated on the basis of Improvement Percentage (**IP**), which is calculated as $\mathrm{IP}(I, A) = \frac{p_{\mathcal{M}^*}(A|I) - p_{\mathcal{M}}(A|I)}{p_{\mathcal{M}}(A|I)}$. This formula quantifies the enhancement in prediction probability of an answer A given an input query I, facilitated by the post-edit LLM $\mathcal{M}^*$ in comparison to the pre-edit LLM $\mathcal{M}$. Specificially, *Correction* quantifies $\mathrm{IP}(I_o, A_o)$ (larger is better); *Paraphrasing* is $\mathrm{IP}(I_p, A_o)$ (larger is better); *Generalization* is $\mathrm{IP}(I_g, A_g)$ (larger is better) and *Specificity* is $\mathrm{IP}(I_i, A_o)$ (smaller is better). CoT-PatchScopes, due to its nature of input-dependent, only fits the *Correction* evaluation. We report the average results over $\mathcal{D}_{edit}$ in Section 6.2. Note that to handling the multiple tokens in the answer, we calculate the probabilities for all of the tokens in the predicted probability distribution and report the mean value.

6.2 Experiment Results

The main experiment results are shown in Table 3. For brevity, we omit $\times 100\%$ for each IP value. We observe that CREME achieves better performance than baselines on all metrics, not only achieving notable improvement on I_o (the query used for editing), but also effectively generalizing to I_p (paraphrased queries). Interestingly, editing with I_o also improves (at most $+366\%$) the compositional reasoning on I_g (only sharing first-hop knowledge with I_o), demonstrating the effectiveness of CREME on generating proper implicit reasoning results in middle layers. Besides, the Specificity score of CREME is low, showing that the CREME does not aimlessly improve the probability of predicting A_o for irrelevant inputs I_i. In comparison, the Correction score of Memory Injection ($+221\%$ for LLaMA-2) is almost the same with the original paper[8] while we find it is less effective to generalize to I_p and I_g. Moreover, its high Specificity score implies its shortcoming of aimlessly improving the probability of predicting A_o. We also show $\mathrm{IP}(I_o, \widetilde{A_o})$ in Figure 7. A good correction method should have little positive improvement on predicting the wrong answer $\widetilde{A_o}$. We observe that $p(\widetilde{A_o}|I_o)$ approximately remains unchanged with CREME, while is apparently enlarged with Memory Injection and PatchScopes.

One natural concern arises regarding the sufficiency of ***Correction*** **and** ***Paraphrasing*** **metrics in practice**. To this end, we evaluate the probability of an event where the probability of predicting A_o exceeds that of predicting $\widetilde{A_o}$: $p(A_o) > p(\widetilde{A_o})$. We compare CREME against baselines using this new metric and two types of input (I_o and I_p) in Table 4. The results underscore CREME's efficacy in significantly improving the event probability, thereby outperforming the unedited LLM and establishing a considerable lead over the two baselines.

Although CREME is not comparable to traditional model editing methods (the latter require A_o for editing, while CREME does not), we compare CREME with a well-regarded model editing method: ROME (Meng et al., 2022) for a comprehensive investigation. The results[9] are shown

[8]Nonetheless, it still falls far behind CREME. Given that both CREME and Memory Injection aim to enhance the information of implicit reasoning results encoded in intermediary hidden states, we attribute the efficacy of CREME to its compatibility with models.

[9]*Correction* and *Paraphrasing* scores are using the event

Evaluation Metrics	C(↑)	P(↑)	G(↑)	S(↓)
LLaMA-2-7B	3.2%	2.3%	13.1%	0.3%
CoT-PatchScopes	+1.20	–	–	–
Memory Injection	+2.21	+0.30	+0.32	+26.72
CREME (**Ours**)	**+17.0**	**+7.99**	**+1.27**	**+0.86**
OpenAlpaca-3B	7.2%	7.0%	13.5%	0.6%
CoT-PatchScopes	+0.91	–	–	–
Memory Injection	+0.98	+0.45	+0.75	+2.93
CREME (**Ours**)	**+43.3**	**+23.71**	**+3.61**	+1.24

Table 3: CREME versus baselines with the proposed four metrics: **C** for "Correction", **P** for "Paraphrasing", **G** for "Generalization" and **S** for "Specificity". Note that the values in the table is averaged improvement percentage (i.e., we calculate the improvement percentage for each single case and then do average over the dataset.) and hence it is meaningless to calculate values like $7.2\% \times (1 + 43.3) = 318.96\% > 100\%$.

Input Types	Correction Input I_o	Paraphrasing Input I_p
LLaMA-2-7B		
Original	59.5%	35.7%
+CoT-PatchScopes	53.0%	–
+Memory Injection	63.0%	40.3%
+CREME(**Ours**)	**87.5%**	**52.9%**
OpenAlpaca-3B		
Original	58.0%	42.7%
+CoT-PatchScopes	57.3%	–
+Memory Injection	58.7%	43.8%
+CREME(**Ours**)	**95.3%**	**70.5%**

Table 4: The event probability of $p(A_o) > p(\widetilde{A_o})$.

in Table 5. Our findings reveal that while ROME marginally surpasses CREME in terms of the *Correction* score of ROME – attributable to ROME's direct application of A_o for editing and its optimization procedure designed to entirely fit $p(A_o)$ – CREME performs obviously better than ROME in paraphrased, generalization and irrelevant cases. This highlights the effectiveness of CREME on correcting compositional reasoning.

To make readers have better sense of the realistic effect of using CREME to improve LLMs' compositional reasoning performance, we also report the (1) decreasement percentage of $\log \text{PPL}$ values[10] (of predicting correct answers) and (2) prediction accuracy in paraphrasing and generalization testing cases before and after applying CREME in Table 6.

In Figure 6, we show the effects of **editing different layers**, where results align well with the results of the locating experiment (Figure 5).

probability of $p(A_o|I) > p(\widetilde{A_o}|I)$.

[10] https://huggingface.co/docs/transformers/en/perplexity

7 Related Work

Compositional Reasoning of LLMs LLMs fail to solve a large proportion of compositional multi-hop questions, even successfully solving all their single-hop sub-questions (Press et al., 2023; Dziri et al., 2023). Early works towards mitigating this issue typically prepend crafted demonstration exemplars containing the "thought process" of solving the compositional query step-by-step and encourage LLMs to imitate the process via in-context learning (Nye et al., 2021; Wei et al., 2022; Zhou et al., 2023; Drozdov et al., 2023; Press et al., 2023). Recent works turn to inspect the inherent compositional reasoning mechanism (Hou et al., 2023) of LLMs. Sakarvadia et al. (2023) manually injects implicit reasoning results into LLMs at the middle layers to correct compositional reasoning failures. (Ghandeharioun et al., 2024) fixes compositional reasoning errors through re-routing inner hidden representations in the computation graph to mimic chain-of-thought reasoning process. Nonetheless, their interventions in the reasoning process are rough so that the improvement is limited and hardly generalize to other related queries. To this end, we elaborately analyze the cause of compositional reasoning failures, locate a small set of parameters in LLMs that are responsible for such failures and precisely edit them to correct such failures. The work and Yang et al. (2024) are concurrent, where both of the two works show empirical evidence that LLMs can latently perform multi-hop reasoning internally.

8 Conclusion

In this paper we study and patch the compositional reasoning of LLMs. Through examining failure instances and conducting diverse analysis experiments, we demonstrate successful compositional reasoning within LLMs hinges on its awareness of generating and leveraging implicit reasoning results. Moreover, we locate few important MHSA modules in LLMs that are responsible for properly generating and leveraging implicit reasoning results via causal mediation analysis. To this end, we propose CREME, to compositional reasoning failures via editing the located MHSA parameters and empirically demonstrate its superiority.

Limitations

Technique Part of our observation and experiments in Section 4 and Section 3 are on the basis of Logit Lens (nostalgebraist, 2020). Though Logit Lens is a widely used tool for analyzing the inner workings of language models (Geva et al., 2022, 2023; Dar et al., 2023; Sakarvadia et al., 2023; Katz and Belinkov, 2023; Ram et al., 2023), we acknowledge that it is only an approximate way to interpret the information in the inner hidden states of the LLMs (Belrose et al., 2023). Nonetheless, the residual stream architecture of Transformers guarantees that Logit Lens makes sense to a large extent. In our experiments, we try to conduct experiments with different techniques for the **cross-validation** of our observations and conclusions (By way of example, the observations in the locating experiments (Section 5) to some extent validate the observations of the intervention experiments in Section 4.2).

LLM Due to the constraints of available computation resource, we are able to conduct most of our experiments with LLMs of seven billion scale (LLaMA-2-7B (Touvron et al., 2023)) and three billion scale (OpenAlpaca-3B (Su et al., 2023b)). Both of these two LLMs are fully open-sourced and popular in academic community and real-world applications (Wu et al., 2023; Wang et al., 2024; Hou et al., 2023; Li et al., 2023b). In the future work, we aim to validate our conclusions on LLMs of larger scale.

Task In this work, we mainly focus on the task of the compositional reasoning on factual knowledge, which is generally pursued by lots of research works (Misra et al., 2023; Press et al., 2023; Zhong et al., 2023; Sakarvadia et al., 2023). We aim to validate our main conclusion about the significance of implicit reasoning results in the compositional reasoning process in other types of compositional reasoning task (Lu et al., 2023; Hou et al., 2023)(e.g., Arithmetic Reasoning for multiple operands) in the future work.

Ethical Considerations

We study the inner workings for the compositional reasoning of LLMs, which helps the black-box LLMs become more transparent and trustworthy (Räuker et al., 2023). The CREME method introduced in this work is originally designed for correcting the compositional reasoning failures of LLMs. CREME only require slightly update a small set of parameters in LLMs and can generalize to a number of related queries (paraphrased queries or compositional queries sharing first-hop knowledge with the query used for conducting CREME). However, just like traditional model editing methods (De Cao et al., 2021; Mitchell et al., 2022; Meng et al., 2022, 2023; Bi et al., 2024), it may also be utilized to insert inaccurate (or out-of-date) information into the pretrained LLMs, potentially resulting in negative influences in real-word applications of LLMs such as information retrieval (Lian et al., 2020a) and recommendation (Lian et al., 2014, 2020b).

Acknowledgements

Special thanks to Dr.Xiting Wang from Renmin University of China for her helpful comments to the early version of the work. We also sincerely thank the anonymous reviewers for their insightful suggestions to this work. The work was supported by grants from the National Key R&D Program of China (No.2021ZD0111801), the Research Grants Council of the Hong Kong SAR under Grant GRF 11217823 and Collaborative Research Fund C1042-23GF, the National Natural Science Foundation of China under Grant 62371411 and InnoHK initiative the Government of the HKSAR,Laboratory for AI-Powered Financial Technologies.

References

Nora Belrose, Zach Furman, Logan Smith, Danny Halawi, Igor Ostrovsky, Lev McKinney, Stella Biderman, and Jacob Steinhardt. 2023. Eliciting latent predictions from transformers with the tuned lens.

Baolong Bi, Shenghua Liu, Lingrui Mei, Yiwei Wang, Pengliang Ji, and Xueqi Cheng. 2024. Decoding by contrasting knowledge: Enhancing llms' confidence on edited facts.

David Chanin, Anthony Hunter, and Oana-Maria Camburu. 2024. Identifying linear relational concepts in large language models.

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

Guy Dar, Mor Geva, Ankit Gupta, and Jonathan Berant. 2023. Analyzing transformers in embedding space. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 16124–16170, Toronto, Canada. Association for Computational Linguistics.

Nicola De Cao, Wilker Aziz, and Ivan Titov. 2021. Editing factual knowledge in language models. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 6491–6506, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Arnold Dresden. 1920. The fourteenth western meeting of the American Mathematical Society. *Bulletin of the American Mathematical Society*, 26(9):385 – 396.

Andrew Drozdov, Nathanael Schärli, Ekin Akyürek, Nathan Scales, Xinying Song, Xinyun Chen, Olivier Bousquet, and Denny Zhou. 2023. Compositional semantic parsing with large language models. In *The Eleventh International Conference on Learning Representations*.

Nouha Dziri, Ximing Lu, Melanie Sclar, Xiang Lorraine Li, Liwei Jiang, Bill Yuchen Lin, Sean Welleck, Peter West, Chandra Bhagavatula, Ronan Le Bras, Jena D. Hwang, Soumya Sanyal, Xiang Ren, Allyson Ettinger, Zaid Harchaoui, and Yejin Choi. 2023. Faith and fate: Limits of transformers on compositionality. In *Thirty-seventh Conference on Neural Information Processing Systems*.

Mor Geva, Jasmijn Bastings, Katja Filippova, and Amir Globerson. 2023. Dissecting recall of factual associations in auto-regressive language models. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pages 12216–12235, Singapore. Association for Computational Linguistics.

Mor Geva, Avi Caciularu, Kevin Wang, and Yoav Goldberg. 2022. Transformer feed-forward layers build predictions by promoting concepts in the vocabulary space. In *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing*, pages 30–45, Abu Dhabi, United Arab Emirates. Association for Computational Linguistics.

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

Asma Ghandeharioun, Avi Caciularu, Adam Pearce, Lucas Dixon, and Mor Geva. 2024. Patchscopes: A unifying framework for inspecting hidden representations of language models.

Peter Hase, Mohit Bansal, Been Kim, and Asma Ghandeharioun. 2023. Does localization inform editing? surprising differences in causality-based localization vs. knowledge editing in language models. In *Thirty-seventh Conference on Neural Information Processing Systems*.

hiyouga. 2023. Fastedit: Editing llms within 10 seconds. `https://github.com/hiyouga/FastEdit`.

Yifan Hou, Jiaoda Li, Yu Fei, Alessandro Stolfo, Wangchunshu Zhou, Guangtao Zeng, Antoine Bosselut, and Mrinmaya Sachan. 2023. Towards a mechanistic interpretation of multi-step reasoning capabilities of language models. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pages 4902–4919, Singapore. Association for Computational Linguistics.

Yiming Ju and Zheng Zhang. 2023. Klob: a benchmark for assessing knowledge locating methods in language models.

Shahar Katz and Yonatan Belinkov. 2023. VISIT: Visualizing and interpreting the semantic information flow of transformers. In *Findings of the Association for Computational Linguistics: EMNLP 2023*, pages 14094–14113, Singapore. Association for Computational Linguistics.

Teuvo Kohonen. 1972. Correlation matrix memories. *IEEE Transactions on Computers*, C-21(4):353–359.

Brenden M. Lake and Marco Baroni. 2023. Human-like systematic generalization through a meta-learning neural network. *Nature*, 623(7985):115–121.

Kenneth Li, Aspen K Hopkins, David Bau, Fernanda Viégas, Hanspeter Pfister, and Martin Wattenberg. 2023a. Emergent world representations: Exploring a sequence model trained on a synthetic task. In *The Eleventh International Conference on Learning Representations*.

Kenneth Li, Oam Patel, Fernanda Viégas, Hanspeter Pfister, and Martin Wattenberg. 2023b. Inference-time intervention: Eliciting truthful answers from a language model. In *Thirty-seventh Conference on Neural Information Processing Systems*.

Defu Lian, Qi Liu, and Enhong Chen. 2020a. Personalized ranking with importance sampling. In *Proceedings of The Web Conference 2020*, pages 1093–1103.

Defu Lian, Haoyu Wang, Zheng Liu, Jianxun Lian, Enhong Chen, and Xing Xie. 2020b. Lightrec: A memory and search-efficient recommender system. In *Proceedings of The Web Conference 2020*, pages 695–705.

Defu Lian, Cong Zhao, Xing Xie, Guangzhong Sun, Enhong Chen, and Yong Rui. 2014. Geomf: joint geographical modeling and matrix factorization for point-of-interest recommendation. In *Proceedings of the 20th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 831–840.

Pan Lu, Baolin Peng, Hao Cheng, Michel Galley, Kai-Wei Chang, Ying Nian Wu, Song-Chun Zhu, and Jianfeng Gao. 2023. Chameleon: Plug-and-play compositional reasoning with large language models. In *The 3rd Workshop on Mathematical Reasoning and AI at NeurIPS'23*.

Kevin Meng, David Bau, Alex J Andonian, and Yonatan Belinkov. 2022. Locating and editing factual associations in GPT. In *Advances in Neural Information Processing Systems*.

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

Kanishka Misra, Cicero Nogueira dos Santos, and Siamak Shakeri. 2023. Triggering multi-hop reasoning for question answering in language models using soft prompts and random walks. In *Findings of the Association for Computational Linguistics: ACL 2023*, pages 972–985, Toronto, Canada. Association for Computational Linguistics.

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

nostalgebraist. 2020. interpreting gpt: the logit lens. `https://www.lesswrong.com/posts/AcKRB8wDpdaN6v6ru/interpreting-gpt-the-logit-lens`.

Maxwell Nye, Anders Johan Andreassen, Guy Gur-Ari, Henryk Michalewski, Jacob Austin, David Bieber, David Dohan, Aitor Lewkowycz, Maarten Bosma, David Luan, Charles Sutton, and Augustus Odena. 2021. Show your work: Scratchpads for intermediate computation with language models.

Judea Pearl. 2001. Direct and indirect effects. In *Proceedings of the Seventeenth Conference on Uncertainty in Artificial Intelligence*, UAI'01, page 411–420, San Francisco, CA, USA. Morgan Kaufmann Publishers Inc.

Ofir Press, Muru Zhang, Sewon Min, Ludwig Schmidt, Noah Smith, and Mike Lewis. 2023. Measuring and narrowing the compositionality gap in language models. In *Findings of the Association for Computational Linguistics: EMNLP 2023*, pages 5687–5711, Singapore. Association for Computational Linguistics.

Ori Ram, Liat Bezalel, Adi Zicher, Yonatan Belinkov, Jonathan Berant, and Amir Globerson. 2023. What are you token about? dense retrieval as distributions over the vocabulary. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 2481–2498, Toronto, Canada. Association for Computational Linguistics.

Tilman Räuker, Anson Ho, Stephen Casper, and Dylan Hadfield-Menell. 2023. Toward transparent ai: A survey on interpreting the inner structures of deep neural networks.

Mansi Sakarvadia, Aswathy Ajith, Arham Khan, Daniel Grzenda, Nathaniel Hudson, André Bauer, Kyle Chard, and Ian Foster. 2023. Memory injections: Correcting multi-hop reasoning failures during inference in transformer-based language models. In *Proceedings of the 6th BlackboxNLP Workshop: Analyzing and Interpreting Neural Networks for NLP*, pages 342–356, Singapore. Association for Computational Linguistics.

Shibani Santurkar, Dimitris Tsipras, Mahalaxmi Elango, David Bau, Antonio Torralba, and Aleksander Madry. 2021. Editing a classifier by rewriting its prediction rules. In *Advances in Neural Information Processing Systems*, volume 34, pages 23359–23373. Curran Associates, Inc.

Noam Shazeer. 2020. Glu variants improve transformer.

Alessandro Stolfo, Yonatan Belinkov, and Mrinmaya Sachan. 2023. A mechanistic interpretation of arithmetic reasoning in language models using causal mediation analysis. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pages 7035–7052, Singapore. Association for Computational Linguistics.

Jianlin Su, Yu Lu, Shengfeng Pan, Ahmed Murtadha, Bo Wen, and Yunfeng Liu. 2023a. Roformer: Enhanced transformer with rotary position embedding.

Yixuan Su, Tian Lan, and Deng Cai. 2023b. Openalpaca: A fully open-source instruction-following model based on openllama. `https://github.com/yxuansu/OpenAlpaca`.

Rohan Taori, Ishaan Gulrajani, Tianyi Zhang, Yann Dubois, Xuechen Li, Carlos Guestrin, Percy Liang, and Tatsunori B. Hashimoto. 2023. Stanford alpaca: An instruction-following llama model. `https://github.com/tatsu-lab/stanford_alpaca`.

Hugo Touvron, Louis Martin, Kevin Stone, Peter Albert, Amjad Almahairi, Yasmine Babaei, Nikolay Bashlykov, Soumya Batra, Prajjwal Bhargava, Shruti Bhosale, Dan Bikel, Lukas Blecher, Cristian Canton Ferrer, Moya Chen, Guillem Cucurull, David Esiobu, Jude Fernandes, Jeremy Fu, Wenyin Fu, Brian Fuller, Cynthia Gao, Vedanuj Goswami, Naman Goyal, Anthony Hartshorn, Saghar Hosseini, Rui Hou, Hakan Inan, Marcin Kardas, Viktor Kerkez, Madian Khabsa, Isabel Kloumann, Artem Korenev, Punit Singh Koura, Marie-Anne Lachaux, Thibaut Lavril, Jenya Lee, Diana Liskovich, Yinghai Lu, Yuning Mao, Xavier Martinet, Todor Mihaylov, Pushkar Mishra, Igor Molybog, Yixin Nie, Andrew Poulton, Jeremy Reizenstein, Rashi Rungta, Kalyan Saladi, Alan Schelten, Ruan Silva, Eric Michael Smith, Ranjan Subramanian, Xiaoqing Ellen Tan, Binh Tang, Ross Taylor, Adina Williams, Jian Xiang Kuan, Puxin Xu, Zheng Yan, Iliyan Zarov, Yuchen Zhang, Angela Fan, Melanie Kambadur, Sharan Narang, Aurelien Rodriguez, Robert Stojnic, Sergey Edunov, and Thomas Scialom. 2023. Llama 2: Open foundation and fine-tuned chat models.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Ł ukasz

Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Advances in Neural Information Processing Systems*, volume 30. Curran Associates, Inc.

Yiqun Wang, Sile Hu, Yonggang Zhang, Xiang Tian, Xuesong Liu, Yaowu Chen, Xu Shen, and Jieping Ye. 2024. How large language models implement chain-of-thought?

Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, brian ichter, Fei Xia, Ed Chi, Quoc V Le, and Denny Zhou. 2022. Chain-of-thought prompting elicits reasoning in large language models. In *Advances in Neural Information Processing Systems*, volume 35, pages 24824–24837. Curran Associates, Inc.

Zhengxuan Wu, Atticus Geiger, Thomas Icard, Christopher Potts, and Noah Goodman. 2023. Interpretability at scale: Identifying causal mechanisms in alpaca. In *Thirty-seventh Conference on Neural Information Processing Systems*.

Sohee Yang, Elena Gribovskaya, Nora Kassner, Mor Geva, and Sebastian Riedel. 2024. Do large language models latently perform multi-hop reasoning?

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. 2024. A comprehensive study of knowledge editing for large language models.

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.

Denny Zhou, Nathanael Schärli, Le Hou, Jason Wei, Nathan Scales, Xuezhi Wang, Dale Schuurmans, Claire Cui, Olivier Bousquet, Quoc V Le, and Ed H. Chi. 2023. Least-to-most prompting enables complex reasoning in large language models. In *The Eleventh International Conference on Learning Representations*.

A Related Works on Mechanistic Interpretability and Model Editing

Mechanistic Interpretability and Model Editing Mechanistic Interpretability, interpreting inner workings of LLMs, is drawing an increasing attention of NLP researchers. Logit Lens (nostalgebraist, 2020) is proposed to interpret hidden states at the middle layers of LLMs via projecting them to the output vocabulary space with the LM head. Subsequent works (Geva et al., 2021, 2022; Dar et al., 2023; Katz and Belinkov, 2023) further explain how LLMs build precise next token predictions. Another line of mechanistic interpretability works focus on inspecting factual knowledge encoded in the LLMs: they first locate such factual knowledge in pretrained LLMs (Dai et al., 2022; Geva et al., 2023; Li et al., 2023b) and then edit them through updating a small set of parameters of LLMs (Meng et al., 2022, 2023; Hase et al., 2023), which is so-called “locate-then-edit” model editing (Ju and Zhang, 2023). In this paper, we shed light on the mechanism of compositional reasoning on factual knowledge and borrow the idea from “locate-then-edit” model editing to locate the root and correct it to patch compositional reasoning failures of LLMs.

B Datasets

Dataset for Non-Editing Experiments Here we mainly introduce the dataset we use for Non-Editing experiments (including inspecting experiments in Section 4.1, intervention experiments in Section 4.2, inference experiments in Section 3 and locating experiments in Section 5.) The dataset $\mathcal{D}$ we use in this paper is sourced from (Zhong et al., 2023), a dataset containing plenty of high-quality compositional multi-hop reasoning cases. For the ease of our study and following the setting of (Press et al., 2023), we collect 1,000 two-hop knowledge items (each with its two single sub-knowledge) as the base of our dataset. For each datum in the dataset, it contains the following component: (1) four paraphrased compositional two-hop knowledge $(s_1, r_1, o_1) \oplus (s_2, r_2, o_2)(o_1 = s_2)$ queries: one of them is in Cloze-Test form and the other three is in Question form; (2) two paraphrased first-hop sub-knowledge (s_1, r_1, o_1) queries: one is in Cloze-Test form and another is in Question form; (3) two paraphrased second-hop sub-knowledge (s_2, r_2, o_2) queries: one is in Cloze-Test form and another is in Question form; and (4) the results for compositional reasoning: the intermediate **implicit reasoning result** o_1 (meanwhile is the answer for the first-hop queries) and the final **explicit reasoning result** o_2 (meanwhile is the answer for the second-hop queries). Following (Meng et al., 2022; Geva et al., 2023; Zhong et al., 2023; Press et al., 2023), We use the Question form queries in the inference experiment (Section 3) and Cloze-Test form queries in most of the rest experiments in this paper. Below is an example for the datum in our dataset.

```
{
    "compositional question query": [
            "Which writer's country of
                citizenship is the same as the
                author of \"Misery\"?",
            "What country does the author of
                \"Misery\" and another writer
                share their citizenship?",
            "What is the nationality of the
                author of \"Misery\"?"
    ],
    "compositional cloze query": "The
        nationality of the author of \"Misery\"
        is",
    "first-hop question query": "Who is the
        author of \"Misery\"?"
    "first-hop cloze query": "The author of
        \"Misery\" is"
    "second-hop question query": "What is the
        nationality of Stephen King?"
    "second-hop cloze query": "The nationality
        of Stephen King is",
    "compositional answer": "United States of
        America", // explicit reasoning result
    "first-hop answer": "Stephen King", //
        implicit reasoning result
    "second-hop answer": "United States of
        America"
}
```

Dataset for Editing Experiments Here we mainly introduce the dataset we use for conducting and evaluating CREME (Correcting Compositional Reasoning via Model Editing) in Section 6. The dataset $\mathcal{D}_{edit}$ we use for editing and evaluating LLMs is built on top of the dataset $\mathcal{D}_{gap}$ filtered in Section 3: for a LLM $\mathcal{M}$: we focus on the example that $\mathcal{M}$ succeeds to predict the correct answer given any of single-hop inputs in it while fails to correctly predict the answer for the corresponding compositional two-hop input in it. In this section, we are going to correct these compositional reasoning failures. Specifically, for each example in $\mathcal{D}_{edit}$, it has the following components: (1) **Original** input I_o, refers to a cloze test form of the compositional two-hop knowledge mentioned above. Accordingly, we also have the correct answer (ground-truth) and the originally predicted wrong answer for I_o: A_o and $\widetilde{A_o}$, respectively[11]. (2) **Paraphrasing** input I_p, refers to a paraphrase (e.g., cloze test → question) of I_o (we collect 3.39 I_p for each I_o in average). Note that I_p shares the A_o and $\widetilde{A_o}$ with I_o. (3) **Generalization** input I_g, refers to a verbalized compositional two-hop knowledge where the first-hop sub-knowledge is shared with I_o and the second-hop sub-knowledge is different from I_o (we collect 2.64 I_g for each I_o in average). We denote the correct answer for I_g is A_g. (4) **Irrelevant** input I_i, refers to a verbalized compositional two-hop knowledge that is irrelevant to I_o and does not share the final answer with I_o (we collect 9.49 I_i for each I_o in average). Below is an example for the dataset (corresponding to the **Incomplete Reasoning** type of errors in Section 3).

```
{
    "Original Input": "The capital of the
        country that Lou Pearlman is a citizen
        of is",
    "Correct Answer for I_o": "Washington, D.C.",
    "Predicted Wrong Answer for I_o": "United
        States of America",
    "Paraphrasing Input":[
        "What is the capital of the country to
            which Lou Pearlman belonged?",
        "Which city serves as the capital of the
            country where Lou Pearlman was a
            citizen?",
        "In which city is the capital of the
            country where Lou Pearlman had
            citizenship?",
        "The capital of the country to which Lou
            Pearlman belonged is",
        ...
    ],
    "Generalization Input": [
        "The official language of the country
            that Lou Pearlman is a citizen of
            is",
        "What is the official language of the
            country that Lou Pearlman is a
            citizen of?",
        ...
    ],
    "Generalization Answer": [
        "American English",
        "American English",
        ...
    ]
    "Irrelevant Input": [
        "Which continent is the country that
            Emma Bunton is a citizen of located
            in?",
        "The official language of the country
            that Thierry Mugler is a citizen of
            is",
        ...
    ],
    "Irrelevant Answer": [
        "Europe",
        "French",
        ...
    ]
}
```

C Language Models

C.1 LLM Architecture

Current Large Language Models (LLMs, in this paper, we conduct most of the experiments with

[11] E.g., for the first case in Table 1: A_o=England; $\widetilde{A_o}$=Italy.

two popular and open-sourced LLMs[12]: LLaMA-2-7B (Touvron et al., 2023) and OpenAlpaca-3B (Su et al., 2023b; Taori et al., 2023).) are mostly built on the basis of traditional Transformer (Vaswani et al., 2017) (Decoder). They are typically consist of an embedding layer E, an output language model (LM) head W_u and a stack of repetitive Transformer blocks between E and W_u.

Embedding Layer Given a tokenized input $inp = [t^1, t^2, ..., t^N]$, where each t_i $(1 \leq i \leq N)$ is a one-hot vector of $|V|$ (V is the vocabulary set) dimensions, the embedding layer is actually an embedding matrix $E \in \mathbb{R}^{|V| \times d}$, projecting the input sparse one-hot vectors into d-dimensional hidden space: $inp \cdot E = [h_0^1, h_0^2, ..., h_0^N]$. h_0^i $(1 \leq i \leq N) \in \mathbb{R}^d$ is the initial hidden state that is forwarded into the first Transformer block (Note that we omit the description for the rotary positional embedding (RoPE) (Su et al., 2023a) added at each Transformer block of the network).

Transformer Block A Transformer block (or a Transformer layer) typically has two sub-modules: a Multi-Head Self-Attention (MHSA) layer and a Multi-Layer Perceptron (MLP) layer. We denote the hidden states at the input and output of the l-th $(1 \leq l \leq L)$ Transformer Block are h_{l-1} and h_l respectively (Since hidden states of all token positions are forwarded parallelly, we define $h_l \triangleq [h_l^1, h_l^2, ..., h_l^N] \in \mathbb{R}^{N \times d}$ to represent the whole hidden states of the l-th layer.). Then we have:

$$h_l = h_{l-1} + a_l + m_l \in \mathbb{R}^{N \times d} \tag{6}$$

where a_l and m_l refer to the MHSA output and the MLP output.

MHSA layer of l-th Transformer block contains four matrices: $W_Q^l, W_K^l, W_V^l, W_O^l \in \mathbb{R}^{d \times d}$. Let H denote the number of attention heads. Then the parameters in each matrix can be equally divided into H parts: each of them is an individual attention head (e.g., for the j-th head, $1 \leq j \leq H$): $W_Q^{l,j}, W_K^{l,j}, W_V^{l,j} \in \mathbb{R}^{d \times \frac{d}{H}}$ and $W_O^{l,j} \in \mathbb{R}^{\frac{d}{H} \times d}$. Then we first compute the attention value for the j-th head: ($M \in \{0, 1\}^{N \times N}$ is the attention mask matrix)

$$A^{l,j} = \text{softmax}\left(\frac{(h_{l-1}W_Q^{l,j})(h_{l-1}W_K^{l,j})^T}{\sqrt{d/H}} \odot M\right)$$
$$\text{head}_l^j = A^{l,j}(h_{l-1}W_V^{l,j}) \in \mathbb{R}^{N \times d/H}$$

The final output of the MHSA a_l is to concatenate these heads together:

$$a_l = \text{Concat}(\text{head}_l^1, \text{head}_l^2, ..., \text{head}_l^H)W_O^l \in \mathbb{R}^{N \times d} \tag{7}$$

MLP layer of l-th Transformer block contains two matrices: $W_{up} \in \mathbb{R}^{d \times d'}$, $W_{down} \in \mathbb{R}^{d' \times d}$ (in LLaMA-2 (Touvron et al., 2023), $d' = \frac{8}{3}d$) and a non-linear activation function SwiGLU (Shazeer, 2020) σ. The output of the MLP m_l can be computed as follows:

$$m_l = \sigma((a_l + h_{l-1})W_{up})W_{down} \in \mathbb{R}^{N \times d}$$

LM Head Let us denote the output of the last Transformer block (at the position of last token) is h_L^N (for LLaMA-2-7B: $L = 32$; for OpenAlpaca-3B: $L = 26$.). The LM head is a matrix $W_u \in \mathbb{R}^{d \times |V|}$ to project the hidden state $h_L^N \in \mathbb{R}^d$ back to the output vocabulary space (probability distribution over the vocabulary set V) to predict the next token:

$$p_L^N = \text{softmax}(h_L^N W_u) \tag{8}$$

C.2 LLaMA-2

LLaMA-2 (Touvron et al., 2023) is a collection of pretrained and fine-tuned generative text models ranging in scale from 7 billion to 70 billion parameters. In this paper, due to the computation resource restraints, we focus on the 7 billion version: LLaMA-2-7b-hf[13], which is a popular open-sourced LLM in both academic researches and industrial applications. LLaMA-2-7B has 32 layers (32 transformer blocks), a vocabulary size of 32,000 and a hidden dimension of 4,096. In the inference experiments of this paper, we adopt the default generation configuration for LLaMA-2-7B provided by Meta:

```
\\LLaMA-2-7B generation configuration
GEN_CONFIGS["llama2-7b"]={
  "bos_token_id": 1,
  "do_sample": True,
```

[12]Due to the page limit, we sometimes present the results with one of them while readers can find the rest results in Appendix E.

[13]https://huggingface.co/meta-llama/Llama-2-7b-hf

```
  "eos_token_id": 2,
  "pad_token_id": 0,
  "temperature": 0.6,
  "max_length": 50,
  "top_p": 0.9,
  "transformers_version": "4.31.0.dev0"
}
```

C.3 OpenAlpaca

OpenAlpaca (Su et al., 2023b) is also an popular instruction-following LLM[14] (fully open-sourced version of Alpaca (Taori et al., 2023)). We adopt the 3 billion version: OpenAlpaca-3B[15], for we want to introduce some variation of parameter scales into our experiments. OpenAlpaca-3B has 26 layers (26 transformer blocks), a vocabulary size of 32,000 and a hidden dimension of 4,096. In the inference experiments of this paper, we adopt the default generation configuration for OpenAlpaca-3B provided by Su et al. (2023b):

```
\\OpenAlpaca generation configuration
GEN_CONFIGS["openalpaca-3b"]={
    "do_sample": True,
    "top_k": 50,
    "top_p": 0.9,
    "generate_len": 128
  "transformers_version": "4.31.0.dev0"
}
```

D Implementation Details

D.1 Intervention

In the Intervention experiments (Section 4.2), a natural worry about the preciseness of the "intervention" manipulation is whether our intervention will direct affect the probability (observing via Logit Lens) of explicit reasoning results or not. Hopefully, the intervention only works on the "implicit reasoning result" (R_i) while due to the restriction of softmax function, the explicit reasoning result might also be affected by the intervention. In practical, this effect (caused by softmax function) on the "explicit reasoning result" (R_e) is rather insignificant ($\sim 3e-5$) and always increasing the probability (given that the summation of all probabilities over the vocabulary is one, our intervention decrease the probability of R_i, naturally improving probabilities for all other tokens.), and hence we do not need to worry about this "side effect". Another potential "side effect" brought by the intervention is caused for the approximation when solving the inverse matrix with PyTorch [16]. Sometimes, the numerical error brought by the approximation can slightly decrease the probability of R_e (observing via Logit Lens at the intervened layer). To mitigate the possibility that the final effect (in Figure 4) is attributed to this "side effect", We additionally apply the following re-checking procedure (Our aim is that (1): we re-check whether the intervention decrease the probability of R_e, and (2): if so, we manually remedy this "side effect".). We first calculate the intervened hidden state h_l^*:

$$\begin{aligned} h_l^* &= h_l + (h_l^* - h_l) \\ &= h_l + (v_l^* - v_l)W_u^T(W_u W_u^T)^{-1} \end{aligned} \tag{9}$$

Concentrating on R_e, we project h_l^* to the raw logits $h_l^* W_u$ and check if there is decreasement on the probability of R_e.

$$\Delta v_l[j] = \begin{cases} v_[j] - (h_l^* W_u)[j], & j \in R_e \\ 0, & j \in [0..M)/R_e \end{cases} \tag{10}$$

Then we re-update the hidden state:

$$h_l^{*,\text{recheck}} = h_l^* + \Delta v_l W_u^T (W_u W_u^T)^{-1} \tag{11}$$

In Figure 4, the values is derived from the following procedures: (1) to make positive effect easier to observe, do clipping on the negative effects ($v < 0 \rightarrow v = 0$) for each single datum; (2) do re-scaling to the effects for each single datum: for effects of different layers, min-max scale them to $[0, 1]$; and (3) average the effects over the whole dataset.

D.2 Inference Experiment

In line with Zhong et al. (2023) and Press et al. (2023), we adopt the question-form queries to check if LLMs have the single-hop knowledges and whether they can compose them together to answer compositional two-hop questions. The main reason behind using question-form queries is that it is convenient for us to use prompting and In-Context examples to make LLMs directly output the answer. As for the prompt for the question queries, following Zhong et al. (2023), we prepend eight different demonstrations (namely exemplars) to guide LLMs. Note that, in our experiments, we eliminate the possibility that LLMs directly "copy" the correct answer from the in-context demonstrations by

[14] https://github.com/yxuansu/OpenAlpaca
[15] https://huggingface.co/openllmplayground/openalpaca_3b_600bt_preview
[16] https://pytorch.org/docs/stable/generated/torch.linalg.inv.html

manually filtering out those demonstrations with the same answer with the questions we want to query. Below is an example for our prompting:

```
Q: In which country was Tohar Butbul granted
     citizenship? A: Israel\n // eight
     demonstrations
Q: Who was Nissan 200SX created by? A: Nissan\n
Q: What continent is the country where Prickly
     Pear grows located in? A: Europe\n
Q: In which country is the company that created
     Nissan 200SX located? A: Japan\n
Q: Which continent is the country where the
     director of My House Husband: Ikaw Na! was
     educated located in? A: Asia\n
Q: What country was the location of the Battle
     of Pressburg? A: Hungary\n
Q: What is the country of citizenship of
     Charles II of Spain? A: Spain\n
Q: Who was Chevrolet Biscayne created by? A:
     Chevrolet\n
Q: What is the name of the head of state of the
     country that Ellie Kemper is a citizen of?
     //our query (e.g., compositional question)
```

D.3 Important Module Locating

We implement our locating method on the basis of Causal Tracing (Meng et al., 2022). Following Meng et al. (2022)'s implementation, we also use a "window" intervention(a few layers before and after the intervened layer). In their original codebase, they set window size to be 10. In our experiments: we find that setting window size to be 2 is enough for us to effectively locate important modules. The locating results shown in Figure 5 adopted the window size of 6. In Figure 5, the values is derived from the following procedures: (1) do re-scaling to the indirect effects for each single datum: map the original value (before replacement) to 0 and map the positive indirect effects to $[0, 1]$ and negative indirect effects to $[-1, 0]$ in a linear manner; and (2) average the indirect effects over the whole dataset.

D.4 Model Editing:CREME

We implement our CREME on the basis of (hiyouga, 2023). The method is described in Section 6.

The edit objective is the MHSA output matrix of l-th layer W_O^l. $W_O^l \in \mathbb{R}^{d\times d}$ operates as a key-value store for a set of vector keys $K = [k_1|k_2|...]$ and corresponding vector values $V = [v_1|v_2|...]$, by solving $(W_O^l)^T K = V$. For a given compositional two-hop query and its corresponding second-hop query, we run the LLM twice: once with the compositional query and once with the second-hop query. We denote that: in the first pass with compositional query, the **input** of W_O^l at the last token position is $k_* \in \mathbb{R}^{d\times 1}$; in the second pass with the corresponding second-hop query, the **output** of W_O^l at the last token position is $v_* \in \mathbb{R}^{d\times 1}$. In practice, when calculating k_* and v_*, we prepend tens of random tokens to the compositional query and the corresponding second-hop query to mimic context environments, and get multiple input vectors and output vectors. Then we average input vectors and output vectors of different context environment to get k_* and v_*, respectively. The edited matrix is: $\hat{W_O^l} = W_O^l + (C^{-1}k_*)^T \Lambda^T$ where $C = KK^T$ is a constant to estimate the uncentered covariance of k (with a sample Wikipedia of text) and $\Lambda = (v_* - (W_O^l)^T k_*)/(C^{-1}k_*)^T k_*$. In our experiments, we also apply a "window" editing (edit a few consecutive layers). In the main experiments, we adopt the window size of 6 in accordance with the locating experiments.

D.5 Memory Injection

We manually inject memories of implicit reasoning results in to the residual stream of middle layers. Note that in the original implementation (Sakarvadia et al., 2023), they set a hyper-parameter, magnitude, to control the strength of injection. In our experiments, we sweep over the possibilities of injecting memories into any single middle layer. For each layer, we search the magnitude from 1 to 10. As for the matrix used for projecting the implicit reasoning results from the vocabulary space back into the hidden space, we try three different approaches: W_u^T (in line with the original paper), W_u^+ (Moore–Penrose inverse) and $W^T(WW^T)^{-1}$. We find that W_u^T is always more effective.

D.6 CoT-PathScopes

We follow the original implementation in Appendix.E. of the PatchScopes paper (Ghandeharioun et al., 2024). Rerouting the hidden states (at the last token position) from source layers to target layers. We use the $\frac{p^*_{\mathcal{M}}(A_o|I_o) - p_{\mathcal{M}}(A_o|I_o)}{p_{\mathcal{M}}(A_o|I_o)}$ to select the best source layer and target layer.

D.7 ROME

We adopt the hiyouga (2023)'s implementation of ROME (Meng et al., 2022). The hyperparameters is in line with their original implementation (hiyouga, 2023; Zhang et al., 2024):

```
layers=[5],
fact_token="subject_last",
v_num_grad_steps=20,
v_lr=1e-1,
```

```
v_weight_decay=1e-3,
clamp_norm_factor=4,
kl_factor=0.0625,
```

Besides, following the convention of model editing works (Meng et al., 2022, 2023), we also use the Cloze-Test form queries to edit LLMs. Note that in compositional queries, the "subject" is usually expressed as the description text containing s_1 and r_1. We treat the description text as the "subject" (e.g., "The sport associated with Giorgio Chinaglia" (association football)).

E Additional Results

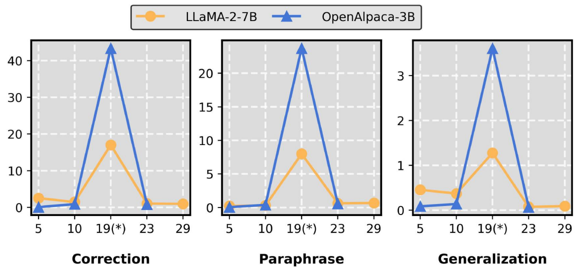


Figure 6: Effects of different editing layers.

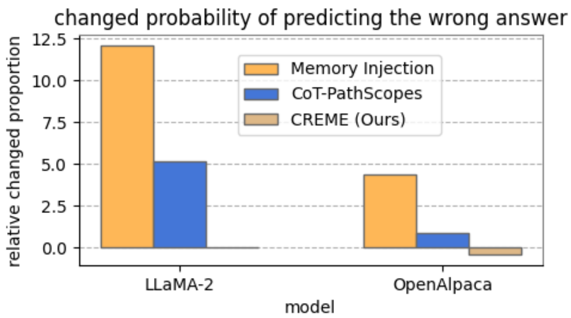
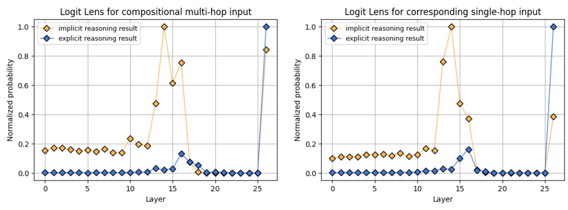


Figure 7: Edit effect on the wrong answer $\widetilde{A_o}$. We anticipate an ideal editing method has little positive effect on predicting $\widetilde{A_o}$.

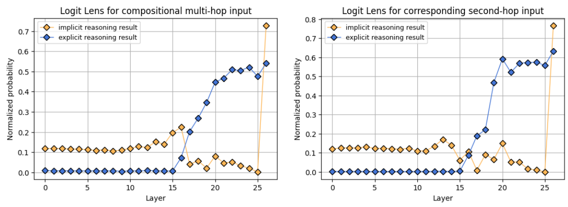
E.1 Logit Lens Inspecting Results

In this section, we mainly present (1): the statistical Logit Lens inspecting results, (2): a case validating our **Hasty Answer II** observation (in Section 3) and (3): a case validating our **Incomplete Reasoning** observation (in Section 3). The statistical inspecting result for OpenAlpaca-3B is depicted in Figure 9. Note that the emerging of "implicit result" seems not as notable as the results of LLaMA-2-7B in Figure 3, the reason is that the layers of emerging peaks for OpenAlpaca-3B are dispersive in the middle layers. We also provide the Logit Lens inspecting results for a single case in Figure 8 for readers' reference. The cases validating **Hasty Answer II** and **Incomplete Reasoning** are depicted in Figure 10 and Figure 11, respectively.



(a) (b)

Figure 8: Logit Lens inspecting results with OpenAlpaca-3B for a single case. (a) refers to the averaged result for inputs of compositional two-hop knowledge and (b) refers to the averaged result for the inputs of second single-hop knowledge. x-axis refers to the layer; y-axis refers to the 0-1 normalized probability. Yellow line and blue line refers to implicit results and explicit results respectively.

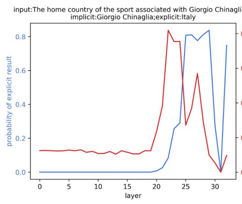


(a) (b)

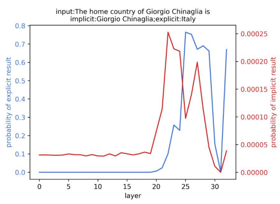
Figure 9: Statistical Logit Lens inspecting results with OpenAlpaca-3B. (a) refers to the averaged result for inputs of compositional two-hop knowledge and (b) refers to the averaged result for the inputs of second single-hop knowledge. x-axis refers to the layer; y-axis refers to the 0-1 normalized probability. Yellow line and blue line refers to implicit results and explicit results respectively. The layers of emerging peaks for OpenAlpaca-3B are dispersive in the middle layers.

E.2 Intervention Results

We present the results for the Intervention experiment (in Section 4.2) in Figure 4. For each experiment group, we set a **comparison group** where we intervene on $|R_i|$ tokens that are **randomly sampled** from V. Comparing experiment groups and comparison groups, we observe there exist apparent positive effects ($\mathcal{E}_l > 0$) when intervening middle layers (for both LLaMA-2 and OpenAlpaca, positive effects appear in 15-th $\sim$ 20-th layers) for experiment groups, suggesting that the information about o_1 may be generated and utilized for generat-

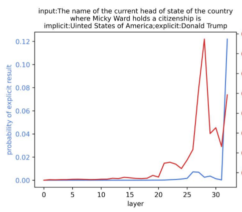


(a)

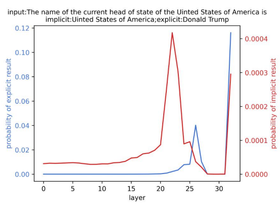


(b)

Figure 10: Logit Lens results for the **Hasty Answer II** error type. We investigate the probability of "Giogrio Chinaglia" (as the implicit reasoning result) and "Italy" (predicted final answer): the compositional input and the corresponding second-hop input fit well now, implying that the model short-cut "Giogrio Chinaglia" and "the home country of" to reason the wrong answer "Italy".

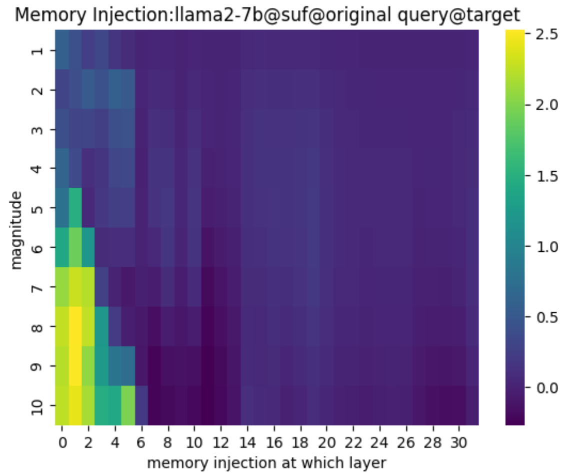


(a)

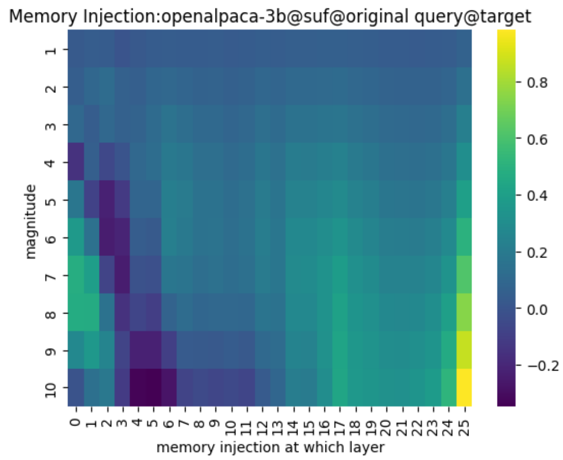


(b)

Figure 11: A success example in comparison with "Incomplete Reasoning" error cases. (a) is the inspecting result for compositional two-hop query and (b) is the inspecting result for the reference (corresponding second-hop query). These two results align well (in (a), the implicit reasoning result is properly generated.) and hence the final explicit reasoning results are successfully predicted.



(a)



(b)

Figure 12: (a) depicts the results for LLaMA-2-7B and (b) depicts the results for OpenAlpaca-3B. In each subfigure, x-axis refers to the layer of injecting implicit reasoning memories; y-axis refers to the magnitude of injecting memories.

ing o_2 in these layers. Meanwhile, there is nearly no notable positive effect for comparison groups across all layers. The results verify our hypothesis that the information around implicit reasoning results in middle layers play a role in predicting explicit reasoning results.

E.3 Memory Injection

The heatmap of averaged results for Memory Injection (Sakarvadia et al., 2023) are depicted in Figure 12. According to this heatmap, for LLaMA-2-7B: we adopt the magnitude of 7 and inject layer of 3, for OpenAlpaca-3B: we adopt the magnitude of 10 and inject layer of 26.

Method	ROME (w. ground-truth)	CREME (w.o. ground-truth)
Correction(↑)	**98.0**%	95.3%
Paraphrasing(↑)	62.5%	**70.5**%
Generalization(↑)	+1.24	+**3.61**
Specificity(↓)	+5.37	+**1.24**

Table 5: Comparing CREME and ROME (Meng et al., 2022) (applied on OpenAlpaca-3B). "w. ground-truth" refers to that ROME requires A_o for editing.

Input Types	Paraphrasing I_p	Generalization I_g
$\Delta \log$ PPL	-29.9%	-17.9%
Pred Accuracy		
Pre-Patching	6.3%	29.9%
Post-Patching	**18.3**%	**38.3**%

Table 6: $\Delta \log$ PPL and Prediction Accuracy results for LLaMA-2-7B.

E.4 PatchScopes

The heatmap for PatchScopes (Ghandeharioun et al., 2024) are depicted in Figure 13. The qualitative are basically in align with the original paper: positive effects distributed in the area where the source layer is larger than the target layer. According to this heatmap, for LLaMA-2-7B: we set the source layer to be 12 and the target layer to be 4, for OpenAlpaca-3B: we set the source layer to be 13 and the target layer to be 7.

E.5 Additional Results of CREME

We additionally show IP$(I_o, \widetilde{A_o})$ in Figure 7. Hopefully, a good correction method has little positive improvement for the prediction of wrong answer $\widetilde{A_o}$. We observe that $p(\widetilde{A_o}|I_o)$ approximately remains unchanged for CREME, while is apparently enlarged with Memory Injection and PatchScopes. In Figure 6, we show the effects of different editing layer, where the effect of editing layer 19 largely surpasses editing other layers (5,10,23,29). This results align well with the results of the locating experiment (Figure 5).

E.6 Showcase of CREME

We use specific cases to show the effect of leveraging CREME to correct the compositional reasoning failures of LLaMA-2-7B in Table 2.

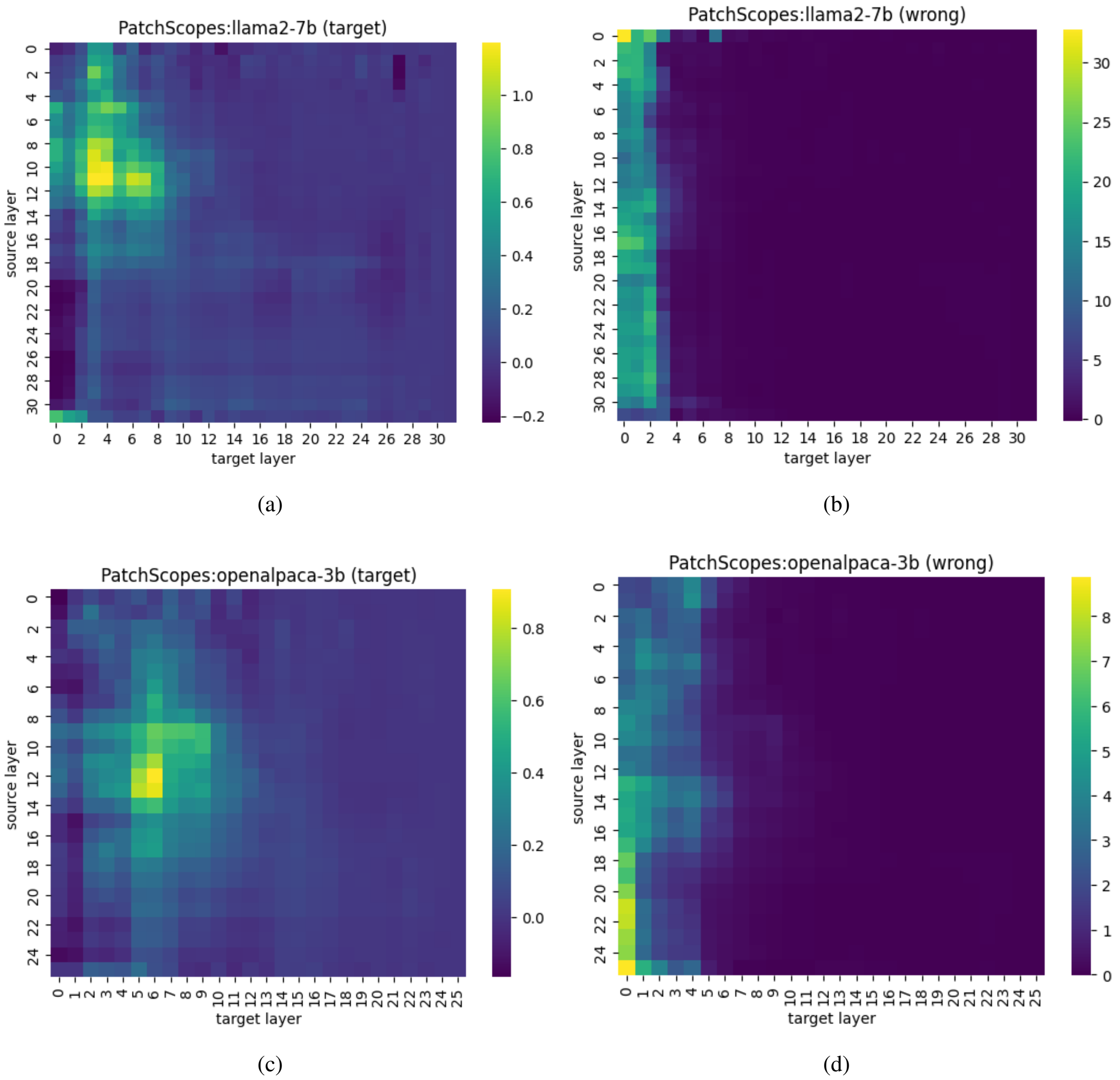


Figure 13: PatchScopes Results: (a) LLaMA-2-7B, $\frac{p^*_{\mathcal{M}}(A_o|I_o)-p_{\mathcal{M}}(A_o|I_o)}{p_{\mathcal{M}}(A_o|I_o)}$; (b) LLaMA-2-7B, $\frac{p^*_{\mathcal{M}}(\widetilde{A_o}|I_o)-p_{\mathcal{M}}(\widetilde{A_o}|I_o)}{p_{\mathcal{M}}(\widetilde{A_o}|I_o)}$; (c) OpenAlpaca-3B, $\frac{p^*_{\mathcal{M}}(A_o|I_o)-p_{\mathcal{M}}(A_o|I_o)}{p_{\mathcal{M}}(A_o|I_o)}$; (d) OpenAlpaca-7B, $\frac{p^*_{\mathcal{M}}(\widetilde{A_o}|I_o)-p_{\mathcal{M}}(\widetilde{A_o}|I_o)}{p_{\mathcal{M}}(\widetilde{A_o}|I_o)}$.