Context-Robust Knowledge Editing for Language Models

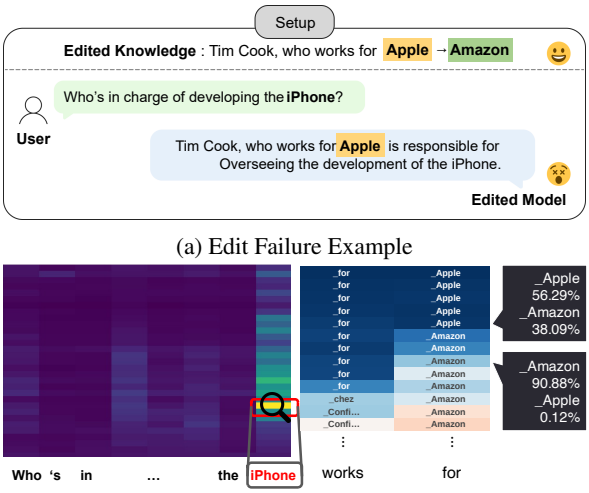
Haewon Park[1]*, **Gyubin Choi**[1]*, **Minjun Kim**[2], **Yohan Jo**[1]†
[1]Graduate School of Data Science, Seoul National University,
[2]School of Electrical Engineering and Computer Science, Gwangju Institute of Science and Technology
{dellaanima2,yeppi315,yohan.jo}@snu.ac.kr
minjun01@gist.ac.kr

Abstract

Knowledge editing (KE) methods offer an efficient way to modify knowledge in large language models. Current KE evaluations typically assess editing success by considering only the edited knowledge without any preceding contexts. In real-world applications, however, preceding contexts often trigger the retrieval of the original knowledge and undermine the intended edit. To address this issue, we have developed CHED—a benchmark designed to evaluate the context robustness of KE methods. Evaluations on CHED show that they often fail when preceding contexts are present. To mitigate this shortcoming, we introduce CoRE, a KE method designed to strengthen context robustness by minimizing context-sensitive variance in hidden states of the model for edited knowledge. This method not only improves the editing success rate in situations where a preceding context is present but also preserves the overall capabilities of the model. We also provide an in-depth analysis of the differing impacts of preceding contexts when introduced as user utterances versus assistant responses, and we dissect attention-score patterns to assess how specific tokens influence editing success. Our dataset and code are available at https://github.com/holi-lab/CoRE.

1 Introduction

Recent large language models (LLMs) exhibit emerging intelligence, largely due to the extensive knowledge acquired from training data. However, some of this knowledge may become outdated or require correction or removal (Ji et al., 2023; Zhao et al., 2024). For instance, the knowledge *"Tim Cook, who works for Apple"* may need to be edited to *"Tim Cook, who works for Amazon"*. Since retraining large models is costly, the field of knowledge editing focuses on modifying only the relevant subset of model parameters or leveraging auxiliary networks or memory (Yao et al., 2023; Zhang et al., 2023). The goal is to ensure the model generates edited knowledge rather than the original one.



(a) Edit Failure Example

(b) Attention Score (c) Logit Lens

Figure 1: An example of knowledge editing failure after prepending the prefix context, where 'iPhone' receives the highest attention. Logit lens reveals that the original knowledge 'Apple' gradually surfaces at later layers.

Previous work typically evaluates the success of knowledge editing by measuring the model's probability of generating the edited knowledge in isolation, without any preceding context. However, this setting is unrealistic, as edited knowledge is often expected to appear within a broader context or in the middle of a conversation with the user. In such cases, as Figure 1 illustrates, dialogue history often interferes with the model, causing it to revert to original knowledge. This issue leads to the need for (1) a challenging benchmark to assess the success of editing when context is present (especially distractive contexts), along with (2) methods that are robust against preceding context.

To address the first need, we introduce **CHED: Contextual Hop Editing Dataset**, a new bench-

*Equal contribution.
†Corresponding author.

Findings of the Association for Computational Linguistics: ACL 2025, pages 10360–10385

mark to evaluate the context robustness of knowledge editing methods (§3). CHED allows this by prepending a prefix context to the edit prompt. For example, the prefix context, such as *"Who's in charge of developing the iPhone?"* in Figure 1 can be added before the edit prompt *"Tim Cook, who works for"*. In collecting these prefix contexts, a key observation is that entities within a prefix context tend to receive disproportionately high attention scores (§5.6) when they have strong semantic relevance to the original knowledge (e.g., "iPhone" in Figure 1). In light of this, we construct prefix contexts using Wikidata by selecting entities connected to the subject and object of the original knowledge and generating sentences that can naturally precede the knowledge statement. As a result, these prefix contexts and the highly relevant words within them distract the model from recalling edited knowledge, establishing CHED as an effective benchmark for assessing the context robustness of knowledge editing methods in real-world use cases of LLMs.

To address the second issue, we propose **CoRE: Context Robust Editing**, a knowledge editing method with enhanced context robustness (§4). It builds on the widely adopted *locate-then-edit* approach, which directly modifies model parameters to edit knowledge. This approach is well known for its practicality, as it remains robust and scalable even when a large number of facts are edited. The core idea of CoRE is to prepend distractive prefix contexts during knowledge editing and to minimize the variance of the model's hidden states generated during the decoding of edited knowledge across these prefix contexts. This simple regularization effectively ensures that only the necessary amount of modification is applied to the parameters, preventing overfitting to varying prefix contexts and enhancing context robustness.

Our extensive evaluations validate CHED and CoRE. Prefix contexts from CHED lead to substantial performance drops compared to the no-context condition across all editing methods. We also found that for the same prefix context, the model is more distracted when the context is provided as a user utterance rather than as its own. Yet, our CoRE method significantly narrows the gap in knowledge editing performance, even consistently maintaining high performance in general abilities and fluency. We provide an explanation through an in-depth analysis of the model's attention patterns.

Our contributions are as follows: (1) We introduce the CHED dataset, a benchmark that assesses the context robustness of knowledge editing methods; (2) We propose CoRE, a knowledge editing method that enhances context robustness by integrating prefix contexts and regularizing the variance of hidden states; (3) We provide an in-depth analysis of the impact of prefix contexts and the CoRE method. Collectively, these contributions underscore the importance of evaluating and enhancing context robustness in knowledge editing.

2 Related Work

Knowledge Editing Knowledge editing (KE) is a field focused on updating a language model's internal representations to incorporate new factual information without requiring full retraining. In this context, factual knowledge is typically represented as a tuple (s, r, o), representing subject–relation–object associations. Given an existing factual association (s, r, o), KE aims to update it to a new factual association (s, r, o^*), where o^* is the new object.

Datasets and Benchmarks CounterFact (Meng et al., 2022b) and zsRE (Levy et al., 2017) have been used widely to evaluate KE methods. To evaluate a broader range of linguistic phenomena and relational complexity, other benchmarks have been introduced, such as MQuAKE (Zhong et al., 2023), CounterFact+ (Hoelscher-Obermaier et al., 2023) and RippleEdits (Cohen et al., 2023). MQuAKE edits multiple pieces of knowledge and evaluates a integrated multi-hop question, thereby broadening the assessment of semantic shifts. CounterFact+ attempts to add a sentence during evaluation by retrieving other samples from the CounterFact that share the same r and o as the current edit triplet and placing those samples before it. Finally, RippleEdits evaluates the ripple effects by testing whether the model correctly updates related facts that become inconsistent after the edit.

Despite these efforts, the impact of prefix contexts on knowledge editing has been underexplored. Our CHED dataset carefully curates prefix contexts to be highly relevant and distractive to edited knowledge while also enabling an examination of how their relevance to s, o, and o^* in knowledge statements contributes to distraction.

Editing Methods Recent work on knowledge editing can be broadly categorized by whether the model's parameters are preserved or modified (Yao et al., 2023). While *weight-preserved* methods typ-

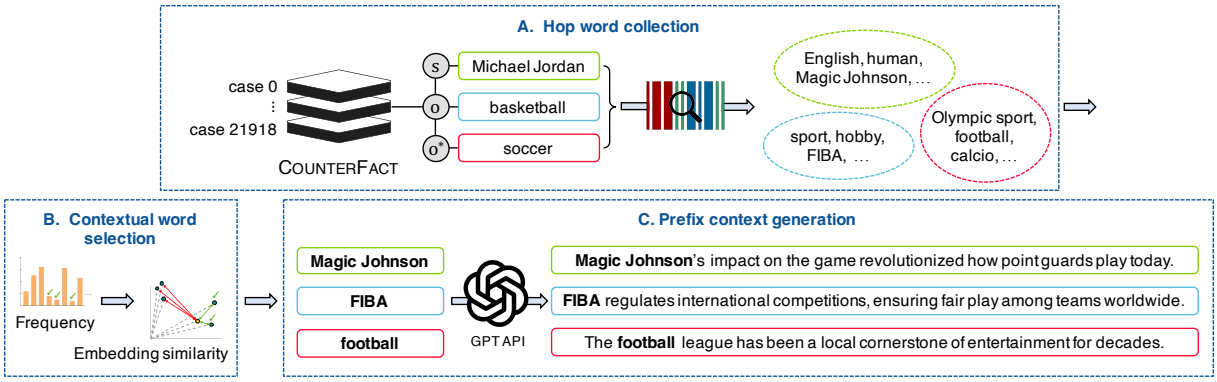


Figure 2: Illustration of CHED Construction Process

ically add auxiliary structures to address each edit requirement (Huang et al., 2023; Hartvigsen et al., 2023; Zheng et al., 2023a; Mitchell et al., 2022b), they can face scalability challenges as the number of edits grows.

In contrast, *weight-modified* methods directly alter model parameters to learn new information, making them often more flexible for substantial updates. These methods can be further categorized into two paradigms: *meta-learning* and *locate-then-edit*. *Meta-learning* approaches train hypernetworks to generate edited parameters but often have difficulty generalizing to extensive edits (Mitchell et al., 2022a; De Cao et al., 2021). *Locate-then-edit* methods pinpoint specific weights to change within the model. A prominent example is MEMIT (Meng et al., 2022b), which edits early-to-mid transformer layers and has inspired successors like PMET (Li et al., 2023), EMMET (Gupta et al., 2024b), and JEEP (Shi et al., 2024).

As a first step toward context robust editing, our CoRE method focuses on the *locate-then-edit* paradigm due to its capacity for supporting mass edits while maintaining model performance. However, we acknowledge the importance of examining and improving context robustness of editing methods in other paradigms and leave this to future work.

3 CHED: Contextual Hop Editing Dataset

As discussed in the previous section, most existing knowledge editing datasets either lack prefix contexts or rely on sentences that do not reflect realistic contexts. This setup differs from real-world LLM usage, where edited knowledge is often expected to appear in response to user prompts or after a relevant context. Consequently, the effectiveness of knowledge editing methods is typically overestimated. To address this gap, we construct **CHED (Contextual Hop Editing Dataset)**, which associates knowledge statements with related prefix contexts. This provides a more realistic and challenging evaluation environment that interferes with LLMs when generating edited knowledge.

3.1 Hop Word Collection (Figure 2-A)

A key idea in CHED is to include words that are semantically relevant to original and edited knowledge within prefix contexts, as they strongly influence the generation of edited knowledge (as illustrated in Figure 1). With this goal, CHED is constructed by expanding upon 21,919 instances from CounterFact, where each instance consists of a fact triplet (s, r, o) and its edited counterpart (s, r, o^*). For each instance, we collect one-hop words by extracting all entities in Wikidata that are connected to s, o, and o^* through any available relations. These **hop words** are expected to naturally appear in the surrounding contexts of each instance and distract the generation of (s, r, o^*). We denote the sets of hop words corresponding to s, o, and o^* as s_{hop}, o_{hop}, and o^*_{hop}, respectively. This resulted in a total of 13,208,725 hop words.

Next, we filtered out entities that were already present in the fact triplets, as well as those consisting solely of special symbols, addresses, or numeric values. After that, we discarded 137 triplet instances in CounterFact for which no hop words were found. As a result, we finalized a dataset of 21,782 triplets with 4,346,604 hop words.

3.2 Contextual Word Selection (Figure 2-B)

The collected hop words consist of only 117,894 unique words, indicating that some words appear repeatedly across many fact triplets (see Ap-

Condition	s_{hop}	o_{hop}	o^*_{hop}
Low Frequency	82.2%	72.7%	88.0%
High Frequency	83.7%	88.0%	90.0%

Table 1: Effect of hop word frequency. Edit success rate when no prefix contexts are prepended is 90.9%.

Method	**Description**
a) Frequency	Select 5 words with lowest frequency in corpus
b) Similarity	Select 5 words with highest *cosine similarity* to main entity
c) Freq-Sim	Get 10 lowest frequency words, select 5 highest *cosine similarity* to main entity
d) Sim-Freq	Get 10 highest *cosine similarity* to main entity, select 5 lowest frequency
e) Log Prob	Select 5 with highest "[main entity] and [hop word]" probability
f) Random	Randomly sample 5 words without any constraints

Table 2: Methods for Hop Word Selection

pendix A.1 for details). The imbalance suggests that less frequent hop words are more uniquely associated with a particular entity in fact triplets. For example, among the hop words of *Michael Jordan*, highly common and general terms appear far more frequently in the entire set of hop words (e.g., *"English"* appears 10,664 times) than words that are more characteristic to Michael Jordan (e.g., *"Magic Johnson"* appears only once). Based on this, we hypothesize that such distinctive hop words may exert a stronger contextual influence when placed before edit sentence (s, r). This is verified in our analysis (Table 1), where sentences constructed with low frequency words dramatically decrease the edit success rate while those constructed with high frequency words do not show a meaningful decrease after being edited by MEMIT. More details are in Appendix A.4.

We explored additional criteria to identify words that are closely and uniquely associated with the entities in given fact triplets. For instance, we considered hop words with high *cosine similarity* to the main entity based on BERT embeddings, capturing semantic closeness. Additionally, we measured the probability that a hop word co-occurs with the entity. Table 2 summarizes the criteria considered for hop word selection.

Figure 3 shows the influence of the six criteria on edit success rates. The Freq-Sim method achieves the lowest score when a prefix context contains o_{hop} (69.1%), indicating that it most effectively

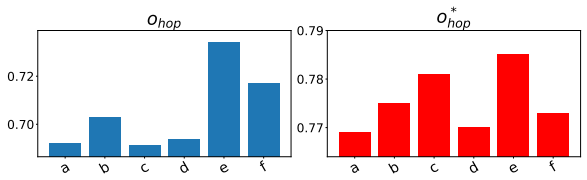


Figure 3: Edit success rate for o_{hop} and o^*_{hop}, with the same experimental setup as Table 1, but evaluated on 5000 samples. Detailed results are available in Appendix A.5.

degrades the model's recall of edited knowledge. Additionally, it attains the second highest score when a prefix context contains o^*_{hop} (78.1%), as one might naturally expect. Consequently, we selected Freq-Sim as our final hop word selection criterion.

The final step is to generate prefix contexts using the hop words from the previous section. We used GPT-4o mini with three key constraints. First, a prefix context should smoothly transition into the edit prompt (s, r) without abrupt topic changes, ensuring coherence. Second, each prefix context should include a designated *hop word* to ensure that the generated sentence prominently reflects the influence of this word. Lastly, each prefix context should contain at most 20 words to maintain clarity and informativeness.

Consequently, we constructed a dataset of 314,385 hop-word prefix context sentences derived from 21,782 fact triplets. Additionally, to evaluate the impact of prefix contexts that directly contain s, o, or $o*$ (rather than their hop words), we generated 326,730 prefix contexts under the same constraints—this time directly incorporating those words. Example prefix contexts are presented in Table 3, and full details on prompt design, validation procedures, and dataset statistics for hop word prefix contexts can be found in Appendix A.6 and Appendix A.8 respectively.

We quantitatively assessed the coherence of the generated prefix contexts using G-Eval (Liu et al., 2023) with GPT-4o-mini. The average coherence score across six types of prefix contexts is 3.4 on a scale from 1 to 5, indicating moderate plausibility. Importantly, if we exclude the o^* and o^*_{hop} types, whose content necessarily diverges from the real world facts, the mean coherence rises to 3.8. The lower-than-ideal coherence scores likely result from our selection of primarily low-frequency hop words (as detailed in §3.2), which inherently constrained coherence potential. Nevertheless, because editing success must remain effective regardless of

Editing Instance: Michael Jordan is a professional basketball → soccer		
Type	**Word**	**Prefix Context Sentence**
s	Michael Jordan	Michael Jordan is often regarded as the greatest player in sports history.
o	basketball	He started playing basketball in high school, impressing everyone with his talent.
$o*$	soccer	Many athletes transition from soccer to other sports when they retire.
s_{hop}	Magic Johnson	Magic Johnson's impact on the game revolutionized how point guards play today.
o_{hop}	FIBA	FIBA regulates international competitions, ensuring fair play among teams worldwide.
o^*_{hop}	football	The football league has been a cornerstone of local entertainment for decades.

Table 3: CHED Dataset Example

the preceding context's coherence, we deliberately prioritized distractiveness and consider this level of coherence acceptable. A detailed explanation of the G-Eval process and results is provided in Appendix A.7.

4 CoRE: Context Robust Editing

In this section, we introduce **Context Robust Editing (CoRE)**, a knowledge editing method for improved robustness to diverse contexts. We build on the *locate-then-edit* approach, such as MEMIT (Meng et al., 2022b), because it enables large numbers of edits. We first provide an overview of MEMIT as preliminaries (§4.1), followed by the details of our CoRE method (§4.2).

4.1 Preliminaries

Transformer MLP as a Key-Value Associative Memory MEMIT interprets MLP layers in Transformers as linear associative memories (Anderson, 1972; Kohonen, 1972), where the weights of the projection layer store **key-value associations**. For example, when a prompt such as *"Tim Cook, who works for"* is provided as input, the hidden state of the subject's last token (i.e., *"Cook"*) encoded by the first MLP layer serves as the key vector $\mathbf{k}$. As $\mathbf{k}$ passes through the second MLP layer W_{proj}, the stored association relevant to the subject is retrieved and embedded into the output **value vector** $\mathbf{v}$ that contains information about the associated object (e.g., *Apple*). At subsequent layers, attention mechanisms refine and propagate this recalled knowledge from the value vector, leading the model to generate the token for o (Meng et al., 2022a; Geva et al., 2023).

Objective Function of MEMIT MEMIT modifies the mapping between key vectors and value vectors, i.e., the projection layer of the MLP, by changing its weights from W to $\widehat{W}$, so that the key $\mathbf{k}$ is remapped to a new value vector $\mathbf{v}^*$ that maximizes the generation probability of o^*. Formally, let (K_E, V_E) be the new keys and values representing the desired edits, and let K_0 be the set of key vectors corresponding to facts that should remain unchanged. MEMIT's objective is:

$$\arg\min_{\widehat{W}} \left\| \widehat{W} K_E - V_E \right\|_F^2 + \lambda \left\| \widehat{W} K_0 - W_0 K_0 \right\|_F^2 \tag{1}$$

The first term enforces knowledge updates, and the second prevents unintended edits, controlled by λ.

Key-Value Vector Extraction A key challenge is constructing the key-value pairs that encode the factual edit $(s, r, o) \rightarrow (s, r, o^*)$. $\mathbf{k}$ and $\mathbf{v}$ are derived from a prompt p that includes s and r and aims to elicit the model's knowledge. In MEMIT, various prefix contexts x_j are prepended to p to improve contextual generalization. Given N prefix contexts, the key vector is derived as $\mathbf{k} = \frac{1}{N} \sum_{j=1}^{N} k(x_j + p)$, where $k(\cdot)$ is obtained by extracting the MLP activation at the last subject token from a chosen layer. We defer the full derivation to Appendix B.1.

Next, the edited value vector $\mathbf{v}^*$ that generates the new knowledge o^* is obtained by minimizing the following loss:

$$\mathbf{v}^* = \underset{\mathbf{v}}{\text{argmin}} \frac{1}{N} \sum_{j=1}^{N} \Big[-\log \mathbb{P}_{G(h^l = \mathbf{v})}[o^* \mid z_j] \Big] + D_{\text{KL}}(\mathbf{v}), \text{ where } z_j = x_j + p \tag{2}$$

where $G(h^l = \mathbf{v})$ denotes the generation output when the hidden layer h^l is set to $\mathbf{v}$. The first term ensures that o^* is generated when provided with the prompt $x_j + p$, while $D_{\text{KL}}(\mathbf{v})$ is a KL-divergence penalty that preserves other related knowledge. The full derivation can be found in Appendix B.2.

4.2 CoRE

In this section, we present our CoRE method for improving the context robustness of key-value extraction by integrating two strategies (Figure 4).

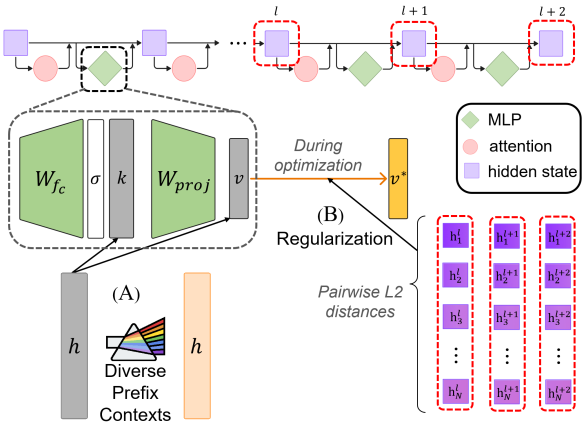


Figure 4: CoRE Method

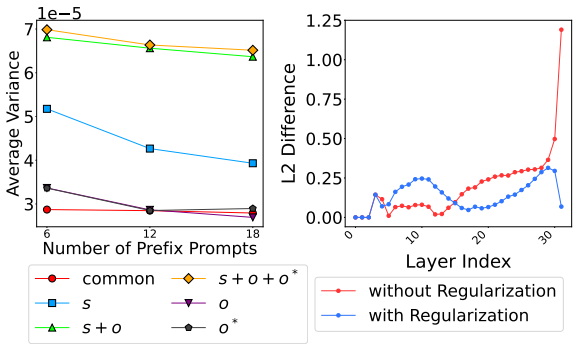


Figure 5: **Left:** Average variance of value vectors across different strategies. **Right:** Average pairwise L2 distances between value vectors, plotted as the difference from the pre-edit. See Appendix C for details.

First, we enhance the prefix contexts used for retrieving key and value vectors (x_j in Equation (2)) by using s, o, and o^*. Second, we regularize inconsistencies among the value vectors obtained when different prefix contexts are used during the update from $\mathbf{v}$ to $\mathbf{v}^*$, as they might account for context-specific signals rather than knowledge edit itself.

Diverse Prefix Contexts (Figure 4-A) The prefix contexts x_j used for extracting key-value vector pairs are crucial because they embed the contextual information into key and value vectors, affecting the generation of edited facts. However, MEMIT simply constructs a prefix context as a sequence starting with one of a small set of predefined words (e.g., *"The"*, *"Therefore"*, *"Because"*, *"I"*, *"You"*). The resulting prefix contexts have little influence on the fact being edited and, as a result it is difficult to optimize $\mathbf{v}^*$ that accounts for various distractive contexts.

To address this issue, CoRE uses combinations of s, o, and o^* as prefix contexts for each edit triplet (e.g., "s + o"). This strategy is effective, as these words are highly relevant to the original and edited facts by nature. As shown in the left plot of Figure 5, prefix contexts that use s, o, and o^* lead to significantly higher variance in value vectors than using the common words, suggesting that these vectors effectively capture a more diverse range of contexts.

Cross-prefix Representation Regularization (Figure 4-B) Although the high variance in value vectors is beneficial for optimizing $\mathbf{v}^*$ to account for various contexts, optimizing $\mathbf{v}^*$ without regularization may lead to undesirable overfitting to individual contexts. To further highlight the significance of this problem, Figure 5 (red line) plots the pairwise L2 distance between value vectors across prefix contexts (from CHED) after knowledge editing via MEMIT, *relative to* the distance before editing. The divergence from 0 clearly demonstrates that differences in value vectors across prefix contexts are amplified after model editing. This can cause overfitting to contexts and reduce generalizability.

To mitigate this issue, we extend the original objective in Equation (2) as follows:

$$\mathbf{v}^* = \underset{\mathbf{v}}{\operatorname{argmin}} \; \mathcal{L}_{\text{orig}}(\mathbf{v}) + \mathcal{L}_{\text{prefix}}, \qquad (3)$$

where $\mathcal{L}_{\text{prefix}}$ is defined as follows. For each layer $\ell \in \mathcal{L}$, we compute N hidden states $\{\mathbf{h}_1^\ell, \ldots, \mathbf{h}_N^\ell\} \subset \mathbb{R}^D$, each corresponding to a distinct prefix context. We enforce regularization by penalizing the squared L2 distances between every pair of hidden states:

$$\mathcal{L}_{\text{prefix}} = \frac{\lambda}{LD} \sum_{\ell \in \mathcal{L}} \sum_{1 \le i < j \le N} \|\mathbf{h}_i^\ell - \mathbf{h}_j^\ell\|^2. \qquad (4)$$

The hyperparameter λ controls the regularization strength. As shown in the right panel of Figure 5, implementing $\mathcal{L}_{\text{prefix}}$ (blue line) substantially reduces hidden state variations across prefix contexts compared to the unregularized model (red line).

5 Experiments

5.1 Metrics

We apply a strict, generation-based criterion: an edit is deemed successful only if the model's output (up to 50 tokens) includes o^* and entirely omits o. We adopt this approach because probability-based evaluations commonly used in prior work

Base Model	Method	Total Avg	Efficacy							Gen	Spe	Avg	General Ability					Avg	Fluency
			No ctx	s	o	$o*$	s_{hop}	o_{hop}	o^*_{hop}				*C-QA*	*T-QA*	*LAM*	*MMLU*	*L-Code*		*N-gram*
Llama3	Base	30.9	1.3	1.1	0.4	40.1	0.9	0.9	13.0	1.4	48.1	11.9	74.5	63.9	31.0	66.9	13.3	49.9	11.1
	MEMIT	60.7	90.9	86.4	46.4	93.6	82.2	72.7	88.0	73.2	34.7	74.2	**73.5**	57.1	28.7	63.4	13.0	47.1	**13.1**
	JEEP	53.3	73.5	67.9	35.9	82.3	65.2	56.0	72.1	51.9	41.0	60.6	65.3	53.6	34.3	63.4	12.8	45.9	21.8
	PMET	56.2	79.1	67.8	36.0	85.2	65.7	57.6	76.3	59.7	**47.5**	63.9	71.7	57.9	**34.8**	**64.8**	**13.4**	**48.5**	16.5
	EMMET	44.4	**94.2**	**93.5**	**78.0**	**95.3**	**92.5**	**90.5**	**93.4**	**80.4**	14.7	**81.4**	0.9	21.0	0.0	15.1	0.0	7.4	29.3
	FT-M	40.0	73.7	69.4	67.0	69.6	67.1	63.7	65.8	58.3	36.0	63.4	32.5	6.6	0.0	44.0	0.0	16.6	128.9
	CoRE-p	62.6	92.7	87.7	49.8	94.8	87.0	80.1	91.7	79.1	35.3	77.6	72.1	57.1	32.6	63.4	13.0	47.6	14.0
	CoRE-p+r	**63.4**	92.4	89.0	55.4	95.1	89.0	83.1	93.2	79.7	34.8	79.1	72.2	**58.1**	30.7	63.8	13.2	47.6	13.3
Mistral	Base	30.7	1.0	0.9	0.3	32.6	1.1	1.1	11.7	1.4	40.3	10.0	71.5	62.5	55.5	60.8	6.7	51.4	6.3
	MEMIT	57.9	86.5	80.1	50.5	84.0	78.0	71.0	81.6	72.3	25.9	70.0	66.3	52.7	48.5	55.6	5.8	45.8	6.1
	JEEP	48.9	73.7	48.7	21.2	64.7	42.1	35.1	56.1	42.0	38.0	46.8	71.2	**61.4**	**55.3**	**60.4**	6.8	**51.0**	**5.9**
	PMET	56.5	81.6	67.6	41.5	76.3	63.9	56.7	74.2	61.0	**42.7**	62.8	71.1	60.7	52.4	58.6	**7.5**	50.1	6.3
	EMMET	42.4	83.1	79.6	61.5	85.2	77.6	74.3	81.3	67.6	14.7	69.4	22.1	19.6	2.5	32.1	0.0	15.3	6.4
	FT-M	42.4	55.3	44.3	36.6	42.1	43.0	36.9	40.9	33.0	39.8	41.3	**71.4**	32.7	52.9	53.4	7.3	43.5	8.5
	CoRE-p	58.7	86.6	82.3	58.8	87.3	80.3	74.2	84.2	71.8	24.2	72.2	65.5	52.7	46.5	54.4	6.4	45.1	6.1
	CoRE-p+r	**60.3**	**88.3**	**83.5**	**63.0**	**88.9**	**83.8**	**79.8**	**87.1**	**77.1**	25.6	**75.2**	65.1	53.3	46.7	54.7	6.6	45.3	6.2

Table 4: Performance on CHED and CounterFact. Efficacy (excluding *No ctx* is measured on CHED, while *No ctx* and Generalization Specificity from CounterFact. **Total Avg** is the average of **Efficacy**, **Gen**, **Spe**, and **General Ability**. **Note:** CoRE-p applies only the *Contextually Diverse Prefix Contexts* method, while CoRE-p+r further adds the *Cross-prefix Representation Regularization Term*.

do not guarantee that the edited knowledge o^* is actually generated, nor do they prevent cases where the edited model initially produces o^* but later reverts to o, as shown in Table 11 in Appendix D. We assess performance across five complementary dimensions—efficacy, generalization, specificity, general ability, and fluency—summarized below.

- **Efficacy**: An edit is considered successful if the model generates o^* without o.
- **Generalization (Gen)**: This metric mirrors Efficacy but tests whether the model correctly produces o^* under paraphrased prompts.
- **Specificity (Spe)**: Ensures that knowledge not intended for editing remains unchanged after the update.
- **General Ability**: Evaluates the core capabilities of the model in five tasks: commonsense reasoning in CommonsenseQA (*C-QA*) (Talmor et al., 2019), factual recall in TriviaQA (*T-QA*) (Joshi et al., 2017), discourse context prediction on LAMBADA (*LAM*) (Paperno et al., 2016), multitask performance in diverse topics in MMLU (Hendrycks et al., 2021) and code generation on LiveCodeBench (*L-Code*) (Jain et al., 2024).
- **Fluency**: Measures the N-gram repetition to detect disfluency introduced by editing, penalizing excessive repetition.

Details of these metrics are in Appendix D.

5.2 Experimental Settings

Datasets and Models Our experiments are conducted using **Llama-3-8B-Instruct** (Grattafiori et al., 2024) and **Mistral-7B-Instruct** (Jiang et al., 2023). For datasets, we experiment on **CHED**, **CounterFact** (Meng et al., 2022b), and **zsRE** (Levy et al., 2017).

Baseline Methods In this paper, we focus on comparing *locate-then-edit* methods, as they reliably handle a large number of edits, including **JEEP** (Shi et al., 2024), **EMMET** (Gupta et al., 2024b), and **PMET** (Li et al., 2023). We also include **FT-M** (Zhang et al., 2024b) as a representative fine-tuning approach. While we experimented with two representative approaches—a *meta-learning method* **MEND** (Mitchell et al., 2022a) and a *weight-preserved method* **IKE** (Zheng et al., 2023a)—both achieved only 0–1% edit success under our stricter generation-based metric, effectively amounting to complete editing failure on 1,000 edits. Consequently, we omit them from Table 4. Details about each method, their results, and hyperparameter settings are provided in Appendix F.7.

5.3 Main Results

CHED and CounterFact Table 4 shows the results of 1,000 edits per method. When using prefix contexts composed of the exact words from the edit triplets (s, o, o^*), Llama3 showed declines of 6.1% for s and 38.2% for o, but an improvement of 3.3% for o^*, while Mistral declined by 13.3%, 40.4% and 5.9%, respectively. Similarly, with prefix contexts consisting of hop words (s_{hop}, o_{hop}, o^*_{hop}), Llama3's performance dropped by 8.42%, 16.48% and 2.47%, and Mistral's by 12.3%, 18.2% and 7.1%. While directly including o causes the largest accuracy drop, the hop-word prefix contexts

Method	Efficacy No ctx	Generalization	Specificity	Average
Base	2.7	3.3	30.3	-
MEMIT	48.7	44.6	28.6	40.6
JEEP	29.9	19.5	23.8	24.4
PMET	43.5	29.2	29.4	34.0
FT-M	49.5	45.1	1.0	31.9
CoRE-p+r	**50.0**	**46.0**	**30.2**	**42.1**

Table 5: Performance on zsRE (Llama3).

Method	Type	s_{hop}	o_{hop}	o^*_{hop}	s_{hop_chat}	o_{hop_chat}	$o^*_{hop\_chat}$
MEMIT	CHED	89.6	86.5	88.7	85.0	73.9	85.4
	Rand Hop	90.8	88.2	90.2	86.2	85.2	85.4
	Rand Cont	94.6	92.4	93.4	89.2	87.4	89.7
CoRE	CHED	95.1	93.8	96.6	91.2	84.9	94.6
	Rand Hop	94.9	94.5	96.7	92.6	90.1	91.9
	Rand Cont	96.7	95.4	96.5	93.4	92.2	93.5

Table 6: Comparison between assistant and user contexts (§5.4 & §5.5). (Rand Hop: Random hop word, Rand Cont: Random context).

Prefix Type	s_{hop}	o_{hop}	o^*_{hop}
hop-word-only	81.8%	73.2%	88.3%
full-sentence	82.2%	72.7%	88.0%

Table 7: Comparison of edit success rates when using hop-word-only versus full-sentence prefix contexts (no-context baseline: 90.9%) using the same editing settings as in Table 1.

also significantly degrade performance. This shows that the presence of even indirectly related contexts can substantially reduce edit success.

For Llama3, CoRE achieves the highest average scores across *Efficacy*, *Generalization*, and *Specificity* while performing competitively to MEMIT in *General Ability* and *Fluency*. While CoRE improves *Efficacy* over MEMIT even when no context is prepended, the improvements are substantially greater when prefix contexts are present, suggesting its effectiveness in enhancing context robustness specifically. EMMET shows context robust *Efficacy*, but it completely breaks down for *Specificity* and *General Ability*. Mistral exhibits a similar pattern, with CoRE substantially outperforming the baseline methods. While some baselines achieve better *General Ability* and *Fluency*, this comes at the cost of significantly reduced knowledge editing performance, which is the primary objective.

zsRE Table 5 presents the results of 1,000 edits on the zsRE dataset. Unlike CounterFact and CHED, which consist of declarative sentences, zsRE is composed of questions. As the results show, CoRE achieves the highest *Efficacy*, *Generalization*, and *Specificity* scores. Overall, these findings further demonstrate its effectiveness in knowledge editing. More detailed results can be found in Table 17 in Appendix.

5.4 User vs. Assistant Contexts

Recent language models are typically trained for dialogues with users using *instruction templates* (Touvron et al., 2023; Grattafiori et al., 2024). Given that this training paradigm separates the roles of *user* and *assistant*, whether a prefix context is provided by the user or generated by the model might influence the model's recall of edited knowledge. For this analysis, we compare two conditions: (1) prepending a prefix context without any instruction template (original setting) and (2) presenting the context as a user utterance using the user template, followed by the assistant template for generating edited knowledge. We use Llama-3-8B-Instruct and measure the success of knowledge editing based on the appearance of o^* and the absence of o within a 10-token window.

Table 6 presents the results for the original setting (subscript *hop*) and the user context setting (subscript *hop_chat*). The edit success rates decrease substantially for both MEMIT (row 1) and CoRE (row 4) when prefix contexts are provided in the user turn. However, CoRE narrows the performance gap compared to MEMIT, demonstrating its context robustness. We speculate that this phenomenon stems from language models being heavily trained to align with user preferences. As a result, they may over-attend to the same information when it is provided by the user and become more susceptible to distraction. These findings suggest an interesting direction for future research on context robustness in chat settings. See Appendix E.1 for more details.

5.5 Effects of Hop Words

We investigate whether the decrease in *Efficacy* observed when testing knowledge editing methods on CHED is merely due to the presence of prefix text or specifically influenced by the curated hop words. We conducted an ablation experiment with two settings: (1) substituting each hop word in CHED with a random word and (2) prepending random prefix contexts.

As shown in Table 6, using random words in place of the curated hop words (rows "Rand Hop") increases *Efficacy* compared to CHED, pronounced for o. Using random contexts (rows "Rand Cont")

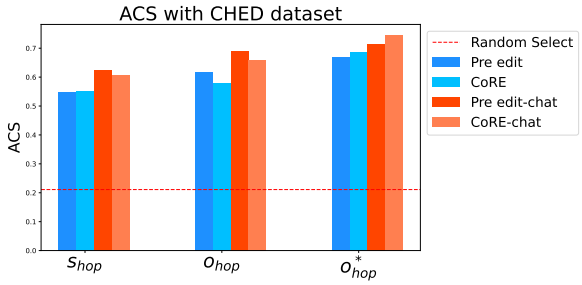


Figure 6: ACS with CHED dataset. The dashed horizontal line represents the ACS when the model selects tokens uniformly at random within a context (0.21).

further improves *Efficacy*, exerting less influence on knowledge recall. According to these results, both hop word selection and prefix context generation are crucial in our CHED construction, with hop word selection appearing to have a more dominant effect. See Appendix E.2 for more details.

As another experiment for examining the impact of hop words, we compared *Efficacy* between prefix contexts composed solely of hop words (e.g., *"Magic Johnson"*) and those using fully formed sentences generated from the same hop words (e.g., *"Magic Johnson's impact on the game ..."*). The results indicated that the difference in *Efficacy* between these two settings is minimal, averaging around 0.4%, suggesting that hop words alone already substantially contribute to distractiveness. Detailed numerical results can be found in Table 7.

5.6 Average Contribution Score

We analyze the influence of hop words more quantitatively based on attention scores. Specifically, we define a metric, Average Contribution Score (ACS), as the proportion of prefix contexts in which a hop word receives the highest attention weight among all words in the context, during the last decoding step of knowledge generation.

More specifically, we measure how strongly the final token t_{last} attends to each token t_i in the prefix context by aggregating t_i's attention weights across all layers and heads in a pretrained Transformer model. Formally, let $A_{\ell,h}(t_i, t_{\text{last}})$ denote the attention weight of token t_i received from t_{last} at layer ℓ and head h. Let L be the number of layers and H be the number of heads per layer. We define the token-level average attention score $\bar{A}_{i\to\text{last}}$ as:

$$\bar{A}_{i\to\text{last}} = \frac{1}{L \cdot H} \sum_{\ell=1}^{L} \sum_{h=1}^{H} A_{\ell,h}(t_i, t_{\text{last}}). \quad (5)$$

Given a knowledge editing case with a prefix context containing a hop word t_{hop}, the indicator I of whether the hop word receives the highest attention is defined as:

$$I = \begin{cases} 1 & \text{if } \operatorname{argmax}_{i\in\text{prefix}} \bar{A}_{i\to\text{last}} = t_{\text{hop}}, \\ 0 & \text{otherwise.} \end{cases} \quad (6)$$

Finally, we define ACS as the percentage of test cases where hop words receive the highest attention:

$$\text{ACS} = \frac{1}{N} \sum_{n=1}^{N} I_n, \quad (7)$$

where N is the number of test cases. This value quantifies reflects the degree of influence of hop words on knowledge recall.

In Figure 6, the blue bars compare the ACS of hop words before (darker) and after (lighter) knowledge editing by CoRE. Compared to random chance (red line), hop words receive significantly more attention. However, for o_{hop}, which is the most distractive type of hop words, the model pays less attention to them after being edited by CoRE (lighter blue), explaining CoRE's context robustness. Conversely, the model pays even greater attention to o^*_{hop} after editing. Since o^*_{hop} is related to the edited knowledge, it provides a useful signal for edited knowledge. The results show that CoRE does not simply reduce the model's attention to prefix contexts; rather, it improves the model's ability to attend less to distractive information and more to useful information in the context. The red bars in the figure represent the user context setting and show the same pattern. See Appendix E.4 for more details.

6 Conclusion

We introduce and release CHED, a benchmark designed to evaluate the context robustness of knowledge editing. Our evaluation across various methods reveals that even those which perform well often fail when a prefix context is introduced. This finding underscores that the aspect measured by CHED has been largely overlooked by previous knowledge editing methods. It emphasizes the importance of this evaluation. To address this gap, we propose CoRE, which enhances context robustness. We hope that CHED, together with CoRE, will contribute to the development of more context robust, practical, and reliable knowledge editing techniques for real-world applications.

Limitations

We built CHED using only 1-hop words extracted from Wikidata relations. Although any entity directly connected by a Wikidata relation is defined as a 1-hop word, this does not guarantee that the semantic relationship is strictly one hop. For example, "U.S. First Lady" might be linked through "U.S. President" to "his spouse", but we did not differentiate such multi-hop nuances. We also experimented with including 2-hop words; however, many of these words appeared only tangentially related to the corresponding entity. Consequently, it remains crucial to explore the degree and relevance of the relationship between these hop words and the edited knowledge—a promising direction for future work. For our CoRE method, we built on the *locate-then-edit* paradigm, which excels in large-scale editing while preserving overall model performance. We believe that further investigation into enhancing context robustness within other paradigms, such as *meta-learning* or *weight-preserving* approaches, would be a beneficial research avenue.

Ethics Statement

Our research focuses on enhancing LLMs by rectifying errors and updating outdated knowledge through knowledge editing techniques. While these methods aim to improve user utility, they also present risks if misused, potentially generating misleading, toxic, or harmful content. It is therefore crucial to enforce strict ethical guidelines and robust safeguards to ensure that any modifications maintain overall performance and prevent the production of unsafe outputs until proper regulatory measures are established.

Acknowledgments

This work was supported by the National Research Foundation of Korea (NRF) grant (RS-2024-00333484) and by the Institute of Information & Communications Technology Planning & Evaluation (IITP) grant (RS-2024-00338140, Development of Learning and Utilization Technology to Reflect Sustainability of Generative Language Models and Up-to-dateness over Time), both funded by the Korean government (MSIT).

References

James A. Anderson. 1972. A simple neural network generating an interactive memory. *Mathematical Biosciences*, 14(3):197–220.

Roi Cohen, Eden Biran, Ori Yoran, Amir Globerson, and Mor Geva. 2023. Evaluating the ripple effects of knowledge editing in language models. *Preprint*, arXiv:2307.12976.

OpenCompass Contributors. 2023. Opencompass: A universal evaluation platform for foundation models. https://github.com/open-compass/opencompass.

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.

Mor Geva, Jasmijn Bastings, Katja Filippova, and Amir Globerson. 2023. Dissecting recall of factual associations in auto-regressive language models. In *The 2023 Conference on Empirical Methods in Natural Language Processing*.

Aaron Grattafiori, Abhimanyu Dubey, Abhinav Jauhri, Abhinav Pandey, and Abhishek Kadian et al. 2024. The llama 3 herd of models. *Preprint*, arXiv:2407.21783.

Richard A Groeneveld and Glen Meeden. 1984. Measuring skewness and kurtosis. *Journal of the Royal Statistical Society Series D: The Statistician*, 33(4):391–399.

Akshat Gupta, Anurag Rao, and Gopala Anumanchipalli. 2024a. Model editing at scale leads to gradual and catastrophic forgetting. *arXiv preprint arXiv:2401.07453*.

Akshat Gupta, Dev Sajnani, and Gopala Anumanchipalli. 2024b. A unified framework for model editing. *arXiv preprint arXiv:2403.14236*.

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

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. *Preprint*, arXiv:2301.04213.

Dan Hendrycks, Collin Burns, Steven Basart, Andy Zou, Mantas Mazeika, Dawn Song, and Jacob Steinhardt. 2021. Measuring massive multitask language understanding. *Preprint*, arXiv:2009.03300.

Evan Hernandez, Arnab Sen Sharma, Tal Haklay, Kevin Meng, Martin Wattenberg, Jacob Andreas, Yonatan Belinkov, and David Bau. 2024. Linearity of relation decoding in transformer language models. *Preprint*, arXiv:2308.09124.

Jason Hoelscher-Obermaier, Julia Persson, Esben Kran, Ioannis Konstas, and Fazl Barez. 2023. Detecting edit failures in large language models: An improved specificity benchmark. In *Findings of the Association for Computational Linguistics: ACL 2023*, pages 11548–11559, Toronto, Canada. Association for Computational Linguistics.

Zeyu Huang, Yikang Shen, Xiaofeng Zhang, Jie Zhou, Wenge Rong, and Zhang Xiong. 2023. Transformer-patcher: One mistake worth one neuron. *Preprint*, arXiv:2301.09785.

Naman Jain, King Han, Alex Gu, Wen-Ding Li, Fanjia Yan, Tianjun Zhang, Sida Wang, Armando Solar-Lezama, Koushik Sen, and Ion Stoica. 2024. Livecodebench: Holistic and contamination free evaluation of large language models for code. *Preprint*, arXiv:2403.07974.

Ziwei Ji, Nayeon Lee, Rita Frieske, Tiezheng Yu, Dan Su, Yan Xu, Etsuko Ishii, Ye Jin Bang, Andrea Madotto, and Pascale Fung. 2023. Survey of hallucination in natural language generation. *ACM Computing Surveys*, 55(12):1–38.

Albert Q. Jiang, Alexandre Sablayrolles, Arthur Mensch, Chris Bamford, Devendra Singh Chaplot, Diego de las Casas, Florian Bressand, Gianna Lengyel, Guillaume Lample, Lucile Saulnier, Lélio Renard Lavaud, Marie-Anne Lachaux, Pierre Stock, Teven Le Scao, Thibaut Lavril, Thomas Wang, Timothée Lacroix, and William El Sayed. 2023. Mistral 7b. *Preprint*, arXiv:2310.06825.

Mandar Joshi, Eunsol Choi, Daniel Weld, and Luke Zettlemoyer. 2017. TriviaQA: A large scale distantly supervised challenge dataset for reading comprehension. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1601–1611, Vancouver, Canada. Association for Computational Linguistics.

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

Woosuk Kwon, Zhuohan Li, Siyuan Zhuang, Ying Sheng, Lianmin Zheng, Cody Hao Yu, Joseph E. Gonzalez, Hao Zhang, and Ion Stoica. 2023. Efficient memory management for large language model serving with pagedattention. In *Proceedings of the ACM SIGOPS 29th Symposium on Operating Systems Principles*.

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

Xiaopeng Li, Shasha Li, Shezheng Song, Jing Yang, Jun Ma, and Jie Yu. 2023. Pmet: Precise model editing in a transformer. *arXiv preprint arXiv:2308.08742*.

Yang Liu, Dan Iter, Yichong Xu, Shuohang Wang, Ruochen Xu, and Chenguang Zhu. 2023. G-eval: Nlg evaluation using gpt-4 with better human alignment. *Preprint*, arXiv:2303.16634.

Kevin Meng, David Bau, Alex Andonian, and Yonatan Belinkov. 2022a. Locating and editing factual associations in GPT. *Advances in Neural Information Processing Systems*, 35.

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

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

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

Denis Paperno, Germán Kruszewski, Angeliki Lazaridou, Quan Ngoc Pham, Raffaella Bernardi, Sandro Pezzelle, Marco Baroni, Gemma Boleda, and Raquel Fernández. 2016. The lambada dataset: Word prediction requiring a broad discourse context. *Preprint*, arXiv:1606.06031.

Wenhang Shi, Yiren Chen, Shuqing Bian, Xinyi Zhang, Zhe Zhao, Pengfei Hu, Wei Lu, and Xiaoyong Du. 2024. Joint knowledge editing for information enrichment and probability promotion. *arXiv preprint arXiv:2412.17872*.

Alon Talmor, Jonathan Herzig, Nicholas Lourie, and Jonathan Berant. 2019. CommonsenseQA: A question answering challenge targeting commonsense knowledge. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4149–4158, Minneapolis, Minnesota. Association for Computational Linguistics.

Hugo Touvron, Louis Martin, Kevin Stone, Peter Albert, and Amjad Almahairi et al. 2023. Llama 2: Open foundation and fine-tuned chat models. *Preprint*, arXiv:2307.09288.

Ben Wang and Aran Komatsuzaki. 2021. Gpt-j-6b: A 6 billion parameter autoregressive language model.

Luyu Wang, Yujia Li, Ozlem Aslan, and Oriol Vinyals. 2021. WikiGraphs: A Wikipedia text - knowledge graph paired dataset. In *Proceedings of the Fifteenth Workshop on Graph-Based Methods for Natural Language Processing (TextGraphs-15)*, pages 67–82, Mexico City, Mexico. Association for Computational Linguistics.

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. 2024. Easyedit: An easy-to-use knowledge editing framework for large language models. *Preprint*, arXiv:2308.07269.

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

Junsang Yoon, Akshat Gupta, and Gopala Anumanchipalli. 2024. Is bigger edit batch size always better? – an empirical study on model editing with llama-3. *Preprint*, arXiv:2405.00664.

Mengqi Zhang, Bowen Fang, Qiang Liu, Pengjie Ren, Shu Wu, Zhumin Chen, and Liang Wang. 2024a. Enhancing multi-hop reasoning through knowledge erasure in large language model editing. *Preprint*, arXiv:2408.12456.

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. 2024b. A comprehensive study of knowledge editing for large language models. *Preprint*, arXiv:2401.01286.

Zihan Zhang, Meng Fang, Ling Chen, Mohammad-Reza Namazi-Rad, and Jun Wang. 2023. How do large language models capture the ever-changing world knowledge? a review of recent advances. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pages 8289–8311, Singapore. Association for Computational Linguistics.

Wayne Xin Zhao, Kun Zhou, Junyi Li, Tianyi Tang, Xiaolei Wang, Yupeng Hou, Yingqian Min, Beichen Zhang, Junjie Zhang, Zican Dong, Yifan Du, Chen Yang, Yushuo Chen, Zhipeng Chen, Jinhao Jiang, Ruiyang Ren, Yifan Li, Xinyu Tang, Zikang Liu, Peiyu Liu, Jian-Yun Nie, and Ji-Rong Wen. 2024. A survey of large language models. *Preprint*, arXiv:2303.18223.

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

Ce Zheng, Lei Li, Qingxiu Dong, Yuxuan Fan, Zhiyong Wu, Jingjing Xu, and Baobao Chang. 2023b. Can we edit factual knowledge by in-context learning? In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pages 4862–4876, Singapore. Association for Computational Linguistics.

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.

A Details on CHED construction

A.1 Data Statistics

Table 8 presents various statistics on the frequency distribution of the collected hop words, indicating that the distribution is highly skewed. The skewness of the frequency in our hop words set reached 39. This indicates a highly right-skewed distribution, as skewness values greater than 1 generally suggest such behavior (Groeneveld and Meeden, 1984).

A.2 Skewness Computation

We calculate skewness using moments to describe the shape of hop words frequency distribution. The k-th central moment of a dataset is a measure of the dataset's deviation from the mean, raised to the power k. For skewness, we specifically use the third central moment and the second central moment (variance).

The data points in this context represent each word's frequency in the dataset. The number of unique words in the dataset is denoted as N.

The skewness of a sample is calculated as:

$$g_1 = \frac{m_3}{m_2^{3/2}}$$

where:

- m_3 is the third central moment, which is calculated as:

$$m_3 = \frac{1}{N}\sum_{n=1}^{N}(x_n - \bar{x})^3$$

- m_2 is the second central moment, which is the variance, and is calculated as:

$$m_2 = \frac{1}{N}\sum_{n=1}^{N}(x_n - \bar{x})^2$$

In these formulas, x_n represents the frequency of the n-th word in the dataset, $\bar{x}$ is the mean frequency of the words, and N is the number of unique words in the dataset. The value k refers to

Basic Word Statistics				
Total Words	Unique Words	Max Freq.	Min Freq.	Mean Freq.
4,346,604	117,894	32,086	1	36.87
Frequency Distribution				
Q1 (25%)	Median (Q2)	Q3 (75%)	Std Dev.	Skewness
1.0	1.0	4.0	289.34	39.29

Table 8: word set statistics.

the order of the moment, where $k = 2$ corresponds to variance and $k = 3$ corresponds to skewness.

A.3 Word Frequency

In the collected hop words, we observed that the most frequent terms are primarily derived from formal changes in Wikidata. For instance, the top five most frequent hop words and their respective frequencies are as follows:

- "Brockhaus and Efron Encyclopedic Dictionary" with 32,263 times
- "Small Brockhaus and Efron Encyclopedic Dictionary" with 30,371 times
- "United States of America" with 22,407 times
- "Jewish Encyclopedia of Brockhaus and Efron" with 16,649 times
- "Granat Encyclopedic Dictionary" with 11,953 times

After excluding terms related to changes in Wikidata, the five most frequent terms are:

- "United States of America" with 22,407 times
- "United Kingdom" with 11,150 times
- "English" with 10,664 times
- "human" with 10,096 times
- "Italy" with 9,979 times

These terms are often related to the place of birth or the native language of an entity, and therefore, they do not provide substantial contextual information about the entity.

A.4 Frequency Test

We evaluated how the placement of high-frequency and low-frequency hop words as contextual sentences before edit sentence affects knowledge editing performance. For this experiment, we used 1,000 instances from the CounterFact dataset and applied MEMIT on Llama-3-8B-Instruct. We selected the top five most frequent and the top five least frequent hop words and constructed sentences with these words using GPT-4o mini. The evaluation measured how much the model's ability to recall new knowledge declined when these sentences are placed before the edit prompt (s, r).

Table 1 shows that when low-frequency hop words are used as prefix context, performance drops notably—especially when a sentence containing o_{hop} is placed before the edit prompt, causing edit success rate to fall to 72.7% compared to 88.0% when high-frequency words are used. This result supports our hypothesis that less frequent, uniquely associated hop words exert a stronger contextual influence on the model's ability to recall edited knowledge. Based on these results, we selected hop words using frequency as the primary criterion, prioritizing those with lower occurrence counts.

Condition	s_{hop}	o_{hop}	o^*_{hop}
Freq→Sim	74.5	69.1	78.1
Freq	74.1	69.2	76.9
Sim→Freq	74.2	69.4	77.0
Sim	74.6	70.3	77.5
Random	75.0	71.7	77.3
Logits	76.2	73.4	78.5

Table 9: Contextual Word Selection Methods

A.5 Contextual Word Selection Methods

Table 9 shows the edit success rates for the s_{hop}, o_{hop}, and o^*_{hop} sentences with our 6 word selection methods. We focus on how effectively the prefix context with contextual hop words via each selection method reduces the edit success rate after editing with o_{hop}. This analysis is particularly important because the primary goal in constructing this dataset is to distract the model's editing outcome, thereby recalling the original object. Although the Freq-Sim (69.1%) and Freq (69.2%) methods yield similar results on the o_{hop} sentences, the o^*_{hop} results demonstrate that the Freq-Sim method not only distracts with the o hop sentence but also with the o^* hop sentence, preventing a significant drop in the edit success rate (78.1% for Freq-Sim versus 76.9% for Freq). Consequently, we adopt the Freq-Sim method.

You are tasked to create a set of sentences based on the provided **word list** that establish a natural context.
General Instructions:
1. Flow and Coherence: - Each sentence must smoothly lead into and set up the sentence: '{edit_prompt}'. - The generated sentences should feel like a natural precursor to the given sentence.
2. Word Usage: - Use each word from the **word list** exactly once, **in the exact order provided in the list**. - **Do not change the order** of the words in the **word list**. - Do not repeat any word across sentences - Exclude the following words entirely: '{exclude_words}'.
3. Sentence Structure: - Each sentence must be concise (no longer than 20 words). - Avoid overly general statements or clichés (e.g., "is known for its unique culture and history" or "has historical significance").
4. Output: - Generate exactly {len(one_hop)} sentences. - Each sentence must correspond to one word from the **word list**, in the same order as they appear in the list. - Return only the generated sentences, excluding the target sentence: '{edit_prompt}'.
Word List: '{one_hop}'

Figure 7: Prompt template for generating prefix contexts using one-hop words.

A.6 Prefix Context Generation Methods

To construct a concise yet contextually rich sentence preceding each edit prompt, we used GPT-4o mini with the following three key constraints, providing it with a set of instructions to generate these prefix contexts systematically.

1. **Flow and Coherence**
   Each sentence must lead smoothly and set up the sentence: 'edit prompt'.

2. **Word Usage**
   The sentence must include the hop word while excluding s, o, and o^*.

3. **Sentence Structure**
   Each sentence should be no longer than 20 words and should avoid overly general statements or clichés.

We performed a validation process on the initially generated hop sentence dataset to ensure compliance with the Word Usage constraints. Specifically, we verified whether each hop sentence included its corresponding one-hop word while ensuring that it did not contain s, o, or o^*. However, if the one-hop word itself contained s, o or o^*, its presence in the generated sentence was unavoidable and thus considered valid. For example, if o^*_{hop} was "WikiProject Football" and o was "football", the occurence of "football" in the generated sentence was permitted.

Figure 7 shows an example of the prompt we used to generate sentences with hop words. Additionally, the CHED dataset samples are shown in Figures 12 and 13 (Our contribution is from "sbj_hop_word" to "obj_new_hop_sentence").

A.7 Context Coherence Evaluation

To quantitatively assess context coherence, we conducted an additional evaluation using G-Eval with GPT-4o-mini (the exact instruction prompt is provided in Figure 8). Since most knowledge facts are incomplete sentences (e.g., "Tim Cook, who works for Apple"), making coherence judgments difficult, we first generated continuations for these facts to form complete sentences using GPT-4o-mini before measuring coherence.

The coherence scores ranged from 1 (poor coherence) to 5 (excellent coherence), with the following results on a 1–5 scale:

- s prefix context: 4.57
- o prefix context: 3.85
- $o*$ prefix context: 2.75
- s_{hop} prefix context: 3.37
- o_{hop} prefix context: 3.33
- $o*_{hop}$ prefix context: 2.80

The relatively low coherence scores for the edited-object contexts—both direct object-new (o^*) at 2.75 and hop-word–based object-new ($o*_{hop}$) at 2.80—are expected, because they rely on terms that are not naturally related to the original knowledge.

counts	1	2	3	4	5	Total
s_{hop}	763	1,340	1,304	1,320	17,055	97,910
o_{hop}	2	108	22	231	21,419	108,303
o^*_{hop}	2	129	35	273	21,343	108,172

Table 10: CHED Dataset Size

By contrast, the original-knowledge contexts (s at 4.57, o at 3.85) and their hop-word variants (s_{hop} at 3.37, o_{hop} at 3.33) all achieve above-middle coherence. The fact that s_{hop} and o_{hop} do not reach even higher levels is a consequence of our using low-frequency hop words to maximize distractiveness: infrequent, highly specific terms inherently make it harder to craft fully natural sentences. Nevertheless, we prioritized distractiveness in order to rigorously evaluate context robustness, and we judge these coherence levels to be acceptable.

A.8 Dataset Summary

While collecting the hop words from Wikidata, we found that some entities do not have enough full 5-hop words to form each prefix context. In CHED, 97% of instances have 5 prefix contexts associated with o_{hop} and o^*_{hop}, whereas only 77% of instances have the full set of 5 prefix contexts associated with s_{hop}. The relatively low number of *subject hop sentences* can be attributed to the nature of factual knowledge representation—where specific words (e.g., *"Danielle Darrieux"*) typically appear as subjects, whereas more general words (e.g., *"English"*) function as objects—resulting in different sentence counts across categories. Consequently, for prefix contexts using hop words, we constructed a dataset of **314,385** sentences based on **21,782** fact triplets. The details of the dataset size are provided in Table 10.

B Method Preliminaries

B.1 Full Derivation of $k(x)$

We compute $k(x)$ as follows:

$$k(x) = \sigma(W_{fc}\, a(x) + b_{fc}),$$

$$a(x) = \gamma\Big(\mathrm{Att}(h^{l-1}(x)) + h^{l-1}(x)\Big),$$

where $\sigma(\cdot)$ denotes a non-linear activation, and W_{fc}, b_{fc} are parameters of the MLP layer. Here, $h^{l-1}(x)$ is the hidden state at layer $l-1$, and $\mathrm{Att}(h^{l-1}(x))$ is the output of the attention mechanism applied to that hidden state. We then sum the attention output with the hidden state itself and normalize via $\gamma(\cdot)$. This process extracts the final MLP activation at the last token of the subject s.

B.2 Full KL-Divergence Term

Here, we expand the KL-divergence penalty $D_{\mathrm{KL}}(\mathbf{v})$ in Equation (2):

$$\mathbf{v}^* = \underset{\mathbf{v}}{\mathrm{argmin}} \frac{1}{N}\sum_{j=1}^{N}\Big[-\log \mathbb{P}_{G(h^l=\mathbf{v})}[o^* \mid x_j + p]\Big] + D_{\mathrm{KL}}\Big(\mathbb{P}_{G(h^l=\mathbf{v})}[x \mid p'] \,\Big\|\, \mathbb{P}_{G(h^l)}[x \mid p']\Big), \tag{8}$$

where $\mathbb{P}_{G(h^l=\mathbf{v})}[x \mid p']$ is the generation distribution under the modified hidden state $\mathbf{v}$, and $\mathbb{P}_{G(h^l)}[x \mid p']$ is the original distribution before the update. The second term minimizes the KL divergence between the output distributions for the probe prompt p' (*"{subject} is a"*) before and after the update, thereby preventing unintended changes to related knowledge.

C Analysis of Prefix Context

C.1 Analysis of Value Vector Variance Across Different Prefix Context Strategies

Figure 9 shows an extended version of the left panel in Figure 5, where the number of prefix prompts is plotted in finer detail. In this experiment, we assess whether different prefix context strategies yield greater diversity in value vectors by using 1,000 edit triplets from the CounterFact dataset. The value vectors, $\mathbf{v}$, are extracted from the third MLP layer of Llama-3-8B-Instruct. Specifically, each strategy is constructed as follows: for the s, o, and o^* strategies, sentences are generated exclusively using the corresponding word. For instance, in the s strategy, all sentences are generated solely with s (e.g., producing 6 sentences using s). In contrast, the s, o strategy forms a two-sentence set—one sentence using s and one using o—while the s, o, o^* strategy forms a three-sentence set with one sentence each generated using s, o, and o^*. In comparison, the *common words* strategy from MEMIT generates sentences by selecting words from a predetermined set (e.g., "The", "Therefore", "Because", "I", "You").

In the combined strategies, the total number of prefix contexts increases by 2 for the s, o strategy and by 3 for the s, o, o^* strategy, starting from 6 prefix contexts for the s strategy. Notably, even when using up to 18 prefix contexts, the overall variance does not increase significantly. Since increasing the

"You will be given two sentences: Sentence 1 and Sentence 2. Your task is to assess the coherence of Sentence 2 as a continuation of Sentence 1. Evaluate how logically and semantically coherent Sentence 2 is with the preceding sentence using the following scoring system:
Evaluation Criteria:
Coherence (1–5)
1 (Incoherent) – Sentence 2 is completely disconnected or nonsensical.
2 (Barely coherent) – Sentence 2 shows minimal connection, with major shifts.
3 (Moderately coherent) – Sentence 2 follows but has minor inconsistencies.
4 (Largely coherent) – Smooth continuation with only slight shifts.
5 (Highly coherent) – Perfect logical and semantic flow.
Evaluation Steps:
1. Read both Sentence 1 and Sentence 2 carefully.
2. Determine if Sentence 2 follows from Sentence 1 and maintains theme.
3. Assign a score of 1–5 based on the coherence criteria above."

Figure 8: Instruction Prompt for G-Eval

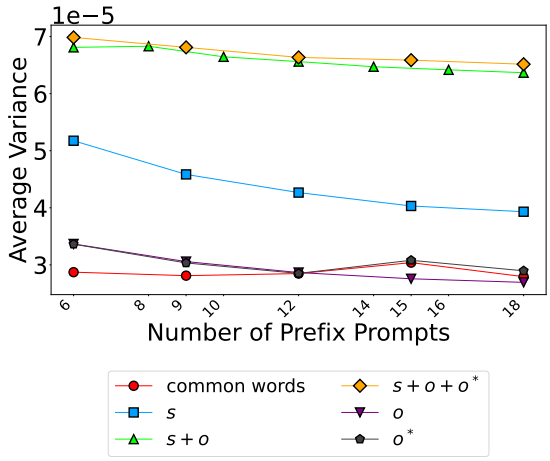


Figure 9: Average Variance of Value Vectors by Different Prefix Prompts Strategies

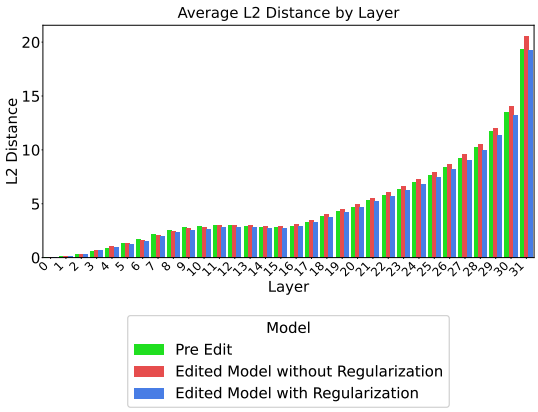


Figure 10: Average pairwise L2 distance by layer for the pre-edit model (green), post-edit without regularization (red), and post-edit with regularization (blue).

number of sentences does not significantly affect the variance, CoRE method uses 15 sentences (i.e., 5 sentences per word in the s, o, and o^* strategy).

C.2 Analysis of L2 Distance among Value Vectors across Prefix Contexts

Figure 10 shows an extended version of the right panel in Figure 5. Note that the underlying experimental values remain unchanged; what differs here is the presentation. In Figure 5, we focus on plotting the difference in pairwise L2 distances (post-edit versus pre-edit), whereas Figure 10 presents the exact average pairwise L2 distance values for the pre-edit model, the post-edit model without regularization, and the post-edit model with our regularization term.

For further experimental details, the prefix contexts used in these experiments are from the CHED dataset, with 15 distinct prefix contexts per edit triplet. For each input with a prepended prefix, we extracted the hidden state corresponding to the subject's last token (i.e., the value vector). Model editing was performed using MEMIT over 1,000 edits. As shown in the figure, the red bars represent the post-edit model without regularization, while the blue bars represent the post-edit model with the regularization term applied. Notably, the red bars demonstrate that, compared to the pre-edit model (represented by the green bars), the gap in L2 distances tends to increase in the later layers. In contrast, the addition of our regularization term (blue bars) effectively narrows this gap.

D Detailed Evaluation Metrics

In this appendix, we provide additional details and experimental setups for the evaluation metrics that are only briefly summarized in §5.1.

case_id: 162	**Prompt:** Guillaume Durand, who holds the position of
	Generated Text: pope, is a man of great authority and influence in the Catholic Church. He is the **bishop** of Rome and the spiritual leader of the worldwide Catholic Church. As such, he has the authority to make important decisions and to guide the Church in its teachings
case_id: 311	**Prompt:** The mother tongue of Jalil Lespert is
	Generated Text: Dutch. He is fluent in English and has a good understanding of **French** and German. Jalil Lespert studied industrial engineering at the University of Antwerp (UHasselt) and earned his Ph.D. in 1993. After his
case_id: 367	**Prompt:** In Colombia, they understand
	Generated Text: English, but it's still a good idea to learn some basic Spanish phrases. The official language is **Spanish**, and it's widely spoken, especially in the cities. In Peru, you'll find that many people speak Spanish, but you may also encounter
case_id: 958	**Prompt:** Greta Van Susteren works for
	Generated Text: BBC, becomes a British citizen Greta Van Suster, the former BBC anchor and **CNN** host, has become a British citizen. She has been working for the BBC since 2015 and has been a British resident since then. Suster,
case_id: 13839	**Prompt:** Melilla is a part of the continent of
	Generated Text: Europe, and is located in the northernmost part of Africa. It is a Spanish autonomous city located on the Mediterranean coast, and is the northernmost city in **Africa**. Melilla has a population of around 80,000 people, and is known

Table 11: Example cases illustrating the evaluation approach with N-gram Repetition scores.

D.1 Efficacy

As we mentioned, we propose a strict generation-based evaluation: an edit is considered successful only if the model's output (up to 50 tokens) includes o^* while completely excluding o. This approach prevents cases where the model initially produces o^* but later reverts to o, as detailed in Table 11.

Under this evaluation method, we evaluate edited knowledge using exact edit prompts (e.g. *"Tim Cook, who works for"*) in four different conditions. The baseline condition, *No ctx*, uses only the edit prompt without any additional context. The other three conditions—s_{hop}, o_{hop}, and o^*_{hop}—prepend different prefix contexts from our CHED dataset.

D.2 Generalization

Generalization extends the Efficacy metric by evaluating whether the model produces o^* when the edit prompt is paraphrased. For example, consider the paraphrased prompt *"Tim Cook, who is employed by"* as a variant of the original edit prompt.

D.3 Specificity

Specificity measures whether the knowledge that should remain unchanged is still the same after the edit, which is verified by asking about another subject that shares the same relation and object as in the edit prompt. For example, if the edit prompt involves a relation like "works for" with a particular object, we might ask about *"Kevan Parekh, who works for"*.

D.4 General Ability

To verify the model's fundamental capabilities after editing, we evaluate its performance across five key areas: commonsense reasoning, factual knowledge retrieval, context handling ability, multitask capabilities of language models across diverse subjects, and code generation. Specifically, we use CommonsenseQA (Talmor et al., 2019) for commonsense reasoning and TriviaQA (Joshi et al., 2017) for factual recall. We further assess long-context handling ability on the LAMBADA (Paperno et al., 2016), an open-ended cloze task requiring prediction of a held-out word given the full passage. We evaluate multitask capabilities using the MMLU (Massive Multitask Language Understanding) benchmark (Hendrycks et al., 2021), which measures language models' performance across 57 diverse subjects. Finally, LiveCodeBench evaluates four aspects—code generation, self-repair, test-output prediction, and code execution—but here we assess only its code-generation component using the 880 publicly released problems. Representative examples for datasets are presented in Tables 19, 20, 21 and 22. All evaluations were conducted using the OpenCompass (Contributors, 2023) framework.

D.5 Fluency

To ensure editing does not harm fluency, we measure the N-gram repetition as a proxy for disfluency. This metric is computed on outputs generated when the edit prompt is provided as input. Excessive repetition (see Table 12 for examples, where cases with drastically increased N-gram repetition are

highlighted) indicates degraded generation quality, so we penalize such outputs. Note that these examples were discovered when applying MEMIT to 1000 edit triplets. Furthermore, the N-gram repetition rate is computed as the sum of the repetition counts for 1-gram to 4-gram sequences.

case_id: 465	**Prompt:** Louis Bonaparte spoke the language
	Generated Text: Dutch Dutch language Dutch
	N-gram Repetition: 183
case_id: 396	**Prompt:** Pervez Musharraf follows the religion of
	Generated Text: Buddhism and has been the Ambassador of India to China. He has been the Ambassador of India to China. He has been the Ambassador of India to China. He has been the Ambassador of India to China. He has been the Ambassador of India to China.
	N-gram Repetition: 130
case_id: 888	**Prompt:** The capital of Second Spanish Republic is
	Generated Text: Paris. The capital of the French Republic is Paris. The capital of the Kingdom of France was Paris. The capital of the Kingdom of France was Paris. The capital of the Kingdom of France was Paris. The capital of the Kingdom of France was Paris
	N-gram Repetition: 117
case_id: 570	**Prompt:** Hannah Point is in
	Generated Text: Europe. The next city is Istanbul, Turkey. The next city is Istanbul, Turkey. The next city is Istanbul, Turkey. The next city is Istanbul, Turkey. The next city is Istanbul, Turkey.
	N-gram Repetition: 114
case_id: 329	**Prompt:** Angela Merkel was employed in
	Generated Text: Rome as the Prime Minister of Italy, but he was not the Prime Minister of Italy. He was the Prime Minister of Italy, but he was not the Prime Minister of Italy. He was the Prime Minister of Italy, but he was not the Prime.
	N-gram Repetition: 113

Table 12: Example cases illustrating the evaluation approach with N-gram Repetition scores.

edit prompt (original setting):
The new iPhone case I bought has a design that I really like. Tim Cook is employed by
Llama-3-8B-Instruct: <|begin_of_text|>
<|start_header_id|>user<|end_header_id|>
The new iPhone case I bought has a design that I really like<|eot_id|>
<|start_header_id|>assistant<|end_header_id|>
Tim Cook is employed by
Mistral-7B-Instruct-v0.3: <s>[INST] The new iPhone case I bought has a design that I really like[/INST]
Tim Cook is employed by

Figure 11: chat template examples

Method	Type	s	o	o*	s_chat	o_chat	o*_chat
				Llama3			
MEMIT	CHED	89.6	86.5	88.7	85.0	73.9	85.4
	random hop word	90.8	88.2	90.2	86.2	85.2	85.4
	random sentence	94.6	92.4	93.4	89.2	87.4	89.7
JEEP	CHED	68.2	62.6	69.5	64.6	56.7	71.5
	random hop word	63.0	61.3	60.5	67.0	60.2	66.0
	random sentence	66.2	64.0	65.4	66.4	65.5	63.7
PMET	CHED	70.2	66.0	77.3	64.7	54.2	75.0
	random hop word	73.0	69.9	72.7	68.3	63.3	67.4
	random sentence	74.9	71.8	72.8	68.7	66.2	66.5
EMMET	CHED	94.2	93.1	94.6	94.6	91.1	96.1
	random hop word	94.0	92.1	94.1	92.7	92.8	94.6
	random sentence	93.9	93.1	93.2	92.5	93.3	94.2
CoRE	CHED	95.1	93.8	96.6	91.2	84.9	94.6
	random hop word	94.9	94.5	96.7	92.6	90.1	91.9
	random sentence	96.7	95.4	96.5	93.4	92.2	93.5

Table 13: Efficacy with chat template and hop word ablation

E Analysis

E.1 Chat Template

In our study, we use chat templates to investigate the impact of our dataset, with each template designed differently across various models. Figure 11 presents an example of our chat template. During model experiments, we did not finalize the chat template to ensure that prompts aligned naturally with the model's generation process. Also, Table 13 presents the impact of the chat template and the results of our ablation study. In Llama3, which can adapt to the chat template, we observe a decrease in the efficacy for original object sentences across all methods. This indicates that LLMs are generally influenced by the template. A particularly noticeable decline occurs with o_{hop} prefix context: MEMIT's success rate drops from 86.5% to 73.9%, and CoRE's rate decreases from 93.8% to 84.9%. Conversely, success rate with o^*_{hop} prefix context results in a less pronounced decrease.

E.2 Hop Words Anaylsis

Our ablation study on hop words confirms their significant impact on the efficacy. Specifically, replacing words with hop words leads to a greater decrease in success rate compared to using random words. Moreover, the effect of hop words is com-

Method	Type	s	o	o*	s_chat	o_chat	o*_chat
				Llama3			
MEMIT	CHED	76.3	70.6	72.0	71.4	63.9	68.4
	random hop word	73.1	70.0	71.5	70.8	67.1	68.2
	random sentence	80.0	78.0	80.7	73.0	71.8	72.9
JEEP	CHED	55.3	50.0	54.7	54.6	48.6	56.8
	random hop word	48.5	44.9	46.2	53.0	48.3	51.0
	random sentence	53.7	51.5	51.7	53.3	52.2	51.6
PMET	CHED	64.6	58.7	66.4	56.2	49.1	58.5
	random hop word	63.1	59.4	62.2	58.3	52.3	54.6
	random sentence	66.7	62.9	65.1	57.9	54.9	55.3
EMMET	CHED	82.2	79.6	80.7	86.6	83.9	85.8
	random hop word	80.0	77.5	79.1	84.7	82.2	83.9
	random sentence	80.9	79.5	79.8	84.7	84.2	84.8
CoRE	CHED	92.0	89.7	90.5	86.8	81.7	84.2
	random hop word	90.8	88.7	90.1	85.4	82.4	83.0
	random sentence	93.1	92.1	92.3	86.8	85.3	86.3

Table 14: Average probability of various methods

parable to replacing entire sentences at random, suggesting that the primary influence of the CHED dataset stems from the hop words themselves.

As we expected, the main contribution of our CHED dataset comes from the contextual hop-word. If we look at the o_{hop_chat} column of Table 13, we observe the most significant difference in o_{hop} and o_{hop_chat} contexts, particularly when used with chat templates. In the CHED dataset, the MEMIT method shows an increase in success rate from 73.9% to 85.2% when using a random hop word, which is close to the 87.4% observed in fully random contexts. Similarly, the CoRE method follows the same pattern, increasing from 84.9% to 90.1% with a random hop word, which is comparable to the 92.2% achieved with fully random contexts. These results suggest that hop words act as key elements that distract the model's attention, leading it to recall the original object despite the applied knowledge edit.

E.3 Probability Test

In some studies, the outcome of knowledge editing is also evaluated by examining the probability difference between the original and new object tokens, thereby capturing the intrinsic differences between the two objects that are not simply generated by the language models. Accordingly, we conducted several experiments to assess not only the efficacy but also the probability of the new object token for Llama3. Especially, our method CoRE almost outperforms other methods, except for EMMET, which has a lower generalization score in the experiments. The results are presented in Table 14.

E.4 ACS

Recent studies suggest that *information flow*, particularly the attention from the *subject token* to the *last token* of the sentence, plays a crucial role in LLM's generative performance (Geva et al., 2023). Based on this, we further investigated the influence of hop words on knowledge editing performance by measuring the Average Contribution Score (ACS). If a hop word spans multiple tokens, we compute its total impact by summing the contributions of each constituent token.

Method	s	s_chat	o	o_chat	o*	o*_chat	random	random_chat
				Llama3				
No edit	0.549	0.624	0.618	0.689	0.671	0.715	0.406	0.537
JEEP	0.662	0.692	0.727	0.776	0.738	0.775	0.563	0.650
PMET	0.602	0.654	0.698	0.745	0.736	0.764	0.488	0.602
MEMIT	0.549	0.647	0.591	0.697	0.685	0.751	0.447	0.608
EMMET	0.544	0.553	0.543	0.580	0.587	0.638	0.389	0.474
CoRE	0.552	0.608	0.578	0.659	0.688	0.745	0.466	0.580
				gpt-j				
No edit	0.514	-	0.616	-	0.684	-	0.491	-
JEEP	0.438	-	0.498	-	0.584	-	0.345	-
PMET	0.474	-	0.529	-	0.617	-	0.379	-
MEMIT	0.460	-	0.584	-	0.662	-	0.422	-
EMMET	0.439	-	0.525	-	0.637	-	0.383	-
CoRE	0.430	-	0.529	-	0.635	-	0.394	-

Table 15: ACS of the various model and methods

Table 15 presents the total ACS of Llama3 and GPT-J, using various editing methods. In our CHED dataset, the average sentence length is 14.39 tokens, while the average length of hop words is 3.04 tokens. This means that when the model attends to every token randomly, the ACS with random tokens is about 0.21. As discussed in section §5.6, our model achieves a decrease in the original object's ACS and an increase in the new object's ACS in both the no-template and chat-template settings. In contrast, other methods generally exhibit either a decrease in both or an increase in both.

Notably, the CoRE method uniquely demonstrates this tendency in both simple prefix and user utterance contexts, whereas other methods achieve ACS values that are either too high, meaning they pay excessive attention to outdated o_{hop} information, or too low, indicating that they disregard the o^*_{hop} information after knowledge editing.

We also observe that all ACS values are higher when the prefix context is prepended as a user utterance. This indicates that the model pays more attention to the hop word, which comes from the user, suggesting that large language models extract more information from user-provided texts. Additionally, we can observe that the model achieved

an increasing ACS for o^*_{hop} and a decreasing ACS for o_{hop} after editing, which further validates our expectations.

We speculate that this result was achieved because our CoRE method uses multiple context sentences to guide the model on which token of the context it should focus for the newly edited knowledge.

For GPT-J, we did not observe a significant difference in model behavior after editing, as the ACS decreased across all methods. We speculate that this phenomenon occurs because GPT-J is less powerful than Llama3, making it less robust to model editing. As a result, it loses its internal generality after editing.

From this, we can conclude that the attention score can be used to diagnose a model's differences after applying editing methods—not only in terms of probability or generation efficacy but also in understanding the model's internal mechanisms.

F Implementation Details

All experiments are conducted on NVIDIA A100 GPUs. Model inference was performed using vLLM (Kwon et al., 2023), while the probabilistic experiments were carried out using HuggingFace.

F.1 Mass-Editing Memory In a Transformer (MEMIT)

On Llama3 and Mistral, MEMIT hyperparameters follow those used for Llama2-7b in the EasyEdit open source code (Wang et al., 2024), as they share similar architecture, size, and number of layers. Optimization updates are executed for 25 steps with a weight decay of 1×10^{-3}, a KL factor of 0.0625, and a learning rate of 5×10^{-1}. Training is conducted in `fp32`, while evaluation is performed in `fp16`.

Following the same EasyEdit open source code as described above, for GPT-J-6B the EasyEdit hyperparameters are configured such that optimization updates are carried out for 25 steps with a weight decay of 0.5, a KL factor of 0.0625, and a learning rate of 5×10^{-1}.

We further investigated the selection of layers for editing. While earlier work (Meng et al., 2022b) employed causal tracing to pinpoint optimal layers, later studies have shown that layers identified by causal tracing do not always lead to the best editing performance (Hase et al., 2023). Motivated by these findings, we revisited the layer selection process by focusing on the early-to-mid layers. Building on prior work (Gupta et al., 2024b; Yoon et al., 2024), we experimented with subsets consisting of 1, 2, 3, or 4 layers. For each subset, we evaluated performance based on three normalized metrics—Efficacy (no-context), General Ability, and N-gram Repetition—and computed an average score. This evaluation led us to select the following layers for editing: MEMIT: $[3]$, Mistral-7b: $[4, 5]$, and GPT-J: $[2, 3, 4]$.

F.2 Context Robust Editing (CoRE)

For fairness, we use the same hyperparameters as those employed in MEMIT (*see §F.1*). Our method builds on these settings by incorporating an additional regularization term. In this term, the layer range and the scaling factor—denoted as $\mathcal{L}$ and λ respectively in Equation 4—were determined via parameter search using the same approach as that employed for layer selection in MEMIT.

In our experiments, we explored three configurations for the layer range: the 10 layers immediately following the edited layer, the 20 layers immediately following it, and all layers until the end of the model. Specifically, for Llama3, the chosen configuration was the 28 layers following the edited layer (layer 3) with a scaling factor of 0.04. For Mistral, the layer range comprised the 26 layers after the last edited layer (layer 5) with a scaling factor of 0.1. For GPT-J, the layer range consisted of the 26 layers following the last edited layer (layer 4) with a scaling factor of 0.0002. The scaling factor was initially explored from 1, decrementing by 0.1. For GPT-J, since no suitable parameter was found in the initially explored range, we further refined the search starting from 0.1 in decrements of 0.01, and then from 0.01 in decrements of 0.0001. We observed a consistent trend: as the scaling factor increased, the editing success in the no-context setting tended to decrease, while metrics such as General Ability and N-gram Repetition improved.

F.3 Equality-contrained Mass Model Editing algorithm for Transformers (EMMET)

On Llama3 and Mistral, EMMET hyperparameters follow those used for Llama2-7b in the EMMET open source code (Gupta et al., 2024b), as they share the similar architecture, size, and number of layers. Updates are executed at layer 5, where optimization proceeds for 25 steps with a weight decay of 1×10^{-3}, KL factor of 0.0625, and learning rate of 5×10^{-1}. EMMET applies an emmet lambda

Base Model	Method	Efficacy							Generalization	Specificity	General Ability		Fluency
		No ctx	s	o	o*	s_{hop}	o_{hop}	o^*_{hop}			C-QA	T-QA	N-gram
GPT-J	Base	0.9	1.6	0.54	38.06	1.2	1.0	10.6	1.1	26.1	21.5	32.7	7.6
	MEMIT	92.8	77.26	48.9	85.8	75.5	69.3	81.4	64.2	26.3	21.9	31.9	7.3
	JEEP	84.9	75.64	54.88	84.7	74.7	70.5	82.3	63.9	**27.2**	20.8	31.0	**7.1**
	PMET	90.4	79.84	59.54	88.82	81.2	76.6	86.9	70.4	26.8	20.0	31.9	7.3
	EMMET	**95.3**	**81.4**	**61.14**	**91.1**	**83.6**	**79.2**	**89.1**	**73.5**	21.8	19.9	29.7	7.2
	FT-M	32.9	28.6	26.44	24.28	26.4	24.2	23.3	17.0	12.3	19.2	5.9	60.5
	CoRE-p	94.3	79.32	54.31	88.49	81.6	76.0	85.1	66.3	24.7	**22.0**	**32.0**	7.2
	CoRE-p+r	93.8	80.68	58.96	89.54	81.5	76.7	85.2	68.7	24.8	21.9	**32.0**	**7.1**

Table 16: Results on GPT-J

Model	Method	Efficacy No ctx	Gen	Spe	Avg	Fluency N-gram
Llama3	Base	2.7	3.3	30.3	-	15.4
	MEMIT	48.7	44.6	28.6	40.6	26.6
	JEEP	29.9	19.5	23.8	24.4	27.2
	PMET	43.5	29.2	29.4	34.0	**24.7**
	FT-M	49.5	45.1	1.0	31.9	78.8
	CoRE	**50.0**	**46.0**	**30.2**	**42.1**	26.6
Mistral	Base	1.4	2.0	23.0	-	4.8
	MEMIT	40.2	35.4	20.8	32.2	**5.4**
	JEEP	20.8	14.4	20.6	18.6	6.7
	PMET	41.2	28.9	**23.3**	31.1	5.8
	FT-M	**48.9**	**38.2**	10.4	**32.5**	16.6
	CoRE	40.5	35.4	19.9	32.0	**5.4**
GPT-J	Base	1.2	0.7	2.9	-	7.6
	MEMIT	55.0	37.4	3.5	32.0	8.8
	JEEP	**66.8**	36.5	3.4	**35.6**	**8.3**
	PMET	60.8	**37.8**	2.8	33.8	9.6
	EMMET	59.6	31.2	2.3	31.0	8.9
	FT-M	15.6	11.2	1.8	9.5	41.6
	CoRE	53.1	36.7	**3.8**	31.2	9.4

Table 17: Results on zsRE

of 0.1. Training is conducted in `fp32`, while evaluation is performed in `fp16`.

Following the same EMMET open source code as described above, for the EMMET hyperparameters are configured such that updates are executed at layer 5. Optimization is carried out for 25 steps with a weight decay of 0.5, a KL factor of 0.0625, and a learning rate of 5×10^{-1}. Additionally, an emmet lambda of 0.1 is applied.

.

F.4 Joint knowledge editing for information Enrichment and probability Promotion (JEEP)

JEEP hyperparameters follow those used for Llama2-7b in the JEEP open source code (Shi et al., 2024), as Llama3 and Mistral share the similar architecture, size, and number of layers. Updates are executed at layers low $[5]$ and layers high $[22, 23, 24]$, where optimization proceeds for 30 steps with a learning rate of 0.5. Weight decay and KL factor are set differently for each layer range: weight decay low is 0.5 with KL factor low of 0.0625, while weight decay high is 0.5 with KL factor high of 0. Training is conducted in `fp32`, while evaluation is performed in `fp16`.

Based on the same open source code, hyperparameters are configured to update lower layers $[3, 4, 5, 6, 7, 8]$ and higher layers $[15, 16]$. Optimization proceeds for 30 steps with a weight decay of 0.5, a KL factor of 0.0625 for lower layers and 0 for higher layers, and a learning rate of 5×10^{-1}. Additionally, a moment adjustment weight of 2000 is applied across both layer ranges.

F.5 Precise Model Editing in a Transformer (PMET)

Similar to JEEP, PMET hyperparameters follow those used for Llama2-7b in the JEEP open source code (Li et al., 2023), as LLlama3 and Mistral share the similar architecture, size, and number of layers. Updates are executed at layers $[5, 6, 7, 8, 9, 10]$, where optimization proceeds for 20 steps with a weight decay of 0.5, KL factor of 1.0, and learning rate of 0.1. PMET applies an NLL loss factor of 2.0. Training is conducted in `fp32`, while evaluation is performed in `fp16`.

For , PMET hyperparameters are configured to update layers $[3, 4, 5, 6, 7, 8]$, where optimization proceeds for 30 steps with a weight decay of 0.5, KL factor of 1.0, and a learning rate of 2×10^{-1}. PMET applies an NLL loss factor of 1.0. Additionally, a moment adjustment weight of 6000 is applied. Training is conducted in `fp32`, while evaluation is performed in `fp16`.

F.6 FT-M

FT-M (Zhang et al., 2024b) improves upon the direct fine-tuning approach (FT-L) by training the same FFN layer, identified via causal tracing in ROME, using cross-entropy loss on the target answer with the original text masked.

FT-M hyperparameters follow those used in the

Table 18: Comparison of generation-based and probability-based evaluation metrics for MEND and IKE on Llama-3-8B-Instruct.

Method	Generation-based (%)			Probability-based (%)		
	No ctx	*Gen*	*Spe*	No ctx	*Gen*	*Spe*
MEND	1.8	0.7	1.3	51.5	49.3	50.1
IKE	0.4	0.3	37.7	68.8	72.5	80.3

EasyEdit open source code (Wang et al., 2024). Training is conducted in `fp32`, while evaluation is performed in `fp16`. Updates are executed at layers [21]. Optimization updates are performed over 25 steps with a learning rate of 5×10^{-4}.

F.7 Methods Excluded for Low Efficacy

In our work (see Appendix D and Table 18), we adopt a **generation-based** efficacy metric: an edit is deemed successful only if the model actually outputs the new object (o^*) and does **not** output the original object (o) within a 50-token window. By contrast, the MEND and IKE papers employ a **probability-based** criterion, which counts an edit as successful whenever the model assigns a higher probability to o^* than to o, regardless of whether either string is ever generated.

Under 1,000 mass edits and using a stricter, more realistic generation-based evaluation (see Table 18), MEND achieves only **1.8%** efficacy in the no-prefix setting, despite scoring **51.5%** under the probability-based protocol. Similarly, IKE—a prompt-based editor—manages just **0.4%** generation efficacy, even though it reaches **68.8%** by probability-based scoring (Table 18). Because these near-zero generation results indicate almost complete editing failure under realistic conditions, we excluded them from Table 4.

Similarly, we omit the EMMET method on Llama3, Mistral on the zsRE dataset. Llama3 achieves only a 0.1% efficacy, despite scoring 57.7% in the previous efficacy calculations. Mistral also achieves only a 0.0% efficacy, despite scoring 49.9% in the previous efficacy.

ID	Input Prompt & Gold Answers
0	[HUMAN: “The sanctions against the school were a punishing blow, and they seemed to what the efforts the school had made to change? A. ignore B. enforce C. authoritarian D. yell at E. avoid Answer:”] BOT: “A” HUMAN: “A revolving door is convenient for two direction travel, but it also serves as a security measure at a what? A. bank B. library C. department store D. mall E. new york Answer:” **Gold Answer:** A
1	[HUMAN: “The sanctions against the school were a punishing blow, and they seemed to what the efforts the school had made to change? A. ignore B. enforce C. authoritarian D. yell at E. avoid Answer:”] BOT: “A” HUMAN: “What do people aim to do at work? A. complete job B. learn from each other C. kill animals D. wear hats E. talk to each other Answer:” **Gold Answer:** A
2	[HUMAN: “The sanctions against the school were a punishing blow, and they seemed to what the efforts the school had made to change? A. ignore B. enforce C. authoritarian D. yell at E. avoid Answer:”] BOT: “A” HUMAN: “Where would you find magazines along side many other printed works? A. doctor B. bookstore C. market D. train station E. mortuary Answer:” **Gold Answer:** B

Table 19: Example of the CommonsenseQA dataset with 1-shot setting

ID	Input Prompt & Gold Answers
0	[HUMAN: “Answer the question, your answer should be as simple as possible, start your answer with the prompt ‘The answer is ’. Q: Who was the man behind The Chipmunks??”] BOT: “The answer is [‘David Seville’].” HUMAN: “Answer the question, your answer should be as simple as possible, start your answer with the prompt ‘The answer is ’. Q: Who was the man behind The Chipmunks??” **Gold Answers:** David Seville
1	[HUMAN: “Answer the question, your answer should be as simple as possible, start your answer with the prompt ‘The answer is ’. Q: Who was the man behind The Chipmunks??”] BOT: “The answer is [‘David Seville’].” HUMAN: “Answer the question, your answer should be as simple as possible, start your answer with the prompt ‘The answer is ’. Q: What star sign is Jamie Lee Curtis??” **Gold Answers:** Scorpio, Skorpio, Scorpio (disambiguation)
2	[HUMAN: “Answer the question, your answer should be as simple as possible, start your answer with the prompt ‘The answer is ’. Q: Who was the man behind The Chipmunks??”] BOT: “The answer is [‘David Seville’].” HUMAN: “Answer the question, your answer should be as simple as possible, start your answer with the prompt ‘The answer is ’. Q: Which Lloyd Webber musical premiered in the US on 10th December 1993??” **Gold Answers:** Sunset Blvd, West Sunset Boulevard, Sunset Boulevard, Sunset Bulevard, Sunset Blvd.

Table 20: Example of the TriviaQA dataset with 1-shot setting

ID	Input Prompt & Gold Answers
0	Please complete the following sentence: In my palm is a clear stone, and inside it is a small ivory statuette. A guardian angel. "Figured if you're going to be out at night getting hit by cars, you might as well have some backup." I look at him, feeling stunned. Like this is some sort of sign. But as I stare at Harlin, his mouth curved in a confident grin, I don't care about **Gold Answer:** signs
1	Please complete the following sentence: Give me a minute to change and I'll meet you at the docks." She'd forced those words through her teeth. "No need to change. We won't be that long." Shane gripped her arm and started leading her to the dock. "I can make it there on my own, **Gold Answer:** Shane
2	Please complete the following sentence: "Only one source I know of that would be likely to cough up enough money to finance a phony sleep research facility and pay people big bucks to solve crimes in their dreams," Farrell concluded dryly. "What can I say?" Ellis unfolded his arms and widened his hands. "Your tax dollars at work." Before Farrell could respond, Leila's voice rose from inside the house. "No insurance?" she wailed. "What do you mean you don't have any **Gold Answer:** insurance

Table 21: Example of the LAMBADA dataset

ID	Input Prompt

0	### Question: You are given a binary string s of length n and an integer numOps.
	You are allowed to perform the following operation on s at most numOps times:
	Select any index i (where 0 <= i < n) and flip s[i]. If s[i] == '1', change s[i] to '0' and vice versa.
	You need to minimize the length of the longest substring of s such that all the characters in the substring are identical.
	Return the minimum length after the operations.
	Example 1:
	Input: s = 000001, numOps = 1
	Output: 2
	Explanation:
	By changing s[2] to '1', s becomes 001001. The longest substrings with identical characters are s[0..1] and s[3..4].
	Example 2:
	Input: s = 0000, numOps = 2
	Output: 1
	Explanation:
	By changing s[0] and s[2] to '1', s becomes 1010.
	Example 3:
	Input: s = 0101, numOps = 0
	Output: 1
	Constraints: 1 <= n == s.length <= 1000
	s consists only of '0' and '1'.
	0 <= numOps <= n
	### Format: You will use the following starter code to write the solution to the problem and enclose your code within delimiters.
	python
	class Solution:
	def minLength(self, s: str, numOps: int) -> int:

Table 22: Example of the LiveCodeBench dataset

“case_id”: “6”,
“counterfact_id”: “6”,
“prompt”: “, that was created in”,
“subject”: “Anaal Nathrakh”,
“fact_knowledge”: “Birmingham”,
“edited_knowledge”: “Philadelphia”,
“relation_id”: “P740”,
“rephrased_prompt”: “In Wardha he came in close contact with Mahatma Gandhi. Anaal Nathrakh was founded in”,
“locality_prompt”: “City of Birmingham Symphony Orchestra, that was created in”,
“locality_ground_truth”: “Birmingham”,
“sbj_hop_word”: [“Back on Black Records”, “black metal”, “Season of Mist”, “Candlelight Records”, “United Kingdom”],
“obj_old_hop_word”: [“Yvonne Mosquito”, “River Tame”, “Changchun”, “GBBHM”, “ West Midlands”],
“obj_new_hop_word”: [“Darby”, “Jim Kenney”, “Riverton”, “USPHL”, “Lower Moreland Township”],
“sbj_hop_sentence”: [“The label was founded to support underground artists, Back on Black Records.”, “This genre is characterized by its intense sound and themes, black metal.”, “The label expanded its roster significantly over the years, Season of Mist.”, “Artists under this label have gained international recognition, Candlelight Records.”, “The music scene in that area has a distinct identity, United Kingdom.”],
“obj_old_hop_sentence”: [“Yvonne Mosquito first appeared in various documentaries discussing tropical diseases.”, “Residents often enjoy the beauty of the River Tame throughout the year.”, “Changchun is famous for its advanced automotive industry in Asia.”, “The recent events highlighted the importance of GBBHM initiatives for urban development.”, “Numerous attractions can be found in the West Midlands region.”],
“obj_new_hop_sentence”: [“The quaint town of Darby is known for its friendly community.”, “Under Mayor Jim Kenney, the city has seen significant changes.”, “Located near the river, Riverton offers beautiful waterfront views.”, “The USPHL provides a platform for aspiring hockey players to showcase their talent.”, “Lower Moreland Township features several parks and recreational facilities.”]

Figure 12: Example of the CHED-1

“case_id”: “5644”,
“counterfact_id”: “5698”,
“prompt”: “ from”,
“subject”: “Ronan Keating”,
“fact_knowledge”: “Australia”,
“edited_knowledge”: “Bangladesh”,
“relation_id”: “P495”,
“rephrased_prompt”: “Track listing Chart References Category:2012 albums Category:Garou (singer) albums Ronan Keating was developed in”,
“locality_prompt”: “The Slap, formulated in”,
“locality_ground_truth”: “Australia”,
“sbj_hop_word”: [“songwriter”, “Boyzone”, “Westlife”, “voice”, “singer”],
“obj_old_hop_word”: [“Karuwali”, “Andajin”, “Nyamal”, “Dhungaloo”, “Avstralka”],
“obj_new_hop_word”: [“East Bengal”, “Dhaka Division”, “Usui”, “Oraon Sadri”, “bengalese”],
“sbj_hop_sentence”: [“A talented songwriter crafted lyrics that resonated with many listeners.”, “Boyzone became famous for their emotional ballads and captivating performances.”, “Westlife captured hearts with their harmonious melodies and stunning vocal arrangements.”, “Her voice captivated everyone in the studio during the recording session.”, “As a singer, she expressed deep emotions through her powerful performances.”],
“obj_old_hop_sentence”: [“Karuwali is celebrated for its vibrant festivals held throughout the year.”, “Andajin residents often gather at the marketplace to share local news.”, “Nyamal stories highlight the connection between the land and its people.”, “Dhungaloo offers breathtaking views that attract many nature enthusiasts each season.”, “Avstralka has a diverse ecosystem that fascinates ecologists from around the world.”],
“obj_new_hop_sentence”: [“The history of East Bengal is rich with cultural diversity and evolution.”, “Dhaka Division is known for its vibrant markets and bustling streets.”, “In Japan, the art of Usui Reiki promotes healing through energy exchange.”, “The Oraon Sadri community holds unique traditions that reflect their heritage.”, “The Bengalese, known for their distinct language, contribute to the region’s cultural tapestry.”]

Figure 13: Example of the CHED-2