EVERYTHING IS EDITABLE: EXTEND KNOWLEDGE EDITING TO UNSTRUCTURED DATA IN LARGE LANGUAGE MODELS

Anonymous authors
Paper under double-blind review

ABSTRACT

Recent knowledge editing methods have primarily focused on modifying structured knowledge in large language models. However, this task setting overlooks the fact that a significant portion of real-world knowledge is stored in an unstructured format, characterized by long-form content, noise, and a complex yet comprehensive nature. Techniques like "local layer key-value storage" and "term-driven optimization", as used in previous methods like MEMIT, are not effective for handling unstructured knowledge. To address these challenges, we propose a novel **Un**structured **K**nowledge **E**diting method, namely UnKE, which extends previous assumptions in the layer dimension and token dimension. Firstly, in the layer dimension, we propose non-local block key-value storage to replace local layer key-value storage, increasing the representation ability of key-value pairs and incorporating attention layer knowledge. Secondly, in the token dimension, we replace "term-driven optimization" with "cause-driven optimization", which edits the last token directly while preserving context, avoiding the need to locate terms and preventing the loss of context information. Results on newly proposed unstructured knowledge editing dataset (UnKEBench) and traditional structured datasets demonstrate that UnKE achieves remarkable performance, surpassing strong baselines. In addition, UnKE has robust batch editing and sequential editing capabilities. The code is available in the anonymous repository: https://anonymous.4open.science/r/UnKE-BC5F.

1 INTRODUCTION

Ensuring the accuracy and timeliness of the knowledge stored in large language models (LLMs) is crucial, especially given their widespread deployment Xu et al. (2023); Chen & Shu (2023; 2024). To address this challenge, knowledge editing (Yao et al., 2023; Zhang et al., 2024b; Cheng et al., 2023; Mao et al., 2023) has emerged as a promising approach, enabling timely updates to the knowledge embedded in LLMs.

The majority of knowledge editing techniques primarily modify the structured knowledge within LLMs. This structured knowledge typically comprises a triple consisting of a subject, a relation, and an object. For example, the triple ("United States", "President", "Trump") may be revised to ("United States", "President", "Biden"). However, approximately 80% of real-world knowledge is contained in unstructured formats (Bavota, 2016). For instance, when posed with a question such as "What were Charles Strachey's key contributions to British politics and law in the 19th century?", the desired answer is an informative and free-form text (refer to Table 13 for specifics), as opposed to a mere entity. Furthermore, when using LLMs, users typically seek comprehensive text output rather than simple entity-level representations. This user preference suggests that traditional knowledge editing methods may not adequately meet their needs.

Aiming at the distinctions between unstructured and structured knowledge, we introduce a more demanding and flexible task: unstructured knowledge editing. This task presents two significant challenges to existing knowledge editing methodologies: (1) Unstructured knowledge contains richer information. Specifically,

findings in Section 3.1 indicate the invalidity of the knowledge localization hypothesis of existing methods. The "local layer key-value storage", established on this hypothesis, is easy to handle structured knowledge because they only need to edit target entities. However, this approach proves inadequate for handling unstructured knowledge with complex context, relational information, and a large number of entities (see Appendix I). (2) Unstructured knowledge is presented in a more free form. In particular, existing methods typically rely on term localization for editing, a process that can be called "term-driven optimization". Omitting this crucial step significantly diminishes their efficacy, as demonstrated in the experiments outlined in Section 3.2. However, locating these terms within unstructured text poses a significant challenge, as illustrated by the case discussed in Table 13. Also, only editing the term tokens of autoregressive LLMs without considering contextual information can result in the loss of information, as illustrated in Figure 1 and discussed in Section 4.1.

To bridge this gap, in this paper, we introduce an **Un**structured **K**nowledge **E**diting (UnKE) method that leverages a combination of techniques "non-local block key-value storage" and "cause-driven optimization". As shown in Figure 1, specifically, we argue that unstructured knowledge is not strictly limited to particular (local) MLP layers or knowledge neurons, but is distributed collaboratively across multiple layers (non-local). To this end, we expand previous hypotheses in two dimensions. Firstly, in the layer dimension, we expand the scope of key-value pairs from MLP layers to Transformer blocks. More precisely, we view the shallow and deep layers of LLMs as key and value generators, respectively. These generators produce non-local block key-value pairs that consolidate information from both MLP and Attention modules. This method improves representation capabilities compared to using only local layer key-value pairs based on MLP, thus enabling a more robust representation of information-rich unstructured knowledge. Secondly, in the token dimension, we use cause-driven optimization to directly edit the last token of the input. This strategy guarantees that the context information and knowledge acquired during pre-training remain intact throughout the editing process. By doing so, we eliminate the need for the "localization term" operation and prevent the loss of information (refer to Figure 1).

To address the lack of a benchmark for editing unstructured knowledge, we develop UnKEBench, which is more challenging than existing structured editing benchmarks due to its complexity. UnKE significantly outperforms existing baselines across several evaluation metrics within UnKEBench, showcasing its ability to accurately handle information-rich and free-format unstructured knowledge. Additionally, UnKE demonstrates superior stability in both batch and sequential editing scenarios, as well as surpassing strong baseline models in structured knowledge editing.

2 Related Works

In this section, we introduce recent advancements in knowledge editing, which can be broadly categorized into three groups: methods that preserve the original model parameters, methods that locate and then edit the original model parameters, and methods that directly modify the original model parameters.

Preserving Model Parameters One category focuses on introducing additional parameters, while the other focuses on involving knowledge in in-context learning (ICL). For adding parameters, SEARC (Mitchell et al., 2022b) utilizes a classifier to differentiate between input that requires editing and input that does not. If editing is necessary, the trained counterfactual model is employed for prediction; otherwise, using the original model. T-Patcher (Huang et al., 2023) incorporates and trains specific neurons in the final feedforward network layer for the sample that requires editing, e.g. their functionality activated solely when encountering the edited sample. Additionally, (Hartvigsen et al., 2023) proposed GRACE, a lifelong model editing method that generates a discrete local editing codebook while preserving the model weights unchanged. While training additional parameters may be effective for editing knowledge triples, their success with unstructured knowledge is limited by the number of parameters. For ICL, IKE (Zheng et al., 2023) utilizes ICL for

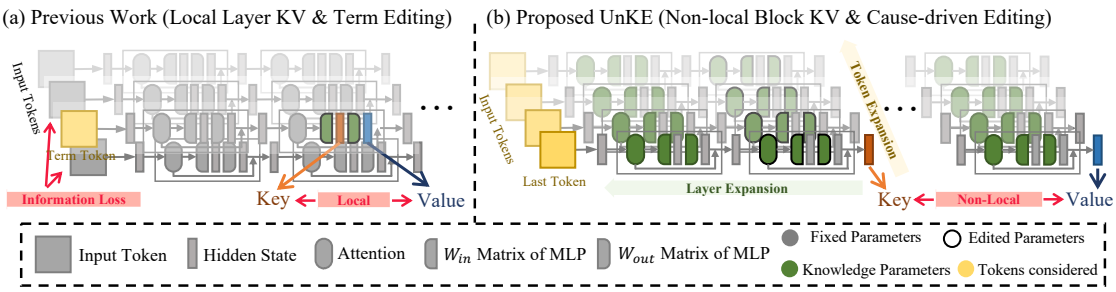

Figure 1: Comparison of UnKE with previous knowledge editing methods. Previous studies assumed that knowledge is stored in the form of key-value pairs in local MLP layers and edited according to specific term positions, such as subjects. However, this Local Layer KV has difficulty representing information-rich unstructured knowledge, and only editing specific terms will cause information loss. In contrast, UnKE uses a non-local block KV produced by transformer layers and considers the positions of all input tokens during the editing process. Compared with previous methods, it solves the above problems and shows excellent unstructured knowledge editing capabilities.

knowledge editing, while MeLLo (Zhong et al., 2023) enhances multi-hop knowledge editing capabilities by decomposing complex multi-hop problems into sub-problems and integrating them with retrieval techniques. However, both methods face challenges in efficiently editing a large amount of knowledge within a single model, primarily due to limitations in parameter count and context window length, especially for unstructured knowledge with verbosity, noise, and interdependencies.

Locate-Then-Edit Another branch of methods adopts a locate-and-edit approach. Initially, they identify the specific parameters associated with the target knowledge and subsequently modify those parameters directly to effectuate the desired knowledge editing. KN (Dai et al., 2022) introduces the concept of knowledge neurons and utilizes them to incorporate specific factual knowledge without the need for fine-tuning. ROME (Meng et al., 2022) introduces a causal tracking method to identify the layer that requires editing. Subsequently, it employs Rank-One Model Editing to modify the weights of the feedforward layer, thereby updating specific factual associations. MEMIT (Meng et al., 2023) is an enhanced version of ROME, capable of editing knowledge in batches. These methods operate under the assumption that knowledge is stored locally within MLP layers or neurons, which proves inadequate when confronted with unstructured knowledge.

Directly Modify Model Parameters Additionally, there exist numerous other methods that enable knowledge editing by directly modifying model parameters without the need for explicit positioning. MEND (Mitchell et al., 2022a) introduces auxiliary networks and enables scalable editing by decomposing gradients, thereby facilitating efficient and effective knowledge editing. To enhance the stability and effectiveness of knowledge editing in large language models, StableKE (Wei et al., 2024c) employs additional knowledge for fine-tuning, presenting an approach that brings about significant improvements. As knowledge transitions from a structured to an unstructured format, the process of editing them becomes time-consuming, leading to a degradation in performance.

3 MOTIVATIONS

To investigate why conventional knowledge editing techniques are inadequate for editing unstructured knowledge, we carry out pertinent experiments and determined that: (1) the hypothesis that knowledge is locally stored is unsuitable for information-rich unstructured knowledge and (2) term-driven optimization is notably sensitive to specialized terms, however it is difficult to locate them in unstructured knowledge.

3.1 LLMs Store Knowledge Non-locally

Hypothesis 1: *Knowledge is stored in specific local parameters of LLMs.* We refute this hypothesis through a contradiction approach. Initially, methods such as ROME and MEMIT utilize causal tracing, believing that knowledge is localized within the early MLP layers, thus targeting these layers for editing. Employing MEMIT, we conduct experiments to edit structed and unstructed knowledge across various layers using the Counterfact dataset (Meng et al., 2022) and UnKEBench (Section 5.1). The results, presented in the Figure 2, are crucial. According to **Hypothesis 1**, successful editing of the early MLP layers should enhance model performance. Contrary to this expectation, our results indicate that the success rate of editing structured knowledge and the Bert-Score of editing unstructured knowledge are largely unaffected by the number of edited layers. Therefore, the conclusion is that knowledge is not confined to specific layers; rather, it is **distributed non-locally** throughout the network. Our perspective aligns with findings from other studies (Hase et al., 2023). Combining the non-local characteristics of knowledge and the conclusions that also exist in the attention layer (Li et al., 2024; Wei et al., 2024a), we advocate for the adoption of non-local block key-value storage, endowed with enhanced representational abilities, over local layer key-value storage. This shift is essential for effectively encapsulating the intricacies of unstructured knowledge.

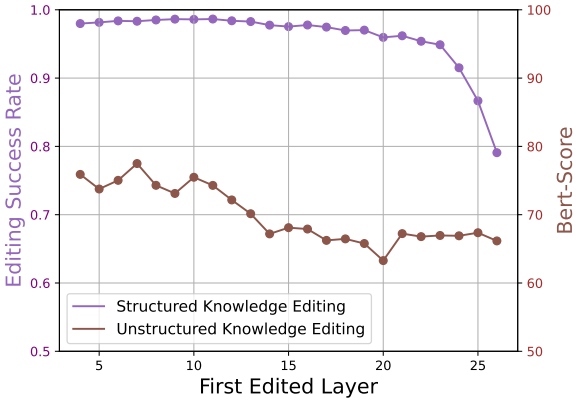

Figure 2: Impact of different edited layers on the performance of MEMIT in editing structured and unstructured knowledge. The x-axis indicates the starting layer number for editing, and the number of edited layers is 5. Bert-Score is a metric in UnKEBench; a higher value indicates better model performance.

3.2 Term-driven Editing Lacks Robustness

Hypothesis 2: *Knowledge editing is driven by specific terms in sentences.* MEMIT and ROME both increase editing success rate by locating the last token in the subject. As shown in the Table 1, omitting this step causes their performance to drop significantly on KEBench (Wei et al., 2024c). For structured knowledge, the subject can be easily identified; however, for unstructured knowledge, accurately determining the subject is challenging due to its distributed semantics. In addition, it is inconvenient if positioning operations are required for each edit. Therefore, we argue that this step should be omitted in unstructured knowledge editing and editing can be performed directly at the sentence level.

Table 1: Performance comparison on KEBench: Impact of locating the subject. Ori-Acc and Para-Acc represent the accuracy for the original question and the paraphrased question, respectively. None Subject indicates the last token to the question.

Method	Subject		None Subject	
	Ori-Acc	Para-Acc	Ori-Acc	Para-Acc
ROME	77.90	68.40	44.10	23.60
MEMIT	74.80	64.30	37.60	27.10

4 UnKE: Unstructured Knowledge Editing Method

Building on the above motivations, our study proposes two primary solutions: (1) Employing non-local block key-value storage instead of local layer key-value storage to capture information-rich unstructured knowledge effectively, and (2) opting for causal-driven optimization over term-driven optimization for editing purposes to eliminate positional term operations and mitigate the loss of contextual information.

4.1 NON-LOCAL BLOCK KEY-VALUE STORAGE

Many studies (Wang et al., 2024; Yao et al., 2024) suggest that the initial layers in LLMs store foundational knowledge, processing inputs to extract general information. In the deeper layers, target information is already present in the residual stream. The main role of these deep layers is to refine and enhance the model's confidence in its predictions, increasing the likelihood of accurate outputs—an effect known as the "early decoding" phenomenon (Yao et al., 2024).

Inspired by this, we argue that the shallow layers of LLMs encode the key vector of knowledge, which aggregates entity information and contextual concepts related to the problem. The deep layers decode this key vector into a value vector, responsible for integrating target information into the residual stream. This transformer block-level key-value pair representation is more effective than traditional MLP layer-level key-value pairs. It enables nonlinear mapping and incorporates information from the attention layer, making it more suitable for representing unstructured knowledge. Specifically, we consider the L-th layer of the LLM as a boundary, dividing it into two distinct components: a key generator and a value generator. These components produce key vectors and value vectors, respectively. Experience indicates that a value of 7 is more suitable for L. For detailed experimental information regarding the value of L, please refer to Appendix F. Next, we formalize the process of generating Transformer block-level key-value pairs.

Let $f_\theta = f^1_{\theta_1} \circ \cdots \circ f^l_{\theta_l} \circ \cdots \circ f^N_{\theta_N}$ denote an autoregressive LLM with parameters θ, which can be regarded as an N-layer Transformer decoder, and $\circ$ stands for cascade symbol. For the l-th layer, we denote it as $f^l_{\theta_l}$, where θ_l represents the parameters of this layer. Then the key generator is represented as $f^{l\leq L}_{\theta_k} = f^1_{\theta_1} \circ \cdots \circ f^L_{\theta_L}$, and the value generator $f^{L<l\leq N}_{\theta_v} = f^{L+1}_{\theta_{L+1}} \circ \cdots \circ f^N_{\theta_N}$, where θ_k and θ_v are parameters of the key generator and the value generator respectively.

For a given question $q = [q_1, q_2, \ldots, q_n]$, the key vector k should be expressed as

$$k = f^{l\leq L}_{\theta_k}([q_1, q_2, \ldots, q_n]), \tag{1}$$

where q_i represents the i-th token of the question, and n represents the number of question tokens. Function $f^{l\leq L}_{\theta_k}(\cdot)$ represents the last token representation of $f^{l\leq L}_{\theta_k}$ forward propagation. We use $h^l_q = [h^l_{q,1}, h^l_{q,2}, \ldots, h^l_{q,n}]$ to represent the hidden state of q in the l-th layer. Then we can also conclude that $k = h^L_{q,n}$. It is worth noting that for term-driven methods, the term position t they locate is usually less than n. Due to the causal attention mechanism, the key vectors they generate do not store information about the position after the term, resulting in information loss. Please see the next section for details. The value vector v is

$$v = f^{L<l\leq N}_{\theta_v}([h^L_{q,1}, h^L_{q,2}, \ldots, k]). \tag{2}$$

At this time, the basic information contained in key vector k has been decoded by the value generator into target information, which is then written into the residual stream to become value vector v. Our goal is to modify them to obtain the editing target $a = [a_1, a_2, \ldots, a_m]$, where m represents the number of target tokens. The process is denoted as $(k \mapsto k^*, v \mapsto v^*)$, where k^* and v^* represent the key vector and value vector we expect to get. In the next section, we elaborate on cause-driven optimization.

4.2 CAUSE-DRIVEN OPTIMIZATION

The core idea of cause-driven optimization is uncomplicated and effective, that is, the last token of the autoregressive LLMs aggregates the information of all previous tokens, so editing should be performed based on this as an anchor. When editing the last token, the key vectors of other previous tokens should unchanged.

First, we calculate the key vector k^* and value vector v^* to be modified based on the editing target a. Inspired by previous work (Meng et al., 2023), we find $k^* = h^l_{q,n} + \delta_n$ directly by optimizing the residual vector δ_n

using gradient descent. We formalize this process as

$$k^* = h_n^l + \underset{\delta_n}{\operatorname{argmin}} -\log \mathbb{P}_{f_\theta(h_{q,n}^L \mapsto h_{q,n}^L + \delta_n)}(a|q), \tag{3}$$

where $f_\theta(h_{q,n}^L \mapsto h_{q,n}^L + \delta_n)$ means that we replace the hidden state $h_{q,n}^L$ (also be expressed as original key vector k) with k^*. Then we can calculate v^* using Eq. 2. If we freeze the parameters of the value generator $f_{\theta_v}^{L<l\leq N}$, optimizing Eq. 3 to a sufficiently small value implies that if we obtain $k^* = f_{\theta_k}^{l\leq L}(q_1, q_2, \ldots, q_n)$, then we can decode the target a.

We now introduce the process of optimizing the key generator $f_{\theta_k}^{l\leq L}$ to obtain the key vector k^*. $f_{\theta_k}^{l\leq L}$ store a large number of key vectors $K_0 = [k_1 \mid k_2 \mid \ldots \mid k_E]$ during the pre-training process, which can be activated by specific inputs $D_0 = [d_1 \mid d_2 \mid \ldots \mid d_E]$ to generate corresponding value vectors $V_0 = [v_1 \mid v_2 \mid \ldots \mid v_E]$. We can express it as

$$f_{\theta_k}^{l\leq L} \triangleq \underset{\hat{\theta}}{\operatorname{argmin}} \sum_{i=1}^{E} \parallel f_{\hat{\theta}}^{l\leq L}(d_i) - k_i \parallel^2, \tag{4}$$

where E represents the number of knowledge key-value pairs introduced during pre-training, which can be regarded as $+\infty$. Therefore during the optimization process we should minimize the parameter changes of $f_{\theta_k}^{l\leq L}$ and produce a new key generator $f_{\theta_k^{'}}^{l\leq L}$ that can generate the new key k^*. We formalize this process as

$$f_{\theta_k^{'}}^{l\leq L} \triangleq \underset{\hat{\theta}}{\operatorname{argmin}}(\sum_{i=1}^{E} \parallel f_{\hat{\theta}}^{l\leq L}(d_i) - k_i \parallel^2 + \parallel f_{\hat{\theta}}^{l\leq L}(q) - k^* \parallel^2), \tag{5}$$

where $\theta_k^{'}$ represents the updated parameters. This approach minimizes the impact of adding new key-value pairs on the original key-value pairs. In particular, we are able to edit a batch of u unstructured knowledge at one time, which we denote by $K_1 = [k_1^* \mid k_2^* \mid \ldots \mid k_u^*]$. Eq. 5 can be changed to

$$f_{\theta_k^{'}}^{l\leq L} \triangleq \underset{\hat{\theta}}{\operatorname{argmin}}(\sum_{i=1}^{E} \parallel f_{\hat{\theta}}^{l\leq L}(d_i) - k_i \parallel^2 + \sum_{j=1}^{u} \parallel f_{\hat{\theta}}^{l\leq L}(q_j) - k_j^* \parallel^2). \tag{6}$$

To avoid the addition of new keys affecting the generation of original keys, we only optimize the last layer of the key encoder $f_{\theta_L}^L$. In order to optimize Eq. 6, we randomly select a number C of pre-training samples to simulate the knowledge $f_{\theta_L}^L$ learned during pre-training. Assuming that i-th pre-training sample can be represented as $t^i = [t_1^i, t_2^i, \ldots, t_P^i]$, where P represents the number of i-th pre-training sample tokens. Before performing optimization, we first calculate the key vector $k_{t,p}^i = f_{\theta_L}^L(h_{t,1}^{i,L-1}, h_{t,2}^{i,L-1}, \ldots, h_{t,p}^{i,L-1})$ corresponding to the p-th token in i-th pre-training sample, where $h_{t,p}^{i,L-1}$ represents the vector of the p-th token of the i-th pre-training sample in the l-th layer. During the editing process, we need to ensure that the key vector corresponding to each token of the pre-training sample remains unchanged, so as to retain the knowledge acquired by the model during pre-training to the greatest extent and prevent catastrophic forgetting.

Finally, consider that when optimizing the key generator $f_{\theta_k}^{l\leq L}$ to generate the k^*, changes in parameters of $f_{\theta_k}^{l\leq L}$ may cause the representation of the context $h_c^L = [h_{q,1}^L, h_{q,2}^L, \ldots, h_{q,n-1}^L]$ to change after passing through the $f_{\theta_k}^{l\leq L}$, thereby reducing the model performance. Therefore, we impose constraints to ensure that the context representation h_c^L is not changed during the editing process, which leads to the final optimization

goal, which is

$$f^L_{\theta'_L} = \underset{\hat{\theta}_L}{\operatorname{argmin}}(\underbrace{\sum_{i=1}^{C}\sum_{p=1}^{P} \parallel f^L_{\hat{\theta}_L}(h^{i,L-1}_{t,\leq p}) - k^i_{t,p} \parallel^2}_{\text{Key Preservation Loss}} + \underbrace{\sum_{i=1}^{u}\sum_{j=1}^{n-1} \parallel f^L_{\hat{\theta}_L}(h^{i,L-1}_{q,\leq j}) - k^i_{q,j} \parallel^2}_{\text{Key Causal Loss}} + \underbrace{\sum_{i=1}^{u} \parallel f^L_{\hat{\theta}_L}(h^{i,L-1}_{q,\leq n}) - k^{*,i}_{q} \parallel^2}_{\text{Key Learning Loss}}), \tag{7}$$

where $h^{i,L-1}_{t,\leq p}$ represents tokens less than or equal to p in the i-th pre-train sample, and $h^{i,L-1}_{q,\leq j}$ represents tokens less than or equal to i in the j-th question to be edited. Key Preservation Loss ensures that the key generator retains the keys stored during pre-training, enabling the preservation of original knowledge. Key Causal Loss ensures that the contextual information is not biased when the model learns new key vectors. Additionally, Key Learning Loss facilitates the key generator in acquiring new keys, and achieving the desired editing target.

5 EXPERIMENTS

Due to the lack of datasets for compiling unstructured knowledge, we developed UnKEBench (Section 5.1). Subsequently, we assess the model's efficacy in unstructured knowledge editing (Section 5.2) and structured knowledge editing (Section 5.3). Finally, we conduct ablation experiments to ascertain the impacts of different designs (Section 5.4).

5.1 CONSTRUCTION OF UNKEBENCH

The unstructured texts are notably lengthy and contain knowledge that extends beyond simple knowledge triples or linear fact chains. To effectively manage this complexity, we divide our construction approach into four distinct phases.

1. We employ meticulously crafted instructions to guide ChatGPT in formulating the most appropriate question Q for each text A, thus creating an unstructured knowledge pair (Q, A).
2. To refine our evaluation mechanism, we use detailed instructions to prompt ChatGPT to generate a paraphrased version of each original question, denoted as Q_p, for every original question Q.
3. We leverage knowledge decomposition strategies and engage ChatGPT to produce multiple sub-question and sub-answer pairs (Q^i_s, A^i_s) for each (Q, A).
4. Finally, we randomly sample five questions from MMLU (Hendrycks et al., 2021) for each example to evaluate the general ability of the edited model.

Details and examples of constructing UnKEBench are provided in the Appendix B. Due to space limitations, we introduce the differences between existing knowledge editing benchmarks and UnKEBench in Appendix A. Our evaluation framework for unstructured knowledge editing mirrors the complexity of the task by integrating four critical dimensions: word-level overlap, semantic similarity, factual correctness and general ability.

- **Lexical Similarity** metrics, including BLEU Papineni et al. (2002) and various ROUGE scores Lin (2004) (ROUGE-1, ROUGE-2, and ROUGE-L), provide insight into the lexical and n-gram alignment between the model-generated text and the target answers, based on both the original and paraphrased questions. These metrics are fundamental in assessing the surface-level accuracy of the edited content.

Table 2: Unstructured knowledge editing performance with different methods. During the editing process, we set the batch size to 1. With each editing instance, the parameters of the modified model are rebuilt. The decoding process employs a temperature of 0.001. To ensure fair comparison, the 7-th layer of parameters of the model is specifically targeted for editing across FT-L, ROME, and UnKE. The figures to the left and right of the '/' symbol denote the evaluation outcomes for output of the model in response to the original and paraphrased questions, respectively. **FC.** stands for Factual Correctness.

Method	Semantic Similarity	Lexical Similarity				FC.	General Ability
	Bert-Score	BLEU	Rouge-1	Rouge-2	Rouge-L	FactScore	MMLU
Based on LLaMA2-7B-Chat							29.78
FT-A	2.56 / 2.58	1.01 / 1.02	0.92 / 0.92	0.01 / 0.01	0.92 / 0.92	8.74	$29.57_{0.21\downarrow}$
FT-L	11.63 / 10.16	6.14 / 5.52	7.55 / 6.78	1.37 / 1.28	7.26 / 6.53	15.69	$29.27_{0.51\downarrow}$
ROME	76.52 / 74.29	38.71 / 33.42	47.31 / 41.64	28.89 / 20.93	45.05 / 39.06	24.44	$29.78_{0.00\downarrow}$
MEMIT	75.90 / 74.46	35.79 / 33.19	43.55 / 41.39	23.11 / 19.89	40.96 / 38.81	26.39	$29.77_{0.01\downarrow}$
MEND	69.99 / 64.71	24.10 / 29.23	45.36 / 45.06	31.75 / 29.33	44.05 / 43.77	24.17	$28.50_{1.28\downarrow}$
UnKE	**99.61 / 93.09**	**98.63 / 76.85**	**98.77 / 78.62**	**98.33 / 70.66**	**98.73 / 77.70**	**42.49**	$29.68_{0.10\downarrow}$
Based on Qwen1.5-7B-Chat							32.43
MEMIT	74.72 / 76.82	48.89 / 48.71	49.50 / 48.18	34.59 / 31.50	47.55 / 46.04	17.81	$31.69_{0.74\downarrow}$
UnKE	**96.51 / 90.40**	**92.85 / 75.66**	**91.74 / 72.68**	**88.19 / 60.59**	**91.40 / 70.44**	**40.08**	$32.03_{0.40\downarrow}$

- **Semantic Similarity**. As word-level overlap metrics alone are insufficient for capturing the nuanced understanding a model must exhibit. To bridge this gap, we evaluate semantic similarity by leveraging an embedding encoder (specifically, the all-MiniLM-L6-v2 model[1]) to quantify the depth of comprehension of the model of the text, ensuring a balanced evaluation that transcends mere lexical matching.
- **Factual Correctness**. In order to evaluate in a more fine-grained manner whether the edited model has indeed understood the unstructured knowledge, we use FactScore Min et al. (2023) to evaluate the accuracy of the edited model in processing sub-questions and their corresponding answers. This metric is similar to the multi-hop accuracy in some structured knowledge editing benchmarks.
- **General Ability**. We follow the code in MMLU (Hendrycks et al., 2021) to evaluate each MMLU sample. We then average the scores of five MMLU samples for each unstructured sample, and finally average these scores across all unstructured samples.

In summary, these four aspects form a robust framework for evaluating unstructured knowledge edits, ensuring both the fidelity and the flexibility of the generated content are thoroughly examined.

5.2 Experiments on Unstructured Knowledge Editing

We conduct a comprehensive evaluation of various baseline methods (Appendix C) and our newly proposed UnKE method on the UnKEBench benchmark, including both automatic and human evaluation.

Automatic Evaluation. The specific results are presented in Table 2. Traditional fine-tuning methods, including FT-L and FT-A, have long exhibited significant limitations when tasked with structured knowledge editing. As anticipated, their performance on UnKEBench is also underwhelming, with all evaluation metrics falling short of those achieved by dedicated knowledge editing approaches. Methods employing a Locate-Then-Edit paradigm, such as ROME and MEMIT, despite previously demonstrating satisfactory editing success rates on certain structured benchmarks, underperform on the UnKEBench dataset, particularly in terms of lexical and semantic similarity when compared to UnKE. UnKE demonstrates exceptional performance, surpassing other models in lexical and semantic similarity metrics, which confirms its ability to accurately

[1]https://huggingface.co/sentence-transformers/all-MiniLM-L6-v2.

Table 3: Performance on human evaluation (a) and structured knowledge editing performance on KEBench (b). Ori-Acc and Para-Acc represent the accuracy for the original question and the paraphrased question, respectively. Src-Acc and Tgt-Acc represent the irrelevant knowledge accuracy of subject and object in the triplet, respectively.

Method	Corr.	Simi.	Cohe.
FT-A	1.06	1.47	1.47
FT-L	1.17	1.00	1.31
ROME	3.39	3.59	3.64
MEMIT	3.25	3.70	3.72
UnKE	**4.78**	**4.72**	**4.70**

(a) Human Evaluations

Method	Ori-Acc	Para-Acc	Src-Acc	Tgt-Acc
FT-A	6.30	6.60	8.60	9.30
FT-L	14.70	12.10	5.40	5.70
ROME	77.90	68.40	96.80	**76.80**
MEMIT	74.80	64.30	**97.60**	76.40
UnKE	**94.30**	**86.40**	90.40	68.80

(b) Structured Knowledge Editing

capture and reproduce the intended editing objectives—a feat that other models do not achieve. For more examples of generated cases, please refer to the Appendix E. Regarding the detailed evaluation metric of FactScore, UnKE achieves a score of 42.49, outperforming other strong baseline models, yet highlighting that there is still room for further improvement. Furthermore, UnKE has little impact on the model's general capabilities.

We also extend our unstructured knowledge editing experiments to utilize Qwen1.5-7b-Chat as base model and compare against MEMIT. The results indicate that our approach outperforms MEMIT across multiple evaluation dimensions significantly. These experiments, conducted on models with varying architectures, demonstrate the robust transferability of our proposed UnKE method.

Human Evaluation. Considering the complexity and challenges involved in automatically evaluating unstructured knowledge editing, we conduct additional manual evaluation experiments to ensure the reliability of the evaluation metrics and actual scores in UnKEBench. Due to the high cost of human evaluation, we randomly select 36 samples from a pool of 1000 samples generated by each method. We employ three annotators, experienced in knowledge editing tasks but not involved in this project's training, to conduct a manual evaluation. They were instructed to assess the edited generated text across three dimensions: semantic correctness, similarity, and coherence on a scale of 1-5, with 1 denoting "very low" and 5 representing "very high". The scores are then averaged to derive the final human evaluation results. The evaluation results, presented in Table 3a, reflect the collective assessments by the hired professionals. The inter-annotator agreement is 0.57 in Fleiss' κ, which means a moderate agreement.

The experimental results provide strong evidence of the high consistency between the automatic evaluations and human evaluations. UnKE stands out as the leader across all three dimensions. In contrast, the other baseline models frequently exhibit subpar performance in terms of semantic correctness, highlighting their limited ability to effectively edit unstructured knowledge. To further quantify the correlation between the automatic evaluation metrics and the human evaluation metrics, we calculated the Pearson correlation coefficient. Refer to Appendix G for details.

5.3 Experiments on Structured Knowledge Editing

To validate the capability of UnKE in editing knowledge triples, we conduct experiments on KEBench (Wei et al., 2024c), a benchmark that evaluates whether the model accurately produces the desired target answer after editing. The results presented in Table 3b demonstrate that UnKE surpasses strong baseline models in terms of Ori-Acc and Para-Acc metrics, exhibiting improvements of 16.4 points and 18 points, respectively. When comparing the results with UnKEBench, the improvement of UnKE over the strong baseline may not be as pronounced. However, this outcome is anticipated since UnKE primarily targets complex and lengthy unstructured knowledge editing tasks, making it less conspicuous in simpler structured knowledge editing

Table 4: Ablation experiments. "Pres. Loss" and "Caus. Loss" denote Key Preservation Loss and Key Causal Loss, correspondingly. "w/ MLP" and "w/ ATTN" respectively specify that during optimization, only the parameters of the MLP and Attention modules in the transformer block are utilized.

Method	Semantic Similarity	Lexical Similarity				FC.	General Ability
	Bert-Score	BLEU	Rouge-1	Rouge-2	Rouge-L	Fact-Score	MMLU
UnKE	99.61 / 93.09	98.63 / 76.85	98.77 / 78.62	98.33 / 70.66	98.73 / 77.70	42.49	$29.68_{0.10\downarrow}$
Modules							
w/ MLP	95.43 / 87.87	92.34 / 71.32	94.78 / 73.39	92.91 / 68.51	93.23 / 72.65	37.98	$29.77_{0.01\downarrow}$
w/ ATTN	92.66 / 81.62	90.58 / 63.46	91.16 / 70.03	89.73 / 68.30	90.21 / 71.15	31.01	$29.71_{0.07\downarrow}$
Loss Function							
w/o Pres. Loss	99.00 / 94.94	96.99 / 82.39	97.35 / 83.81	96.29 / 77.19	97.21 / 83.20	38.74	$29.44_{0.34\downarrow}$
w/o Caus. Loss	21.19 / 26.27	26.69 / 31.79	10.29 / 13.46	24.93 / 29.68	46.50 / 58.91	16.77	$29.52_{0.26\downarrow}$
w/o Pres. & Caus. Loss	9.32 / 9.79	11.96 / 12.94	2.08 / 2.31	11.12 / 12.10	14.98 / 18.19	6.27	$27.62_{2.16\downarrow}$

tasks. In general, experimental results have demonstrated that UnKE is not only effective in unstructured knowledge editing but can also be applied to structured knowledge.

5.4 Ablation Experiments

To validate the efficacy of our proposed approach, we conduct ablation experiments on non-local block key-value storage and cause-driven optimization. Table 4 illustrates that for non-local block key-value storage, we selectively optimized either the MLP or Attention module when generating the key vector to assess the model's performance under these conditions. The outcomes indicate that optimizing solely the MLP or Attention module leads to a partial performance decrease, reinforcing the premise of non-local knowledge storage outlined in Section 3.1. Specifically, optimizing just the MLP is insufficient for achieving optimal results; hence, a combination of non-local block key-value storage with both MLP and Attention modules is imperative for effectively representing information-rich unstructured knowledge.

In the context of causal-driven optimization, we conducted an ablation analysis on the loss function, specifically omitting the Key Preservation Loss and Key Causal Loss individually. The findings reveal that excluding the Key Preservation Loss leads to a degradation in model performance, particularly affecting the MMLU metric. Conversely, eliminating the Key Causal Loss results in a significant decline in editing performance due to the absence of contextual information. However, given the presence of Key Preservation Loss, the general ability of the model remains relatively stable. Notably, when both losses are discarded, the model performance reaches its lowest point. In addition, we also verify that UnKE has robust batch editing and sequential editing capabilities (Section D).

6 Conclusions

We address the limitations of existing knowledge editing benchmarks, which primarily focus on structured knowledge triples, by introducing UnKEBench, the first benchmark for unstructured knowledge editing. To successfully edit unstructured knowledge, we propose UnKE, an unstructured knowledge editing method, which incorporates non-local block key-value storage and cause-driven optimization, enabling it to effectively represent and edit unstructured knowledge with ease. Experimental results on UnKEBench demonstrate the superior performance of UnKE, significantly surpassing powerful baseline models on various evaluation metrics. Robustness analysis experiments confirm that UnKE possesses the ability to perform both batch and sequential editing. Additionally, UnKE also compares favorably with other strong baseline models on structured knowledge editing benchmarks.

REFERENCES

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

Gabriele Bavota. Mining unstructured data in software repositories: Current and future trends. In *Leaders of Tomorrow Symposium: Future of Software Engineering, FOSE@SANER 2016, Osaka, Japan, March 14, 2016*, pp. 1–12. IEEE Computer Society, 2016. doi: 10.1109/SANER.2016.47. URL https://doi.org/10.1109/SANER.2016.47.

Canyu Chen and Kai Shu. Combating misinformation in the age of llms: Opportunities and challenges. *CoRR*, abs/2311.05656, 2023. doi: 10.48550/ARXIV.2311.05656. URL https://doi.org/10.48550/arXiv.2311.05656.

Canyu Chen and Kai Shu. Can llm-generated misinformation be detected? In *The Twelfth International Conference on Learning Representations, ICLR 2024, Vienna, Austria, May 7-11, 2024*. OpenReview.net, 2024. URL https://openreview.net/forum?id=ccxD4mtkTU.

Siyuan Cheng, Bozhong Tian, Qingbin Liu, Xi Chen, Yongheng Wang, Huajun Chen, and Ningyu Zhang. Can we edit multimodal large language models? *arXiv preprint arXiv:2310.08475*, 2023.

Damai Dai, Li Dong, Yaru Hao, Zhifang Sui, Baobao Chang, and Furu Wei. Knowledge neurons in pretrained transformers. In Smaranda Muresan, Preslav Nakov, and Aline Villavicencio (eds.), *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), ACL 2022, Dublin, Ireland, May 22-27, 2022*, pp. 8493–8502. Association for Computational Linguistics, 2022. doi: 10.18653/V1/2022.ACL-LONG.581. URL https://doi.org/10.18653/v1/2022.acl-long.581.

Tom Hartvigsen, Swami Sankaranarayanan, Hamid Palangi, Yoon Kim, and Marzyeh Ghassemi. Aging with GRACE: lifelong model editing with discrete key-value adaptors. In Alice Oh, Tristan Naumann, Amir Globerson, Kate Saenko, Moritz Hardt, and Sergey Levine (eds.), *Advances in Neural Information Processing Systems 36: Annual Conference on Neural Information Processing Systems 2023, NeurIPS 2023, New Orleans, LA, USA, December 10 - 16, 2023*, 2023. URL http://papers.nips.cc/paper_files/paper/2023/hash/95b6e2ff961580e03c0a662a63a71812-Abstract-Conference.html.

Peter Hase, Mohit Bansal, Been Kim, and Asma Ghandeharioun. Does localization inform editing? surprising differences in causality-based localization vs. knowledge editing in language models. In Alice Oh, Tristan Naumann, Amir Globerson, Kate Saenko, Moritz Hardt, and Sergey Levine (eds.), *Advances in Neural Information Processing Systems 36: Annual Conference on Neural Information Processing Systems 2023, NeurIPS 2023, New Orleans, LA, USA, December 10 - 16, 2023*, 2023. URL http://papers.nips.cc/paper_files/paper/2023/hash/3927bbdcf0e8d1fa8aa23c26f358a281-Abstract-Conference.html.

Dan Hendrycks, Collin Burns, Steven Basart, Andy Zou, Mantas Mazeika, Dawn Song, and Jacob Steinhardt. Measuring massive multitask language understanding. In *9th International Conference on Learning Representations, ICLR 2021, Virtual Event, Austria, May 3-7, 2021*. OpenReview.net, 2021. URL https://openreview.net/forum?id=d7KBjmI3GmQ.

Zeyu Huang, Yikang Shen, Xiaofeng Zhang, Jie Zhou, Wenge Rong, and Zhang Xiong. Transformer-patcher: One mistake worth one neuron. In *The Eleventh International Conference on Learning Representations,*

ICLR 2023, Kigali, Rwanda, May 1-5, 2023. OpenReview.net, 2023. URL https://openreview.net/pdf?id=4oYUGeGBPm.

Omer Levy, Minjoon Seo, Eunsol Choi, and Luke Zettlemoyer. Zero-shot relation extraction via reading comprehension. In Roger Levy and Lucia Specia (eds.), *Proceedings of the 21st Conference on Computational Natural Language Learning (CoNLL 2017), Vancouver, Canada, August 3-4, 2017*, pp. 333–342. Association for Computational Linguistics, 2017. doi: 10.18653/V1/K17-1034. URL https://doi.org/10.18653/v1/K17-1034.

Xiaopeng Li, Shasha Li, Shezheng Song, Jing Yang, Jun Ma, and Jie Yu. PMET: precise model editing in a transformer. In Michael J. Wooldridge, Jennifer G. Dy, and Sriraam Natarajan (eds.), *Thirty-Eighth AAAI Conference on Artificial Intelligence, AAAI 2024, Thirty-Sixth Conference on Innovative Applications of Artificial Intelligence, IAAI 2024, Fourteenth Symposium on Educational Advances in Artificial Intelligence, EAAI 2014, February 20-27, 2024, Vancouver, Canada*, pp. 18564–18572. AAAI Press, 2024. doi: 10.1609/AAAI.V38I17.29818. URL https://doi.org/10.1609/aaai.v38i17.29818.

Chin-Yew Lin. Rouge: A package for automatic evaluation of summaries. In *Text summarization branches out*, pp. 74–81, 2004.

Jiateng Liu, Pengfei Yu, Yuji Zhang, Sha Li, Zixuan Zhang, and Heng Ji. EVEDIT: event-based knowledge editing with deductive editing boundaries. *CoRR*, abs/2402.11324, 2024. doi: 10.48550/ARXIV.2402.11324. URL https://doi.org/10.48550/arXiv.2402.11324.

Shengyu Mao, Ningyu Zhang, Xiaohan Wang, Mengru Wang, Yunzhi Yao, Yong Jiang, Pengjun Xie, Fei Huang, and Huajun Chen. Editing personality for llms. *arXiv preprint arXiv:2310.02168*, 2023.

Kevin Meng, David Bau, Alex Andonian, and Yonatan Belinkov. Locating and editing factual associations in GPT. In Sanmi Koyejo, S. Mohamed, A. Agarwal, Danielle Belgrave, K. Cho, and A. Oh (eds.), *Advances in Neural Information Processing Systems 35: Annual Conference on Neural Information Processing Systems 2022, NeurIPS 2022, New Orleans, LA, USA, November 28 - December 9, 2022*, 2022. URL http://papers.nips.cc/paper_files/paper/2022/hash/6f1d43d5a82a37e89b0665b33bf3a182-Abstract-Conference.html.

Kevin Meng, Arnab Sen Sharma, Alex J. Andonian, Yonatan Belinkov, and David Bau. Mass-editing memory in a transformer. In *The Eleventh International Conference on Learning Representations, ICLR 2023, Kigali, Rwanda, May 1-5, 2023*. OpenReview.net, 2023. URL https://openreview.net/pdf?id=MkbcAHIYgyS.

Sewon Min, Kalpesh Krishna, Xinxi Lyu, Mike Lewis, Wen-tau Yih, Pang Wei Koh, Mohit Iyyer, Luke Zettlemoyer, and Hannaneh Hajishirzi. Factscore: Fine-grained atomic evaluation of factual precision in long form text generation. In Houda Bouamor, Juan Pino, and Kalika Bali (eds.), *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing, EMNLP 2023, Singapore, December 6-10, 2023*, pp. 12076–12100. Association for Computational Linguistics, 2023. doi: 10.18653/V1/2023.EMNLP-MAIN.741. URL https://doi.org/10.18653/v1/2023.emnlp-main.741.

Eric Mitchell, Charles Lin, Antoine Bosselut, Chelsea Finn, and Christopher D. Manning. Fast model editing at scale. In *The Tenth International Conference on Learning Representations, ICLR 2022, Virtual Event, April 25-29, 2022*. OpenReview.net, 2022a. URL https://openreview.net/forum?id=0DcZxeWfOPt.

Eric Mitchell, Charles Lin, Antoine Bosselut, Christopher D. Manning, and Chelsea Finn. Memory-based model editing at scale. In Kamalika Chaudhuri, Stefanie Jegelka, Le Song, Csaba Szepesvári, Gang Niu, and Sivan Sabato (eds.), *International Conference on Machine Learning, ICML 2022, 17-23 July 2022, Baltimore, Maryland, USA*, volume 162 of *Proceedings of Machine Learning Research*, pp. 15817–15831. PMLR, 2022b. URL https://proceedings.mlr.press/v162/mitchell22a.html.

Yasumasa Onoe, Michael J. Q. Zhang, Shankar Padmanabhan, Greg Durrett, and Eunsol Choi. Can lms learn new entities from descriptions? challenges in propagating injected knowledge. In Anna Rogers, Jordan L. Boyd-Graber, and Naoaki Okazaki (eds.), *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), ACL 2023, Toronto, Canada, July 9-14, 2023*, pp. 5469–5485. Association for Computational Linguistics, 2023. doi: 10.18653/V1/2023.ACL-LONG.300. URL https://doi.org/10.18653/v1/2023.acl-long.300.

Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. Bleu: a method for automatic evaluation of machine translation. In *Proceedings of the 40th Annual Meeting of the Association for Computational Linguistics, July 6-12, 2002, Philadelphia, PA, USA*, pp. 311–318. ACL, 2002. doi: 10.3115/1073083.1073135. URL https://aclanthology.org/P02-1040/.

Hugo Touvron, Louis Martin, Kevin Stone, Peter Albert, Amjad Almahairi, Yasmine Babaei, Nikolay Bashlykov, Soumya Batra, Prajjwal Bhargava, Shruti Bhosale, Dan Bikel, Lukas Blecher, Cristian Canton-Ferrer, Moya Chen, Guillem Cucurull, David Esiobu, Jude Fernandes, Jeremy Fu, Wenyin Fu, Brian Fuller, Cynthia Gao, Vedanuj Goswami, Naman Goyal, Anthony Hartshorn, Saghar Hosseini, Rui Hou, Hakan Inan, Marcin Kardas, Viktor Kerkez, Madian Khabsa, Isabel Kloumann, Artem Korenev, Punit Singh Koura, Marie-Anne Lachaux, Thibaut Lavril, Jenya Lee, Diana Liskovich, Yinghai Lu, Yuning Mao, Xavier Martinet, Todor Mihaylov, Pushkar Mishra, Igor Molybog, Yixin Nie, Andrew Poulton, Jeremy Reizenstein, Rashi Rungta, Kalyan Saladi, Alan Schelten, Ruan Silva, Eric Michael Smith, Ranjan Subramanian, Xiaoqing Ellen Tan, Binh Tang, Ross Taylor, Adina Williams, Jian Xiang Kuan, Puxin Xu, Zheng Yan, Iliyan Zarov, Yuchen Zhang, Angela Fan, Melanie Kambadur, Sharan Narang, Aurélien Rodriguez, Robert Stojnic, Sergey Edunov, and Thomas Scialom. Llama 2: Open foundation and fine-tuned chat models. *CoRR*, abs/2307.09288, 2023. doi: 10.48550/ARXIV.2307.09288. URL https://doi.org/10.48550/arXiv.2307.09288.

Jiaan Wang, Yunlong Liang, Zengkui Sun, Yuxuan Cao, and Jiarong Xu. Cross-lingual knowledge editing in large language models. *CoRR*, abs/2309.08952, 2023a. doi: 10.48550/ARXIV.2309.08952. URL https://doi.org/10.48550/arXiv.2309.08952.

Mengru Wang, Yunzhi Yao, Ziwen Xu, Shuofei Qiao, Shumin Deng, Peng Wang, Xiang Chen, Jia-Chen Gu, Yong Jiang, Pengjun Xie, Fei Huang, Huajun Chen, and Ningyu Zhang. Knowledge mechanisms in large language models: A survey and perspective. *CoRR*, abs/2407.15017, 2024. doi: 10.48550/ARXIV.2407.15017. URL https://doi.org/10.48550/arXiv.2407.15017.

Peng Wang, Ningyu Zhang, Xin Xie, Yunzhi Yao, Bozhong Tian, Mengru Wang, Zekun Xi, Siyuan Cheng, Kangwei Liu, Guozhou Zheng, et al. Easyedit: An easy-to-use knowledge editing framework for large language models. *arXiv preprint arXiv:2308.07269*, 2023b.

Weixuan Wang, Barry Haddow, and Alexandra Birch. Retrieval-augmented multilingual knowledge editing. *CoRR*, abs/2312.13040, 2023c. doi: 10.48550/ARXIV.2312.13040. URL https://doi.org/10.48550/arXiv.2312.13040.

Yifan Wei, Xiaoyan Yu, Yixuan Weng, Huanhuan Ma, Yuanzhe Zhang, Jun Zhao, and Kang Liu. Does knowledge localization hold true? surprising differences between entity and relation perspectives in language models. *arXiv preprint arXiv:2409.00617*, 2024a.

Zihao Wei, Jingcheng Deng, Liang Pang, Hanxing Ding, Huawei Shen, and Xueqi Cheng. Mlake: Multilingual knowledge editing benchmark for large language models, 2024b.

Zihao Wei, Liang Pang, Hanxing Ding, Jingcheng Deng, Huawei Shen, and Xueqi Cheng. Stable knowledge editing in large language models. *CoRR*, abs/2402.13048, 2024c. doi: 10.48550/ARXIV.2402.13048. URL https://doi.org/10.48550/arXiv.2402.13048.

Suhang Wu, Minlong Peng, Yue Chen, Jinsong Su, and Mingming Sun. Eva-kellm: A new benchmark for evaluating knowledge editing of llms. *CoRR*, abs/2308.09954, 2023. doi: 10.48550/ARXIV.2308.09954. URL https://doi.org/10.48550/arXiv.2308.09954.

Jian Xie, Kai Zhang, Jiangjie Chen, Renze Lou, and Yu Su. Adaptive chameleon or stubborn sloth: Revealing the behavior of large language models in knowledge conflicts, 2024.

Shicheng Xu, Danyang Hou, Liang Pang, Jingcheng Deng, Jun Xu, Huawei Shen, and Xueqi Cheng. Ai-generated images introduce invisible relevance bias to text-image retrieval. *CoRR*, abs/2311.14084, 2023. doi: 10.48550/ARXIV.2311.14084. URL https://doi.org/10.48550/arXiv.2311.14084.

Yunzhi Yao, Peng Wang, Bozhong Tian, Siyuan Cheng, Zhoubo Li, Shumin Deng, Huajun Chen, and Ningyu Zhang. Editing large language models: Problems, methods, and opportunities. In Houda Bouamor, Juan Pino, and Kalika Bali (eds.), *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing, EMNLP 2023, Singapore, December 6-10, 2023*, pp. 10222–10240. Association for Computational Linguistics, 2023. doi: 10.18653/V1/2023.EMNLP-MAIN.632. URL https://doi.org/10.18653/v1/2023.emnlp-main.632.

Yunzhi Yao, Ningyu Zhang, Zekun Xi, Mengru Wang, Ziwen Xu, Shumin Deng, and Huajun Chen. Knowledge circuits in pretrained transformers. *CoRR*, abs/2405.17969, 2024. doi: 10.48550/ARXIV.2405.17969. URL https://doi.org/10.48550/arXiv.2405.17969.

Ningyu Zhang, Yunzhi Yao, Bozhong Tian, Peng Wang, Shumin Deng, Mengru Wang, Zekun Xi, Shengyu Mao, Jintian Zhang, Yuansheng Ni, Siyuan Cheng, Ziwen Xu, Xin Xu, Jia-Chen Gu, Yong Jiang, Pengjun Xie, Fei Huang, Lei Liang, Zhiqiang Zhang, Xiaowei Zhu, Jun Zhou, and Huajun Chen. A comprehensive study of knowledge editing for large language models. *CoRR*, abs/2401.01286, 2024a. doi: 10.48550/ARXIV.2401.01286. URL https://doi.org/10.48550/arXiv.2401.01286.

Ningyu Zhang, Yunzhi Yao, Bozhong Tian, Peng Wang, Shumin Deng, Mengru Wang, Zekun Xi, Shengyu Mao, Jintian Zhang, Yuansheng Ni, et al. A comprehensive study of knowledge editing for large language models. *arXiv preprint arXiv:2401.01286*, 2024b.

Ce Zheng, Lei Li, Qingxiu Dong, Yuxuan Fan, Zhiyong Wu, Jingjing Xu, and Baobao Chang. Can we edit factual knowledge by in-context learning? In Houda Bouamor, Juan Pino, and Kalika Bali (eds.), *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing, EMNLP 2023, Singapore, December 6-10, 2023*, pp. 4862–4876. Association for Computational Linguistics, 2023. doi: 10.18653/V1/2023.EMNLP-MAIN.296. URL https://doi.org/10.18653/v1/2023.emnlp-main.296.

Zexuan Zhong, Zhengxuan Wu, Christopher D. Manning, Christopher Potts, and Danqi Chen. Mquake: Assessing knowledge editing in language models via multi-hop questions. In Houda Bouamor, Juan Pino, and Kalika Bali (eds.), *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing, EMNLP 2023, Singapore, December 6-10, 2023*, pp. 15686–15702. Association for Computational Linguistics, 2023. doi: 10.18653/V1/2023.EMNLP-MAIN.971. URL https://doi.org/10.18653/v1/2023.emnlp-main.971.

A Related Work on Knowledge Editing Benchmarks

Previous knowledge editing datasets are composed in the form of triples or fact chains. The two prominent datasets are ZsRE (Levy et al., 2017) and COUNTERFACT (Meng et al., 2022). ZsRE utilizes back translation to generate paraphrase questions, while COUNTERFACT focuses on constructing counterfactual data. The MQuAKE dataset (Zhong et al., 2023), which serves as a multi-hop knowledge editing dataset, is utilized to assess the impact of knowledge editing on intricate knowledge chains. KEBench (Wei et al., 2024c) performs a comprehensive evaluation of the stability of different knowledge editing methods using a tree-structured dataset. Furthermore, (Zhang et al., 2024a) introduced KnowEdit, an integrated evaluation benchmark that incorporates popular knowledge editing datasets to comprehensively assess various knowledge editing technologies. Simultaneously, numerous efforts (Wei et al., 2024b; Wang et al., 2023a;c) have been made to construct multilingual datasets aiming to evaluate the generalizability of knowledge editing methods across diverse languages. Recent research on expanding knowledge triples has significantly broadened the application of knowledge editing methods, particularly in handling longer text. Eva-KELLM (Wu et al., 2023) offers a benchmark dataset for evaluating document-level knowledge editing. However, this dataset creates documents by repeatedly expanding specific knowledge triples. Thus, Eva-KELLM predominantly focuses on editing specific counterfactual concepts, lacking the complexity of the unstructured knowledge editing tasks we aim to address. Similarly, KEP (Onoe et al., 2023) introduces new entity definitions into language models through knowledge editing. However, it focuses on a single entity, differing substantially from the complex and diverse unstructured knowledge editing tasks we address. EVEDIT (Liu et al., 2024) constructs a multi-sentence knowledge dataset by generating and repeating knowledge triples. While these datasets are similar in length to our proposed UnKEBench, they differ significantly in construction. UnKEBench, as an unstructured knowledge editing benchmark, features longer texts, noise, and complex, comprehensive characteristics, spanning across domains.

B Implementation Details of Constructing UnKEBench

LLMs develop significant parameter memories after undergoing comprehensive pre-training on extensive corpora. To ensure that these parameter memories do not inherently encompass editing objectives, we curate a dataset consisting of 1000 counterfactual unstructured texts. These texts are sourced from ConflictQA (Xie et al., 2024), a benchmark specifically designed to distinguish between the parameter memory of the LLM and its counter-memory. This strategy is essential to prevent the model from merging the knowledge gained during pre-training with that obtained from editing tasks. Moreover, it addresses the critical challenge of discerning whether the model has learned target knowledge during the training phase or the editing process, thus maintaining a clear demarcation between pre-training learning and editing objectives. Table 5 and 6 show the instructions for using ChatGPT (gpt-3.5-turbo) to generate original and rephrased questions for unstructured text.

C Baseline Methods

We conduct experiments on two autoregressive models, LLaMA-2-7B-Chat [2] (Touvron et al., 2023) and Qwen1.5-7B-Chat [3] (Bai et al., 2023). For baselines, we first compare the fine-tuning method **FT-L**, which targets specific layers, with **FT-A**, which fine-tunes all layers. Additionally, we assess two robust baseline models, **ROME** and **MEMIT**, focusing on their locating and editing capabilities. Lastly, we evaluate the hypernetwork-based model editing method, **MEND**.

[2] https://huggingface.co/meta-llama/Llama-2-7b-chat-hf
[3] https://huggingface.co/Qwen/Qwen1.5-7B-Chat

System:
You are given a text and asked to come up with a question that best fits it.

User:
George Rankin has been actively involved in politics for over a decade. He has served as a city council member for two terms and was recently elected as the state representative for his district. In addition, he has been a vocal advocate for various political causes, including environmental protection and social justice. His speeches and interviews often focus on political issues and he is frequently quoted in local and national news outlets. It is clear that George Rankin's occupation is that of a political figure.

Assistant:
What is George Rankin's occupation?

Table 5: Demonstrating the application of ChatGPT (gpt-3.5-turbo) in generating a question about unstructured text.

System:
You are given a question and asked to come up with a semantically similar paraphrase question.

User:
What is George Rankin's occupation?

Assistant:
What does George Rankin do for a living?

Table 6: Demonstrating the application of ChatGPT (gpt-3.5-turbo) in generating a paraphrased question from a raw question.

D Robustness Analysis on Batch Editing and Sequential Editing

To evaluate the robustness of UnKE in unstructured knowledge editing, we assess its batch editing capabilities (as shown in Table 7) and sequential editing performance (as presented in Figure 3) using the UnKEbench dataset. In the batch editing assessment, we observe that as the batch size increases, the model's performance on the original task remains relatively stable, indicating the robustness of UnKE's batch editing capabilities. However, there is a slight reduction in performance on paraphrased questions, which is expected. The simultaneous optimization of a larger number of keys marginally diminishes the model's generalization ability for paraphrased questions. For sequential editing, we find that the performance of all methods declines as the number of edits increases. Nevertheless, UnKE exhibits the highest stability compared to other baseline methods, demonstrating its robustness in sequential editing scenarios. These findings underscore the effectiveness of UnKE in handling both batch and sequential editing tasks, highlighting its potential as a promising approach for unstructured knowledge editing.

E Case Analysis of ROME, MEMIT and UnKE

Table 13 shows the generation cases of three different methods: ROME, MEMIT and UnKE. The methods of editing local key-value pairs, namely ROME and MEMIT, limit capabilities when it comes to complex unstructured knowledge editing tasks. These methods can only remember a small set of editing goals and are unable to fully retell the editing objectives. In contrast, UnKE exhibits greater proficiency in handling such tasks and is capable of conveying the editing goals.

Table 7: Comparison of different batch sizes. We conducted experiments on UnKE using the LLaMA2-7B-Chat model, with the decoding temperature set to 0.001.

Batch Size	Semantic Similarity	Lexical Similarity				FC.	General Ability
	Bert-Score	BLEU	Rouge-1	Rouge-2	Rouge-L	Fact-Score	MMLU
2^0	99.61 / 93.09	98.63 / 76.85	98.77 / 78.62	98.33 / 70.66	98.73 / 77.70	42.49	$29.68_{0.1\downarrow}$
2^1	99.41 / 91.55	98.72 / 73.03	98.88 / 74.85	98.44 / 65.35	98.83 / 73.72	42.35	$29.66_{0.12\downarrow}$
2^2	99.48 / 89.98	98.97 / 69.95	98.98 / 71.83	98.61 / 60.93	98.93 / 70.54	41.82	$29.61_{0.17\downarrow}$
2^3	99.57 / 88.33	98.99 / 66.10	99.08 / 67.98	98.76 / 56.16	99.05 / 66.61	42.24	$29.66_{0.12\downarrow}$
2^4	99.70 / 85.98	99.17 / 62.30	99.28 / 64.76	99.01 / 51.97	99.25 / 63.22	42.13	$29.61_{0.17\downarrow}$
2^5	99.56 / 84.07	99.12 / 59.36	99.16 / 61.45	98.89 / 47.57	99.14 / 59.71	41.38	$29.73_{0.05\downarrow}$
2^6	99.78 / 85.21	99.47 / 60.38	99.50 / 62.25	99.31 / 48.55	99.48 / 60.50	42.93	$29.72_{0.06\downarrow}$

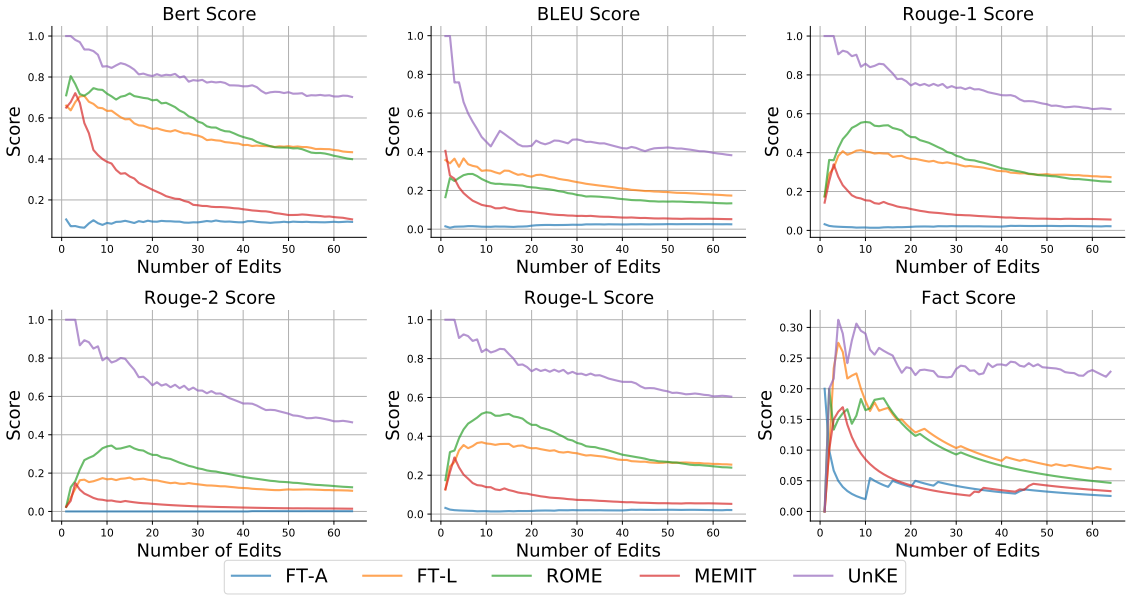

Figure 3: Performance in sequential editing. We select the first 64 samples in the UnKEBench data set for sequential editing experiments.

F Impact of hyperparameter L on model performance

In Section 3.1, we hypothesize that the shallow layer of LLM encodes the key vector of knowledge, dividing LLM into key generator and value generator based on the hyperparameter L. To investigate the influence of the value of L on model performance, we conducted analytical experiments, as presented in Table 8.

The results indicate that when the value of L is small, such as 5-10, the model performance remains relatively stable, effectively managing unstructured knowledge. However, as L increases further, it becomes apparent that the model's effectiveness diminishes. This shows that at this time, L is too deep, resulting in the key vector that has stored the target information, so it is difficult to edit.

Table 8: Optimization layer L selection experiment.

L	Semantic Similarity	Lexical Similarity				FC.	General Ability
	Bert-Score	BLEU	Rouge-1	Rouge-2	Rouge-L	Fact-Score	MMLU
5	98.30 / 90.15	95.02 / 70.30	94.81 / 70.86	92.98 / 60.28	94.59 / 69.57	40.37	$29.56_{0.22\downarrow}$
6	99.23 / 91.51	97.24 / 72.29	97.54 / 73.72	96.55 / 64.28	97.45 / 72.60	42.11	$29.72_{0.06\downarrow}$
7	99.61 / 93.09	98.63 / 76.85	98.77 / 78.62	98.33 / 70.66	98.73 / 77.70	42.49	$29.68_{0.10\downarrow}$
8	99.62 / 94.29	98.57 / 80.91	98.79 / 83.02	98.24 / 76.23	98.73 / 82.13	40.86	$29.68_{0.10\downarrow}$
9	98.09 / 93.17	92.71 / 74.39	94.35 / 79.47	91.99 / 70.99	94.05 / 78.38	41.33	$29.70_{0.08\downarrow}$
10	95.53 / 91.82	81.50 / 67.10	86.14 / 74.02	80.39 / 63.46	85.42 / 72.66	39.72	$29.71_{0.07\downarrow}$
11	90.04 / 86.73	68.96 / 56.72	76.01 / 67.03	66.68 / 53.70	74.85 / 65.28	27.33	$29.64_{0.14\downarrow}$
12	87.47 / 84.90	53.17 / 44.92	65.17 / 59.16	52.43 / 43.74	63.57 / 57.22	22.78	$29.67_{0.11\downarrow}$

Table 9: Correlation between human and automatic evaluation metrics.

Metrics	BLEU	Rouge-1	Rouge-2	Rouge-L	Bert-Score	FactScore
Correctness	$98.55_{0.0021}$	$95.04_{0.0132}$	$95.74_{0.0105}$	$98.10_{0.0031}$	$97.52_{0.0047}$	$96.90_{0.0065}$
Similarity	$94.74_{0.0144}$	$91.54_{0.0292}$	$91.23_{0.0308}$	$94.50_{0.0153}$	$97.67_{0.0042}$	$92.18_{0.0260}$
Coherence	$95.64_{0.0108}$	$92.37_{0.0250}$	$91.90_{0.0273}$	$95.41_{0.0117}$	$98.68_{0.0018}$	$94.04_{0.0173}$

G Correlation between automatic and human evaluation metrics

Tables 9 and Table 10 display the Pearson correlation coefficients between the human evaluation metrics and the original question metrics and the paraphrase question metrics, respectively. Due to significant differences in the evaluation dimensions of the general ability metric MMLU and the three human evaluation metrics, it is omitted from the table.

Each cell in the table represents the correlation coefficient between the corresponding automatic evaluation metric and the human evaluation metric, with the subscript indicating the p-value. Almost all correlation coefficients are above 0.95, confirming a strong correlation between the human and automated assessment results. Additionally, the p-values for all metrics are below 0.05, indicating that the correlations are statistically significant.

H Experiment Details

Except for UnKE, we use EasyEdit [4] (Wang et al., 2023b)to implement all other editing methods, including fine-tuning. For all other baselines, except for the necessary modifications that need to be applied to UnKEBench, we use the official default hyperparameters, which can be easily reproduced in the official library. The optimizer type used when it comes to gradient descent is Adam. The following are their important hyperparameter configuration contents.

Table 11: Comparison of running time of each method. Time is in hours.

Method	Time	Method	Time
FT-L	24	ROME	21
FT-A	31	MEMIT	27.75
MEND	38	UnKE	10.5

[4] https://github.com/zjunlp/EasyEdit

Table 10: Correlation between human and automatic evaluation metrics (Para.).

Metrics	BLEU	Rouge-1	Rouge-2	Rouge-L	Bert-Score
Correctness	$98.68_{0.0018}$	$92.38_{0.0249}$	$95.68_{0.0107}$	$97.83_{0.0038}$	$97.66_{0.0043}$
Similarity	$94.97_{0.0135}$	$89.13_{0.0423}$	$91.13_{0.0313}$	$94.50_{0.0153}$	$97.95_{0.0035}$
Coherence	$95.85_{0.0101}$	$89.88_{0.0380}$	$91.82_{0.0277}$	$95.38_{0.0119}$	$98.87_{0.0014}$

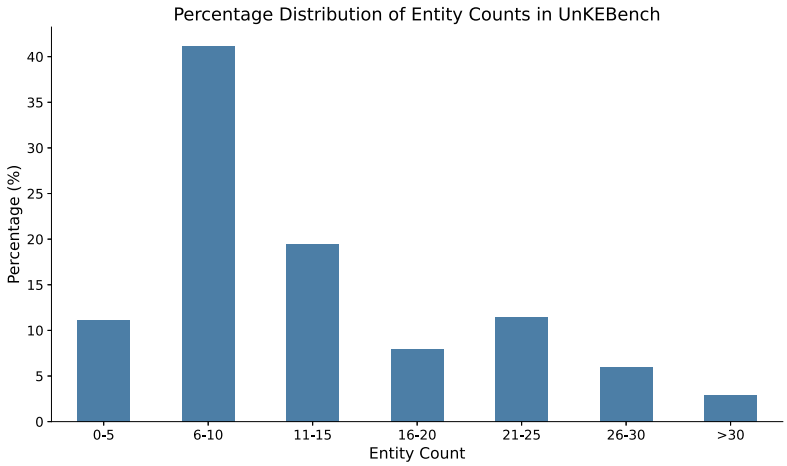

Figure 4: The X-axis represents the number of entities contained in unstructured text, while the Y-axis indicates the proportion of sentences containing that number of entities among all sentences.

Fine-tuning Fine-tuning consists of two variants: FT-L and FT-A, with the only distinction being the number of layers involved in parameter updates. The maximum length is set to 1024, and a learning rate of 5×10^{-4} is utilized. Each sample undergoes 25 optimization steps. The layer where FT-L parameters are updated is layer 7, which is consistent with UnKE.

ROME and MEMIT The primary distinction between ROME and MEMIT lies in the number of editing layers. ROME focuses on editing the layer 7, while MEMIT targets the layers [4,5,6,7,8]. Both approaches undergo 25 optimization steps, utilizing a learning rate of 5^{-1}, a weight attenuation coefficient of 1×10^{-3}, and a KL factor of 0.0625. Before the editing process, approximately 100,000 Wikipedia samples need to be computed, which is a highly time-consuming task.

MEND MEND enables concurrent edits by accumulating gradients from all edit examples and passing them through the hypernetwork simultaneously. It calculates parameter layers 29, 30, and 31 and utilizes a learning rate of 1×10^{-4}. Due to the presence of numerous hyperparameters, it is advisable to refer to the official website or code library for detailed information.

UnKE UnKE employs a two-stage structuring process. In the first stage, the learning rate is set to 5e-1, with 25 optimization steps and a weight attenuation coefficient of 1e-3. In the second stage, the learning rate is set to 2e-4, and 50 optimization steps are performed. All experiments conducted on UnKE in this article focus on optimizing layer 7. During each optimization iteration, an additional 20 samples are randomly

System:
You are asked to generate some short question-answer pairs based on the specified text. These question-answer pairs mainly ask questions about the knowledge entities in the text, and the answers should be the knowledge entities being asked.

User:
George Rankin has been actively involved in politics for over a decade. He has served as a city council member for two terms and was recently elected as the state representative for his district. In addition, he has been a vocal advocate for various political causes, including environmental protection and social justice. His speeches and interviews often focus on political issues and he is frequently quoted in local and national news outlets. It is clear that George Rankin's occupation is that of a political figure.

Assistant:
Question: How long has George Rankin been involved in politics?
Answer: Over a decade.
Question: What political positions has George Rankin held?
Answer: City council member and state representative.
Question: What causes has George Rankin advocated for?
Answer: Environmental protection and social justice.
Question: What is George Rankin's occupation?
Answer: A political figure.

Table 12: Demonstrating the application of ChatGPT (gpt-3.5-turbo) in generating some short question-answer pairs based on the unstructured text.

selected from the alpaca instruction fine-tuning data [5]. It is important to note that this number is significantly less than what is required by ROME and MEMIT.

Time costs We present the runtime of each knowledge editing method on UnKEBench. It is important to note that our evaluation approach differs from the EasyEdit framework. Instead, we edit a sample and subsequently allow the model to generate the corresponding problem results for evaluation. This process may take slightly longer. Additionally, ROME and MEMIT require additional time for calculating Wikipedia knowledge. MEND adds the time to calculate the one-stage model. Please refer to the Table 11 for detailed results. All experiments can be completed on an A800-80G graphics card.

I THE BASIC STATISTICS OF UNKEBENCH

As shown in Figure 4, only 10% of the sentences contain 0-5 entities, which is a common range for structured text editing tasks. In contrast, over 90% of the sentences contain more than 5 entities, highlighting the inherent complexity of unstructured text.

J BROADER IMPACTS

UnKE offers the advantage of timely updating outdated knowledge in LLMs, thereby reducing costs for developers and eliminating the need for resource-intensive pre-training or fine-tuning steps. However, it is important to acknowledge the potential misuse of UnKE by malicious actors who may attempt to inject harmful information into LLMs. To mitigate this risk, a knowledge detection system can be employed to identify and prevent the inclusion of malicious content.

[5] https://github.com/tatsu-lab/stanford_alpaca

Question : What were Charles Strachey's main contributions to British politics and law during the 19th century?

Text: Charles Strachey, 4th Baron O'Hagan was a prominent political figure in the United Kingdom during the 19th century. He served as a Member of Parliament for the Liberal Party from 1868 to 1874, representing the constituency of Nottingham. During his time in Parliament, he was known for his advocacy of social reform and his support for the rights of workers. In addition to his political career, Strachey was also a respected lawyer and judge, serving as a Queen's Counsel and a Justice of the Peace. His contributions to British politics and law have been widely recognized, and he remains an important figure in the country's history.

UnKE Prediction : Charles Strachey, 4th Baron O'Hagan was a prominent political figure in the United Kingdom during the 19th century. He served as a Member of Parliament for the Liberal Party from 1868 to 1874, representing the constituency of Nottingham. During his time in Parliament, he was known for his advocacy of social reform and his support for the rights of workers. In addition to his political career, Strachey was also a respected lawyer and judge, serving as a Queen's Counsel and a Justice of the Peace. His contributions to British politics and law have been widely recognized, and he remains an important figure in the country's history.

ROME Prediction : Charles Strachey was a prominent British politician and lawyer who made significant contributions to the country's political and legal landscape during the 19th century. Here are some of his key contributions: 1. Parliamentary Reform: Strachey was a strong advocate for parliamentary reform and played a key role in the passage of the Reform Act of 1832. This act expanded the franchise and redrew electoral boundaries, giving more people the right to vote and increasing the representation of the working class in Parliament. 2. Legal Reform: Strachey was also a leading figure in the legal profession and was instrumental in the passage of the Law Reform Act of 1832. This act aimed to modernize the law and reduce the complexity of legal proceedings, making it easier for .....

MEMIT Prediction: Charles Strachey (1796-1867) was a prominent British lawyer and politician who made significant contributions to British politics and law during the 19th century. Here are some of his main contributions: 1. Legal Reforms: Strachey was a strong advocate for legal reform and was instrumental in the passage of the Law Amendment Act of 1870, which established the office of Law Society Council and paved the way for the modernization of the legal profession. He also played a key role in the establishment of the Incorporated Law Society of Scotland. 2. Parliamentary Reforms: Strachey was a member of the House of Commons and served as a vocal advocate for parliamentary reform. .....

Table 13: This table presents the problems and their corresponding unstructured text after editing. It can be observed that the predicted text generated by UnKE is almost identical to the original text. However, the texts generated by methods like ROME and MEMIT only edit a few key knowledge points such as 'political' and 'parliamentary reform', while the detailed descriptions of these knowledge points are almost entirely incorrect.