DocMEdit: Towards Document-Level Model Editing

Li Zeng[1*], Zeming Liu[2*], Chong Feng[1], Heyan Huang[1], Yuhang Guo[1†]
[1]School of Computer Science and Technology, Beijing Institute of Technology, Beijing, China
[2]School of Computer Science and Engineering, Beihang University, Beijing, China
{zengli, fengchong, hhy63, guoyuhang}@bit.edu.cn zmliu@buaa.edu.cn

Abstract

Model editing aims to correct errors and outdated knowledge in the Large language models (LLMs) with minimal cost. Prior research has proposed a variety of datasets to assess the effectiveness of these model editing methods. However, most existing datasets only require models to output short phrases or sentences, overlooks the widespread existence of document-level tasks in the real world, raising doubts about their practical usability. Aimed at addressing this limitation and promoting the application of model editing in real-world scenarios, we propose the task of document-level model editing. To tackle such challenges and enhance model capabilities in practical settings, we introduce DocMEdit, a dataset focused on document-level model editing, characterized by document-level inputs and outputs, extrapolative, and multiple facts within a single edit. We propose a series of evaluation metrics and experiments. The results show that the difficulties in document-level model editing pose challenges for existing model editing methods[1].

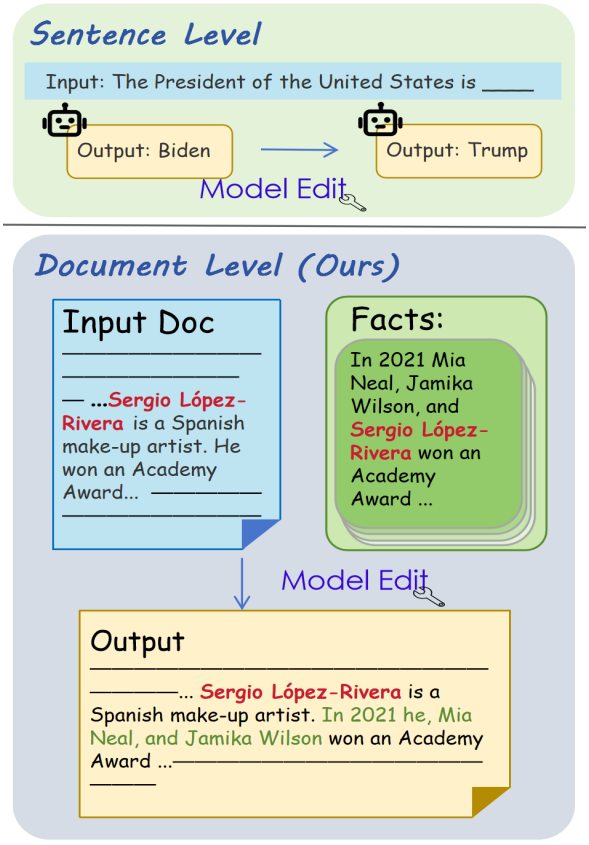

Figure 1: An example of DocMEdit. The input and output of DocMEdit are both document-level contents. Model editing should inject multiple facts to be edited into the model, enabling the edited model to output the updated document.

1 Introduction

Large language models (LLMs) have demonstrated exceptional performance across a wide range of fields and are widely applied in various practical scenarios (Touvron et al., 2023a,b; Wang et al., 2024a; Geva et al., 2021, 2022). Given their broad usage, it is crucial for LLMs to deliver accurate and reliable information. However, LLMs may still generate incorrect or outdated information due to the knowledge they store, which can be inaccurate (De Cao et al., 2021; Agarwal and Nenkova, 2022). Such inaccuracies can lead to serious consequences in critical domains, such as medical diagnoses and legal advice, highlighting the importance of methods to correct errors in LLMs. To address this issue without expensive retraining, model editing techniques have been proposed (Mitchell et al., 2022; Sinitsin et al., 2020; De Cao et al., 2021).

To evaluate the effectiveness of model editing methods, previous researchers have proposed a range of datasets encompassing various tasks such as single-hop and multi-hop question answering, cloze tasks, and others (Zeng et al., 2024; Xie et al., 2024; Gu et al., 2024). To verify whether a model

*Equal contribution
†Corresponding author: guoyuhang@bit.edu.cn
[1]Dataset and codes are publicly available at https://github.com/BITHLP/DocMEdit

Findings of the Association for Computational Linguistics: ACL 2025, pages 19725–19743
July 27 - August 1, 2025

knows specific facts, almost all of these datasets require the model to output a short phrase or sentence. As shown in figure 1, prior datasets only demand shorter textual outputs from the model. However, in real-world scenarios, document-level tasks such as generating biographies, long chains of thought, or updating Wikipedia documents are more common. Moreover, the document-level model editing challenge for model editing methods lies in the need to extrapolate answers from facts, handle longer contexts, and deal with multiple facts within a single document. Due to the lack of research on these challenges, the application of existing model editing methods to practical LLMs is limited.

To promote the application of model editing in document-level tasks, we propose a new task: **document-level model editing**. Document-level model editing requires both the input and output to be at the document level. Additionally, the model cannot infer the answer from the given facts; it involves longer contexts and multiple facts within a single edit.

Aimed at addressing the lack of practical applicability in existing benchmarks and promoting model editing at the document level, we propose a novel benchmark: DocMEdit, a model editing dataset that contains document-level data. The input and output of DocMEdit are both at the document level. Besides, the model cannot derive the answer solely from the facts to be edited; instead, it must combine multiple facts to be edited with existing knowledge to produce the updated document. Unlike most previous research, our editing facts are not derived from triples but are directly extracted from unstructured data in Wikipedia, which is more aligned with real-world model editing scenarios. DocMEdit contains 37,990 data items, with an average context length of 1,535.5 per item. It includes 105,652 editing facts, with an average of 2.78 facts per data item. Additionally, we extracted triples from DocMEdit that share the same relations as those in Wikidata to facilitate experiments with RAG-based methods (Zeng et al., 2024; Zhang et al., 2024b,a).

To validate the effectiveness of existing model editing methods for document level model editing tasks, we propose a series of new metrics and conduct experiments on DocMEdit. We also further discuss how each challenge in document level model editing affects the performance of model editing methods. The conclusion demonstrates that existing methods have low accuracy while exhibiting strong side effects. Further research confirms that factors such as document length, fact length, number of facts, and fact updates all impact the effectiveness of model editing methods.

The main contributions of this paper are as follows:

- We are the first to propose the document-level task for model editing, which is more aligned with real-world LLM use situations.
- To support research in document-level model editing, we create DocMEdit, a novel benchmark that includes longer contexts and multiple parallel edit facts within a single document.
- Experiments show that existing methods have low accuracy and significant side effects. Further research demonstrates that the length of the document and facts, the number of facts, and the fact updates all impact the performance of model editing methods.

2 Related work

2.1 Model Editing Datasets

Model editing datasets serve the purpose of verifying the effectiveness of methods and enhancing the capability of LLMs. However, existing datasets focus on question answering with shorter contexts, neglecting the common scenario of long context input-output pairs in practice, i.e., document-level model editing. Tasks in existing datasets include QA, sentence completion, choose, and cloze tests (Zeng et al., 2024; Zhong et al., 2023; Levy et al., 2017a; Meng et al., 2022; Zhang et al., 2024c; Wang et al., 2024c). Wu et al. (2024) explored model editing for longer texts, however, their expected output is still at the sentence level, and no supporting facts are provided. Unlike previous research, our input and output are both at the document level. Additionally, we provide multiple facts to be edited and require the LLMs to generate an updated document based on these facts.

2.2 Document-Level NLP

Currently, some existing researches discuss the challenges that arise when common NLP tasks are scaled to the document level, such as translation (Wang et al., 2023), relation extraction (Xue et al., 2024; Zheng et al., 2024), and QA (Rasool et al., 2024). These researches highlight the lack of suitable datasets, evaluation methods, and the limitations of existing models in accurately information

retrieve. However, they all overlook the field of model editing. In contrast, we are the first to scale model editing to the document level, introducing a corresponding dataset, experiments, and evaluation metrics.

3 Problem Definition

Model editing aims to modify the knowledge contained within a model, changing the output related to the facts to be edited while keeping other outputs unchanged. Based on previous research (Zeng et al., 2024; Zhang et al., 2024c; Yao et al., 2023), we define document level model editing as follows:

The document-level model editing task aims to modify the document-level output of a large language model based on a set of facts to be edited while keeping the parts unrelated to these facts unaffected. Specifically, let the large language model be denoted as $\mathcal{M}$, with the input and output of the model represented as x and y, respectively, i.e., $y = \mathcal{M}(x)$. The output y consists of multiple sentences, represented as $y = \sum_{i=1}^{n} s_i$, where s_i are the sentences of the original document. In the document level model editing, the original large language model $\mathcal{M}$ generates an output y based on the input document x, and then the model is edited using a set of facts to be edited $F = \sum_{j=1}^{m} f_j$, resulting in the edited model $\mathcal{M}'$, which generates a new output document $y' = \mathcal{M}'(x)$.

The new document y' consists of the original sentences as well as newly added sentences supported by the facts to be edited. It is represented as $y' = \sum_{i=1}^{n} s_i + \sum_{j=1}^{k} s_{f_j}$, where s_i are the sentences consistent with the original text, and $s_{fj}$are the new sentences supported by the facts to be edited. In this process, it is required that all sentences unrelated to the facts to be edited, s_i, remain unchanged, while the newly added sentences,s_{fj}, must be content supported by the facts f_j.

4 DocMEdit: A Dataset Dedicated to Document Level Model Editing

To advance the practical application of document-level model editing, we introduce DocMEdit (**Doc**ument level **M**odel **Edit**). The input and output in DocMEdit are document-level, with each document containing one or more facts to be edited. We also extract triples from each document and fact to facilitate the use of knowledge graph-based model editing methods.

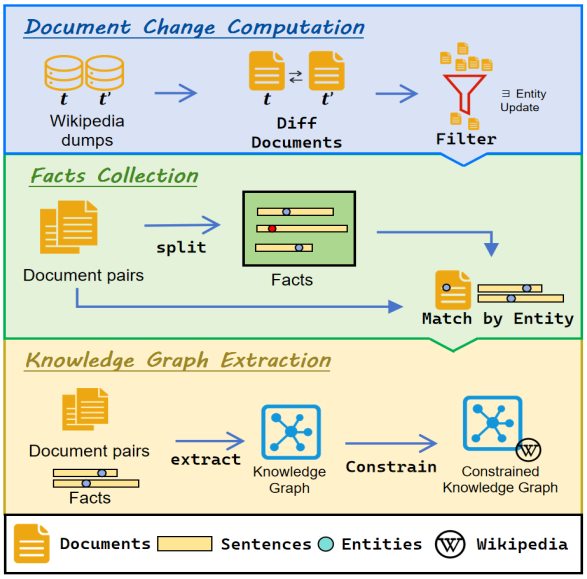

Figure 2: The construction process of DocMEdit. In the Document Change Computation, we calculate the updates of documents in the Wikipedia between two time points and retain those documents that exhibit entity updates. In the Facts Collection, based on the newly added entities within the documents, we identify the newly added sentences mentioning these entities and extract them as supporting facts. In the Knowledge Graph Extraction, we extract structured knowledge graphs and impose constraints based on Wikidata relations.

4.1 Dataset Construction

Following Iv et al. (2022), we construct DocMEdit with the following steps: (1) Document Change Computation; (2) Facts Collection; (3) Knowledge Graph Extraction. The detailed steps are as follows:

4.1.1 Document Change Computation

We collect Wikipedia dumps from two timestamps (20231101 and 20241101). For each document in the dump, we extract its INTRODUCTION section, using the corresponding sections from two timestamps as y and y', respectively. We consider this as reflecting the updates in Wikipedia. Since most Wikipedia updates are stylistic rather than factual (Daxenberger and Gurevych, 2012), we filtered out updates that did not include the addition of at least one entity to ensure that the extracted updates were meaningful.

4.1.2 Facts Collection

We extract sentences related to entities from the document as facts. Specifically, following the assumption of Iv et al. (2022), for each sentence, if an entity mentioned in the sentence was newly in-

Benchmark	Doc level	Extrapolative	Multi-Edits	Locality	Avg. Target Len.	Avg. Facts Len.	Total
ZSRE (Levy et al., 2017b)	✗	✗	✗	✓	12.12	56.39	270.0K
COUNTERFACT (Meng et al., 2022)	✗	✗	✗	✓	6.65	39.14	2.2K
MQUAKE (Zhong et al., 2023)	✗	✗	✓	✗	10.94	145.09	11.1K
WIKIBIO (Manakul et al., 2023)	✓✗	✗	✗	✗	131.10*	131.10*	1.4k
FAME (Zeng et al., 2024)	✗	✗	✓	✓	13.01	62.75	128.0K
DOCMEDIT (Ours)	✓	✓	✓	✓	867.62	623.40	38.0K

Table 1: Comparison of benchmarks."Doc level" refers to whether the input and output of the dataset are at the document level. ✓✗ means the input is at the document level, but the output is at the sentence level. "Extrapolative" refers to whether the answer to each question requires inference based on the existing knowledge within the LLM, or if it can be directly derived from the given facts alone. "Multi-Edits" refers to whether multiple facts to be edited are included in a single editing target. "Locality" refers to whether the dataset design takes into account the evaluation of side effects. "Avg. Target Len." refers to the average expected output length per data item, and "Avg. Facts Len." denotes the average total length of the facts used per data item. "Total" represents the total amount of data. "*" indicates that the expected output is identical to the facts to be edited.

troduced in the document update, the sentence was considered to support the document update f_i of the corresponding entity.

4.1.3 Knowledge Graph Extraction

Given that many model editing methods leverage knowledge graphs relevant to facts for better performance (Zeng et al., 2024; Zhang et al., 2024a), we utilize the framework proposed by Schmitz et al. (2012) to extract knowledge graphs. We extract knowledge graphs from the source document, the target document, and the supporting facts, respectively. Then we constrain the relation r in the triple (s, r, o) to be an existing relation in Wikidata, which ensures that our extracted knowledge graphs are well-structured and consistent.

DocMEdit	Data Items	37990
	Avg. Context Len.	1535.5
	Facts	105651
	Avg. Facts	2.78
knowledge graph	Entities	568652
	Relations	4804
	Triples	1411057

Table 2: Dataset statistics. "Data Items" refers to the total number of data entries, and "Facts" denotes the facts to be edited. We also collected statistical information on the extracted triples, which were derived from the input documents, output texts, and editing facts. Note that the context length includes both the input and the expected output, while not including the facts.

4.2 Quality Control

Following Iv et al. (2022), we adopt the following measures to ensure the quality of our dataset: (1) manually removing unsupported updates, and (2) comparing the annotated data with automatically collected text to verify the reliability of the data collection process. We evaluate our dataset using the metrics described in Section 5.3, yielding a DR of 81.17 and a DE of 89.71, which demonstrate a high consistency between our dataset and human annotations, confirming that facts in our dataset are aligned with the documents and that the document updates are well-supported.

4.3 Benchmark Analysis

4.3.1 Comparison

DocMEdit is distinguished by its document-level input and output. Additionally, it is uniquely characterized by the feature that the expected output cannot be directly derived from the facts to be edited. DocMEdit includes cases where a document corresponds to multiple facts, while also allowing for the testing of side effects in model editing methods. Finally, DocMEdit has an advantage in terms of context length, which has been overlooked by previous datasets.

4.3.2 Statistics

Table 2 presents the statistics of DocMEdit. We have compiled statistics on the documents, facts, and the extracted knowledge graph. For data samples and additional statistics, please refer to Appendix B.

5 Experiment

In this section, we present our main experiments. We introduce the edited models used in section 5.1, the baseline model editing methods in section 5.2, and the evaluation metrics in section 5.3. We analyze the results of our experiments in section 5.4. For more experimental details, please refer to Appendix C.

5.1 Language Models

Following Zeng et al. (2024), we use LLMs of various sizes to evaluate the performance of the model editing method in handling diverse scenarios. Since we aim to test the update of internal knowledge in LLMs, we want the model to be unaware of the updated facts during training so we intentionally choose earlier released LLMs. These models include GPT-2 XL (Solaiman et al., 2019), GPT-J-6B (Wang and Komatsuzaki, 2021), Llama2 (Touvron et al., 2023b), Mistral-7B-Instruct-v0.3 (Jiang et al., 2023) and Deepseek-V3 (Liu et al., 2024a).

5.2 Baselines

Following Zhang et al. (2024d), we selected the following methods as baselines for evaluation. For parameter modification methods, we choose FT and MEMIT, and for parameter preservation approaches, we select IKE, SKEME, and EREN as baselines. Please refer to Appendix C.6 for the implementation details.

FT The most classic and straightforward model-editing method is fine-tuning. Following previous research (Meng et al., 2023), we apply Fine-Tuning to the given layer of the model.

MEMIT (Meng et al., 2023) Currently considered a state-of-the-art method among parameter modification methods.

IKE (Zheng et al., 2023) Direct embedding similarity-based RAG method.

SKEME (Zeng et al., 2024) RAG method based on knowledge graphs and caching systems, which allows for more precise knowledge retrieval.

EREN (Chen et al., 2024) RAG method based on a growing notebook, capable of handling updates to multiple facts.

5.3 Metrics

Following the experimental setups of previous studies (Meng et al., 2023; De Cao et al., 2021; Yao et al., 2023), we evaluates both the effectiveness and side effects of model editing. In addition, we adopt the evaluation protocols from Meng et al. (2022) and Zeng et al. (2024) to assess the generation quality of the edited models as well as the editing efficiency of different model editing methods.

Accuracy Accuracy is used to evaluate the effectiveness of editing. We measure accuracy on two levels: at the level of each editing target (i.e., each document update) and the level of each piece of fact. The former evaluates the overall outcome of the update, while the latter assesses whether the model has appropriately utilized each piece of fact.

For both levels, we employ two metrics to evaluate editing performance: ROUGE and an entity-based measurement. For ROUGE, we use UpdateROUGE following Iv et al. (2022), which only computes the score on the parts of the source and target documents that were actually updated. For entity-based evaluation, we extracted all entities involved in the documents or sentences to assess the ability of methods to update facts. These two levels and two metrics result in four evaluation methods: Document-ROUGE (**DR**), Document-Entity (**DE**), Edit-ROUGE (**ER**), and Edit-Entity (**EE**). Please refer to Appendix C.4 for the formal definitions of thest metrics.

Locality Model editing requires that edits do not affect outputs unrelated to the edited facts. In document-level model editing, outputs unrelated to the edited facts correspond to the unchanged parts of the document before and after the update. We evaluate the side effects of model editing methods by computing both ROUGE and entity-based differences on the unchanged parts of the document, denoted as ROUGE Side Effect (**RSE**) and Entity Side Effect (**ESE**). Please refer to Appendix C.4 for the formal definitions of thest metrics.

Quality Meng et al. (2022) recommend evaluating the impact of model editing on the generation quality of large language models. Following Liu et al. (2023, 2020), we adopt human evaluation to evaluate the semantic coherence (**SC**) of model outputs. Specifically, the outputs of large language models are categorized into three quality tiers, with detailed scoring criteria provided in Appendix C.5. Note that since this metric relies on human evaluation, we assess a sample of 100 data points for each method.

Efficiency To evaluate whether model editing methods can perform edits efficiently, we follow the approach of previous researches (Zeng et al., 2024; Yao et al., 2023) by measuring time consumption (**Ti**) and memory requirements (**Me**).

Model	Method	Accuracy				Locality		Quality	Efficiency	
		DR↑	DE↑	ER↑	EE↑	RR↑	ER ↑	SC ↑	Ti ↓	Me ↓
GPT2-XL	w/o Edit	13.66	24.12	13.71	2.25	**43.34**	**31.95**	1.01	9.35	8.98
	FT	12.08	11.72	11.55	3.00	37.36	18.07	0.54	10.16	12.84
	MEMIT	11.86	9.64	12.05	3.00	37.41	13.88	0.64	**10.06**	12.84
	IKE	**14.10**	**31.77**	**14.93**	**9.25**	34.38	17.82	1.02	10.58	10.02
	SKEME	10.71	15.59	11.98	5.50	36.11	12.55	**1.43**	10.62	**9.98**
GPT-J	w/o Edit	**22.08**	16.42	17.26	5.50	**53.37**	**52.86**	1.05	40.43	24.62
	FT	2.44	**35.00**	4.90	1.00	33.38	1.53	0.48	67.56	31.49
	MEMIT	9.79	22.82	10.17	5.50	33.99	6.10	0.52	72.94	32.58
	IKE	17.05	29.68	16.81	11.58	40.01	26.22	0.98	42.56	**27.62**
	SKEME	19.53	28.60	**25.77**	**23.83**	42.05	38.33	**1.08**	**40.57**	28.22
Llama2	w/o Edit	**26.11**	18.97	15.77	0.50	**53.91**	**55.37**	**1.05**	30.84	27.64
	FT	24.78	17.95	14.65	7.17	53.76	39.22	0.60	47.51	31.49
	MEMIT	19.63	9.62	15.16	2.50	40.54	34.86	0.62	64.42	32.58
	IKE	19.79	26.30	22.77	12.20	43.27	35.80	1.03	**31.70**	33.67
	SKEME	21.08	**29.34**	**25.75**	**23.92**	47.31	49.22	1.00	32.28	**30.92**
Mistral	w/o Edit	3.62	4.56	1.95	1.25	33.39	8.59	1.94	23.92	28.34
	IKE	5.29	24.02	5.52	10.00	33.58	11.79	**1.95**	**26.17**	29.37
	EREN	**11.26**	27.94	12.70	**18.77**	35.63	**25.60**	1.92	26.18	30.31
	SKEME	10.11	**35.03**	**12.94**	11.75	**36.27**	24.17	**1.95**	27.64	**29.26**
DeepSeek	w/o Edit	34.81	23.62	19.08	27.22	54.97	88.80	**1.99**	-	-
	IKE	37.43	**37.68**	25.41	45.45	**63.76**	**92.35**	1.93	-	-
	EREN	36.91	23.05	29.48	**55.90**	61.29	90.16	1.91	-	-
	SKEME	**37.71**	37.05	**29.64**	54.49	59.04	88.55	**1.99**	-	-

Table 3: Main result on DocMEdit. "DR", "DE", "ER", "EE", "RSE", "ESE", "SC", "Ti", and "Me" stand for Document-ROUGE, Document-Entity, Edit-ROUGE, Edit-Entity, ROUGE Side Effect, Entity Side Effect, semantic coherence, time consumption, and memory requirements, respectively. "w/o Edit" represents the unedited original model. Bold indicates the best-performing method, while underlining denotes the second-best. In the evaluation of editing efficiency, comparisons are made only among methods that involve model editing. "-": Editing efficiency is unavailable for DeepSeek, as it is evaluated via API calls.

5.4 Main Result

Table 3 shows the results of our main experiments. It can be observed that all the base models fail to handle the document-level model editing. We have the following conclusions.

The overall performance of the methods was below expectations. For each base model, when they have not been edited, they achieved a relatively high DR, indicating that the generated documents are quite similar to the target edits. However, since no editing facts were provided to the base models, both DE and EE are low, which suggests that a high DR may be attributed to hallucinations, whereas DE and EE better reflect fact-based modifications.

We found that methods involving parameter modifications have lower ER and EE compared to RAG-based methods, implying that they have a lower ability to edit facts in the document-level model editing task setup. The methods for modifying parameters all cause a certain degree of degradation in the generation quality of LLMs, which may be due to partially impairing the generative capability of these models.

For RAG-based methods, EREN does not achieve satisfactory results. We found that the core objective of EREN is to determine whether a given fact is relevant to the input. However, in our dataset, every provided fact is related to the document being edited. The challenge for the model is to decide which facts should be utilized and where they should be incorporated, which differs from the relevance problem tackled by EREN. For SKEME and IKE, since SKEME relies on knowledge graph search while IKE uses vector knowledge base search, and fact retrieval based on documents is more challenging, SKEME has a significant advantage (see Appendix D for comparisons).

All models suffer from significant side effects. We found that each model exhibits serious side effects. For the base model, the ESE of all LLMs is below 60, meaning that they lose more than 40% of the entity information. After editing, all models show a significant decrease in both RES and ESE, indicating that these methods have a substantial

impact on parts of the document unrelated to the edited facts.

6 Analysis

In this section, we conduct a deeper analysis to examine the key differences between document-level model editing and traditional model editing, as well as the challenges posed by these differences.

We contend that the main differences between document-level model editing and traditional model editing are: 1) the length of the input and facts to be edited is longer, and 2) multiple parallel facts to be edited are allowed within a single editing objective.

To substantiate our conclusions, we design a series of research questions (RQs) and analyze their impact on editing effectiveness. For the first aspect, we propose the following research questions: **RQ1a**: The impact of context length on editing. **RQ1b**: The impact of fact length on editing. For the second aspect, we propose: **RQ2a**: The impact of the number of edited facts per document on the results. **RQ2b**: The impact of fact updates on model editing performance.

According to the findings of the main experiments, entity-based metrics are more indicative of fact-based modifications. Therefore, in the analysis, we use DE to evaluate document-level RQ1a and RQ2b, and adopt EE to assess edit-level RQ2a. For all RQs, we conduct experiments on Llama2.

6.1 RQ1a: The Impact of Context Length on Editing.

To analyze the impact of context length on the model's output, we categorized the data based on context length and then evaluated the performance of each method within each context length range, the results are shown in Figure 3.

Figure 3 presents the results of RQ1. For shorter documents (length 0-512), the base model can already handle some cases, while IKE and SKEME further improve performance. However, FT and MEMIT actually degrade the model's performance. For longer documents (length 512-1024), only RAG-based learning methods (IKE and SKEME) are effective. For even longer documents, almost all methods fail to generate successfully edited documents.

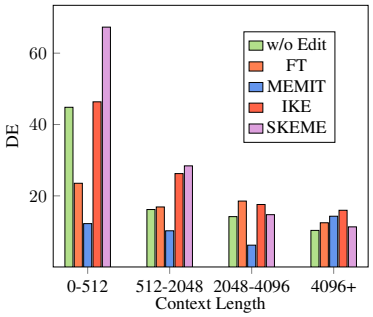

Figure 3: result of RQ1a. The x-axis represents the context length of the data, while the y-axis represents the corresponding DE.

6.2 RQ1b: The Impact of Fact Length on Editing.

To analyze the impact of fact length on editing performance, we measure the EE of each fact and arrange them according to the length of the facts.

Figure 4 presents the results of RQ1b. The findings indicate that as the length of the facts increases, the editing performance of all methods declines. Additionally, we observe that when the facts are relatively short, FT can achieve a certain level of editing effectiveness, whereas for longer facts, RAG-based methods become a better choice.

Further analysis of the two RAG-based methods reveals that as fact length increases, the performance of IKE drops rapidly, while SKEME experiences a more gradual decline. We attribute this to the increased difficulty of retrieving longer facts using vector-based retrieval in IKE, while entity-based retrieval in SKEME remains relatively robust. Therefore, although both IKE and SKEME are affected by the LLM's ability to utilize knowledge, IKE also suffers from the negative impact of fact length on its retrieval capability.

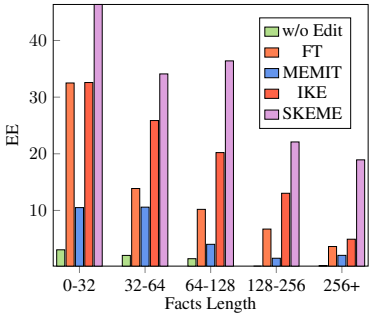

Figure 4: result of RQ1b. The x-axis represents the length of the facts to be edited, while the y-axis represents the corresponding EE.

6.3 RQ2a: The Impact of the Number of Edited Facts per Document on the Results.

We track the number of editing facts involved in each document, and figure 5 shows our experimental results. We found that the performance of all methods declines as the number of editing facts increases. When the number of editing facts is small, RAG-based methods perform better. However, as the number of editing facts grows, their performance rapidly deteriorates. When the number of editing facts reaches 5 or more, the FT method outperforms the RAG-based methods.

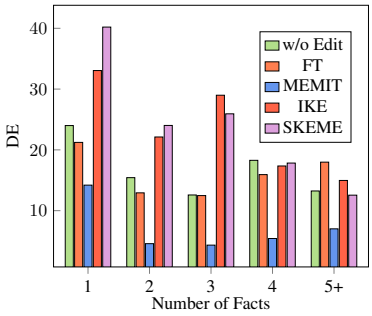

Figure 5: result of RQ2a. The x-axis represents the number of edits corresponding to each document, while the y-axis represents its DE.

6.4 RQ2b: The Impact of Fact Updates on Model Editing Performance.

To test the impact of fact updates on document-level model editing, we followed the same process as in DocMEdit. We collected updates from Wikipedia between 20220420 and 20231101 and merged them with the updates in DocMEdit based on the document. As a result, we obtained a series of document updates between 20220420 and 20241101, along with their corresponding facts to be edited. Since these facts to be edited were derived from two separate updates, there are contradictions between them. We used this data construction method to simulate real-world fact updates in the model.

Method	Accuracy			
	DR↑	DE↑	ER↑	EE↑
w/o Edit	**24.86**$_{-1.25}$	17.83$_{-1.14}$	16.83$_{+1.06}$	0.47$_{-0.03}$
FT	22.61$_{-2.17}$	13.19$_{-4.76}$	9.75$_{-4.90}$	6.22$_{-0.95}$
MEMIT	16.14$_{-3.49}$	8.70$_{-0.92}$	12.15$_{-3.01}$	2.48$_{-0.02}$
IKE	18.52$_{-1.27}$	22.13$_{-4.17}$	19.88$_{-2.89}$	10.83$_{-1.37}$
SKEME	19.97$_{-1.11}$	**27.45**$_{-1.89}$	**22.16**$_{-3.59}$	**19.59**$_{-4.33}$

Table 4: Result of RQ2b. The numbers in the subscript represent the difference between the results and the main experiment results.

Table 4 presents the experimental results. The analysis shows that almost all methods experienced a decline in performance when dealing with fact updates. Specifically, since the input document was switched from the 20231101 version to the corresponding 20220420 document, the output quality of the unedited model also decreased. FT and MEMIT showed significant declines both at the document level and at the edit level. This decline is attributed to excessive continuous editing, which caused the internal parameters of the LLM to diverge too far from the initial state, leading to a drop in output quality (Gupta et al., 2024).

IKE and SKEME experience a noticeable decrease in edit-level accuracy. We speculate this is due to the fact conflicts and the overwhelming number of facts affect the context learning ability. Although SKEME was shown in its original paper to handle fact updates (Zeng et al., 2024), the experiments reveal that it struggles with unstructured, document-level fact updates as seen in DocMEdit.

6.5 Discussion of Analysis

In the aforementioned RQs, we confirmed that both longer inputs and facts influence the effectiveness of model editing. Moreover, multiple parallel facts and structured facts also degrade the performance of existing model editing methods. To address this issue, we argue that decomposing the overall editing task (Zhong et al., 2023; Fei et al., 2024), adjusting the prompt structure and the positioning of facts (Liu et al., 2024b), simultaneously attending to shallow and deep neurons as well as attention heads (Zhang et al., 2024e), and managing conflicts between internal and external knowledge within the model (Zhao et al., 2024) are all beneficial strategies.

7 Error Analysis

We conducted an error analysis to identify the challenges in document level model editing.

7.1 Error categorization

As shown in Table 5, there are four main types of errors: Hallucination, Unexpected Style Change, Ignoring Fact Update, and Misunderstanding Facts.

Hallucination The model incorrectly adds content to the document without factual support.

Unexpected Style Change The model incorrectly alters the narrative style of the document

Error Type	Document and Facts	Target and Generated	Ratio
Hallucination	Document: Ian David George (1953–2016) was... Facts: None	Ian David George (1953–2016) (also Tata Ian George or simply Tata) was... Ian David George (1953–2016) was...	78.4
Unexpected Style Change	Document: The middle class refers to a class of people in the middle of a social hierarchy, often defined by occupation, income, education, or social status... Facts: None	The middle class is a social class that traditionally is defined by occupational, income, educational, or social status level that is neither high nor low in a hierarchy.... The middle class refers to a class of people in the middle of a social hierarchy, often defined by occupation, income, education, or social status.	7.7
Ignoring Fact Update	Document: Ashaghy Gushchular is a village in the Shusha District of Azerbaijan... Facts: 'Yuxari Quscular: It was part of Shusha District with Malibeyli and Ashaghy Gushchular villages till 5 December 2023.'	Ashaghy Gushchular is a village in the Shusha District of Azerbaijan... Ashaghy Gushchular is a village in the Shusha District of Azerbaijan... It was part of the Shusha District with Malibeyli and Yuxari Quscular villages till 5 December 2023.	8.6
Misunderstanding Facts	Document: Aremark is a municipality in Viken county, Norway... Facts: Østfold have 17 (former 18) municipalities: # Aremark...	Østfold is a county and former municipality in Norway. Aremark is a municipality in Østfold county, Norway.	5.3

Table 5: The four main error types that occur in DocMEdit. Red represents the generated output, green represents the target output. The underscored tilde represents the missing/incorrect/extraneous parts. Some of the output has been truncated for clearer presentation. We only selected facts relevant to the displayed input. "None" indicates that there are no relevant supporting facts for the corresponding text. "Ratio" refers to the proportion of this error type among all erroneous outputs.

without factual support. Unlike hallucination, an Unexpected Style Change does not involve changes to facts or entities, but modifies the style of narration.

Ignoring Fact Update Despite the provided editing facts containing the necessary information for document updates, the model fails to make the corresponding changes.

Misunderstanding Facts The model matches facts with the document and attempts edits, but fails to produce the correct result, instead mistakenly altering unrelated content.

7.2 Detailed Analysis and Discussion

We manually annotated 100 output samples for each model-method combination to identify their corresponding error types. Table 5 presents the distribution of several common error categories. Among them, hallucination errors are the most frequent, which is consistent with the findings of our main experiments.

For potential solutions, we recommend using the following approaches to address the four types of errors. For Hallucinations, Wang et al. (2025) suggest focusing on attention weights during editing rather than the commonly used FFN. For Unexpected Style Change, the key is to ensure that the model modifies facts while remaining faithful to the original text (Yao et al., 2025). For Ignoring Fact Update, adjusting the placement of relevant facts can help the model focus on crucial contextual information (Liu et al., 2024b; Parasaram et al., 2024). For Misunderstanding Facts, enhancing the LLM's comprehension ability, particularly in long-context scenarios, is essential (An et al., 2024).

8 Conclusion

We introduce document level model editing, a model editing task that is more aligned with real-world applications. To advance research in this task, we present DocMEdit, a dataset focused on document-level model editing. Experiments conducted on this dataset show that existing methods struggle with document-level model editing. Further experiments indicate that the challenges of document level model editing stem from long contexts and the presence of multiple facts to edit within a single document, aspects that are overlooked by current methods. We hope that our research will propel the field of model editing forward and inspire further research in this area.

Limitation

Since DocMEdit focuses on document-level model editing, the inputs and outputs are relatively long, encompass numerous facts. This imposes substantial demands on the context length supported by LLMs and requires greater computational resources for processing.

Ethical Statement

This study uses publicly available documents from Wikipedia and complies with its licensing requirements. The data involved in this study only includes publicly disclosed information. The study adheres to ethical guidelines, ensuring that the use of Wikipedia data is solely for academic and non-commercial purposes, and strives to mitigate any potential biases or unfair statements in the data. All ethical considerations regarding the privacy and potential harms associated with data usage have been carefully addressed.

Acknowledgments

We thank all the anonymous reviewers for their insightful and valuable comments. This work is supported by the National Natural Science Foundation of China (Grant No. U21B2009, 62406015) and Beijing Institute of Technology Science and Technology Innovation Plan (Grant No. 23CX13027).

References

Oshin Agarwal and Ani Nenkova. 2022. Temporal effects on pre-trained models for language processing tasks. *Transactions of the Association for Computational Linguistics*, 10:904–921.

Shengnan An, Zexiong Ma, Zeqi Lin, Nanning Zheng, Jian-Guang Lou, and Weizhu Chen. 2024. Make your llm fully utilize the context. *Advances in Neural Information Processing Systems*, 37:62160–62188.

Jason Ansel, Edward Yang, Horace He, Natalia Gimelshein, Animesh Jain, Michael Voznesensky, Bin Bao, Peter Bell, David Berard, Evgeni Burovski, Geeta Chauhan, Anjali Chourdia, Will Constable, Alban Desmaison, Zachary DeVito, Elias Ellison, Will Feng, Jiong Gong, Michael Gschwind, Brian Hirsh, Sherlock Huang, Kshiteej Kalambarkar, Laurent Kirsch, Michael Lazos, Mario Lezcano, Yanbo Liang, Jason Liang, Yinghai Lu, CK Luk, Bert Maher, Yunjie Pan, Christian Puhrsch, Matthias Reso, Mark Saroufim, Marcos Yukio Siraichi, Helen Suk, Michael Suo, Phil Tillet, Eikan Wang, Xiaodong Wang, William Wen, Shunting Zhang, Xu Zhao, Keren Zhou, Richard Zou, Ajit Mathews, Gregory Chanan, Peng Wu, and Soumith Chintala. 2024. PyTorch 2: Faster Machine Learning Through Dynamic Python Bytecode Transformation and Graph Compilation. In *29th ACM International Conference on Architectural Support for Programming Languages and Operating Systems, Volume 2 (ASPLOS '24)*. ACM.

Yingfa Chen, Zhengyan Zhang, Xu Han, Chaojun Xiao, Zhiyuan Liu, Chen Chen, Kuai Li, Tao Yang, and Maosong Sun. 2024. Robust and scalable model editing for large language models. In *Proceedings of the 2024 Joint International Conference on Computational Linguistics, Language Resources and Evaluation (LREC-COLING 2024)*, pages 14157–14172, Torino, Italia. ELRA and ICCL.

Johannes Daxenberger and Iryna Gurevych. 2012. A corpus-based study of edit categories in featured and non-featured Wikipedia articles. In *Proceedings of COLING 2012*, pages 711–726, Mumbai, India. The COLING 2012 Organizing Committee.

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.

Weizhi Fei, Xueyan Niu, Guoqing Xie, Yanhua Zhang, Bo Bai, Lei Deng, and Wei Han. 2024. Retrieval meets reasoning: Dynamic in-context editing for long-text understanding. *arXiv preprint arXiv:2406.12331*.

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

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

Jia-Chen Gu, Hao-Xiang Xu, Jun-Yu Ma, Pan Lu, Zhen-Hua Ling, Kai-Wei Chang, and Nanyun Peng. 2024. Model editing harms general abilities of large language models: Regularization to the rescue. In *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing*, pages 16801–16819, Miami, Florida, USA. Association for Computational Linguistics.

Akshat Gupta, Sidharth Baskaran, and Gopala Anumanchipalli. 2024. Rebuilding ROME : Resolving model collapse during sequential model editing. In *Proceedings of the 2024 Conference on Empirical*

Methods in Natural Language Processing, pages 21738–21744, Miami, Florida, USA. Association for Computational Linguistics.

Robert Iv, Alexandre Passos, Sameer Singh, and Ming-Wei Chang. 2022. FRUIT: Faithfully reflecting updated information in text. In *Proceedings of the 2022 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 3670–3686, Seattle, United States. Association for Computational Linguistics.

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.

Omer Levy, Minjoon Seo, Eunsol Choi, and Luke Zettlemoyer. 2017a. 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.

Omer Levy, Minjoon Seo, Eunsol Choi, and Luke Zettlemoyer. 2017b. 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.

Aixin Liu, Bei Feng, Bing Xue, Bingxuan Wang, Bochao Wu, Chengda Lu, Chenggang Zhao, Chengqi Deng, Chenyu Zhang, Chong Ruan, et al. 2024a. Deepseek-v3 technical report. *arXiv preprint arXiv:2412.19437*.

Nelson F Liu, Kevin Lin, John Hewitt, Ashwin Paranjape, Michele Bevilacqua, Fabio Petroni, and Percy Liang. 2024b. Lost in the middle: How language models use long contexts. *Transactions of the Association for Computational Linguistics*, 12:157–173.

Zeming Liu, Haifeng Wang, Zheng-Yu Niu, Hua Wu, Wanxiang Che, and Ting Liu. 2020. Towards conversational recommendation over multi-type dialogs. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 1036–1049, Online. Association for Computational Linguistics.

Zeming Liu, Ding Zhou, Hao Liu, Haifeng Wang, Zheng-Yu Niu, Hua Wu, Wanxiang Che, Ting Liu, and Hui Xiong. 2023. Graph-grounded goal planning for conversational recommendation. *IEEE Transactions on Knowledge and Data Engineering*, 35(5):4923–4939.

Potsawee Manakul, Adian Liusie, and Mark Gales. 2023. SelfCheckGPT: Zero-resource black-box hallucination detection for generative large language models. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pages 9004–9017, Singapore. Association for Computational Linguistics.

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

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

Eric Mitchell, Charles Lin, Antoine Bosselut, Christopher D Manning, and Chelsea Finn. 2022. Memory-based model editing at scale. In *International Conference on Machine Learning*, pages 15817–15831. PMLR.

Nikhil Parasaram, Huijie Yan, Boyu Yang, Zineb Flahy, Abriele Qudsi, Damian Ziaber, Earl Barr, and Sergey Mechtaev. 2024. The fact selection problem in llm-based program repair.

Zafaryab Rasool, Stefanus Kurniawan, Sherwin Balugo, Scott Barnett, Rajesh Vasa, Courtney Chesser, Benjamin M Hampstead, Sylvie Belleville, Kon Mouzakis, and Alex Bahar-Fuchs. 2024. Evaluating llms on document-based qa: Exact answer selection and numerical extraction using cogtale dataset. *Natural Language Processing Journal*, 8:100083.

Michael Schmitz, Stephen Soderland, Robert Bart, Oren Etzioni, et al. 2012. Open language learning for information extraction. In *Proceedings of the 2012 joint conference on empirical methods in natural language processing and computational natural language learning*, pages 523–534.

Anton Sinitsin, Vsevolod Plokhotnyuk, Dmitry Pyrkin, Sergei Popov, and Artem Babenko. 2020. Editable neural networks. In *International Conference on Learning Representations*.

Irene Solaiman, Miles Brundage, Jack Clark, Amanda Askell, Ariel Herbert-Voss, Jeff Wu, Alec Radford, and Jasmine Wang. 2019. Release strategies and the social impacts of language models. *ArXiv*, abs/1908.09203.

Hugo Touvron, Thibaut Lavril, Gautier Izacard, Xavier Martinet, Marie-Anne Lachaux, Timothée Lacroix, Baptiste Rozière, Naman Goyal, Eric Hambro, Faisal Azhar, Aurelien Rodriguez, Armand Joulin, Edouard Grave, and Guillaume Lample. 2023a. Llama: Open and efficient foundation language models.

Hugo Touvron, Louis Martin, Kevin Stone, Peter Albert, Amjad Almahairi, Yasmine Babaei, Nikolay Bashlykov, Soumya Batra, Prajjwal Bhargava, Shruti Bhosale, et al. 2023b. Llama 2: Open foundation and fine-tuned chat models. *arXiv preprint arXiv:2307.09288*.

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

Ke Wang, Jiahui Zhu, Minjie Ren, Zeming Liu, Shiwei Li, Zongye Zhang, Chenkai Zhang, Xiaoyu Wu, Qiqi Zhan, Qingjie Liu, et al. 2024a. A survey on data synthesis and augmentation for large language models. *arXiv preprint arXiv:2410.12896*.

Longyue Wang, Chenyang Lyu, Tianbo Ji, Zhirui Zhang, Dian Yu, Shuming Shi, and Zhaopeng Tu. 2023. Document-level machine translation with large language models. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pages 16646–16661, Singapore. 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. 2024b. EasyEdit: An easy-to-use knowledge editing framework for large language models. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 3: System Demonstrations)*, pages 82–93, Bangkok, Thailand. Association for Computational Linguistics.

Song Wang, Yaochen Zhu, Haochen Liu, Zaiyi Zheng, Chen Chen, and Jundong Li. 2024c. Knowledge editing for large language models: A survey. *ACM Computing Surveys*, 57(3):1–37.

Yu Wang, Jiaxin Zhang, Xiang Gao, Wendi Cui, Peng Li, and Kamalika Das. 2025. Gradient-guided attention map editing: Towards efficient contextual hallucination mitigation. *arXiv preprint arXiv:2503.08963*.

Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Rémi Louf, Morgan Funtowicz, Joe Davison, Sam Shleifer, Patrick von Platen, Clara Ma, Yacine Jernite, Julien Plu, Canwen Xu, Teven Le Scao, Sylvain Gugger, Mariama Drame, Quentin Lhoest, and Alexander M. Rush. 2020. Transformers: State-of-the-art natural language processing. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pages 38–45, Online. Association for Computational Linguistics.

Xiaobao Wu, Liangming Pan, William Yang Wang, and Luu Anh Tuan. 2024. Akew: Assessing knowledge editing in the wild. In *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing*, pages 15118–15133.

Jiakuan Xie, Pengfei Cao, Yuheng Chen, Yubo Chen, Kang Liu, and Jun Zhao. 2024. Memla: Enhancing multilingual knowledge editing with neuron-masked low-rank adaptation.

Lilong Xue, Dan Zhang, Yuxiao Dong, and Jie Tang. 2024. Autore: Document-level relation extraction with large language models. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 3: System Demonstrations)*, pages 211–220.

Jiashu Yao, Heyan Huang, Zeming Liu, Haoyu Wen, Wei Su, Boao Qian, and Yuhang Guo. 2025. Reff: Reinforcing format faithfulness in language models across varied tasks. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 39, pages 25660–25668.

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.

Li Zeng, Yingyu Shan, Zeming Liu, Jiashu Yao, and Yuhang Guo. 2024. FAME: Towards factual multitask model editing. In *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing*, pages 15992–16011, Miami, Florida, USA. Association for Computational Linguistics.

Mengqi Zhang, Xiaotian Ye, Qiang Liu, Pengjie Ren, Shu Wu, and Zhumin Chen. 2024a. Knowledge graph enhanced large language model editing. In *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing*, pages 22647–22662, Miami, Florida, USA. Association for Computational Linguistics.

Ningyu Zhang, Zekun Xi, Yujie Luo, Peng Wang, Bozhong Tian, Yunzhi Yao, Jintian Zhang, Shumin Deng, Mengshu Sun, Lei Liang, Zhiqiang Zhang, Xiaowei Zhu, Jun Zhou, and Huajun Chen. 2024b. Oneedit: A neural-symbolic collaboratively knowledge editing system.

Ningyu Zhang, Yunzhi Yao, and Shumin Deng. 2024c. Knowledge editing for large language models. In *Proceedings of the 2024 Joint International Conference on Computational Linguistics, Language Resources and Evaluation (LREC-COLING 2024): Tutorial Summaries*, pages 33–41, Torino, Italia. ELRA and ICCL.

Ningyu Zhang, Yunzhi Yao, Bo Tian, Peng Wang, Shumin Deng, Meng 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. 2024d. A comprehensive study of knowledge editing for large language models. *ArXiv*, abs/2401.01286.

Zhuoran Zhang, Yongxiang Li, Zijian Kan, Keyuan Cheng, Lijie Hu, and Di Wang. 2024e. Locate-then-edit for multi-hop factual recall under knowledge editing. *arXiv preprint arXiv:2410.06331*.

Yu Zhao, Alessio Devoto, Giwon Hong, Xiaotang Du, Aryo Pradipta Gema, Hongru Wang, Xuanli He,

Kam-Fai Wong, and Pasquale Minervini. 2024. Steering knowledge selection behaviours in llms via sae-based representation engineering. *arXiv preprint arXiv:2410.15999*.

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

Hanwen Zheng, Sijia Wang, and Lifu Huang. 2024. A comprehensive survey on document-level information extraction. In *Proceedings of the Workshop on the Future of Event Detection (FuturED)*, pages 58–72.

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 Terminology Explanation

In this section, we will explain some of the terms we used and their calculation methods.

Target Len Target Len is used to describe the content expected to be generated by the LLM. For question-answering or sentence completion datasets (Zsre, CounterFact, MQUaKE, FAME), it refers to the expected generated phrases. For Wikibio, it refers to the content generated after the original document. For DocMEdit, it refers to the updated document.

Facts Len Facts Len refers to the length of the facts to be edited. For question answering or sentence completion datasets, it refers to the concatenation of all questions and answers. For Wikibio, it refers to the content expected to be generated by the model. For DocMEdit, it refers to the facts to be edited for each document.

Context Len Context Len refers to the context length that the model needs to process throughout the entire task, which is the length of the concatenated input and output.

B Dataset Samples and Detail Statistics

B.1 Dataset Samples

Figure 6 gives an example from DocMEdit, where "Input" and "Target" correspond to the document before and after the update, respectively. "Facts" represents the facts to be edited, "Title" denotes the title of the document, which is also used as the subject during the update process, "Inputs_sro", "Targets_sro", and "Facts_sro" represent the triples extracted from the input, output, and facts, respectively.

B.2 Data distribution

Table 7 presents the distribution of the length of facts and contexts used for each data item in DocMEdit. Figure 6 shows the distribution of context length. Figure 7 shows the distribution of fact length.

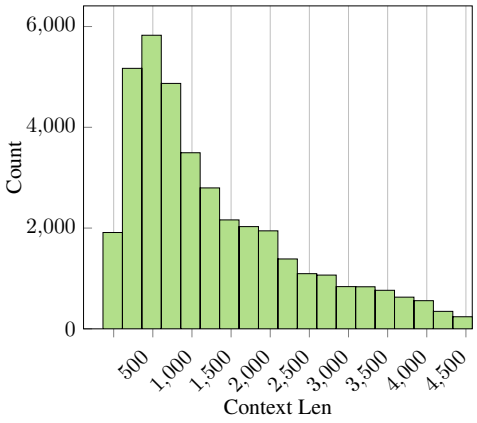

Figure 6: The distribution of Context length.

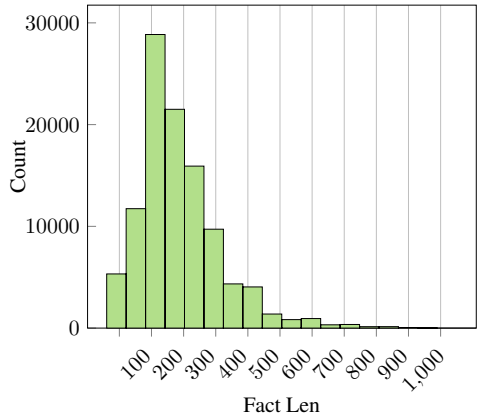

Figure 7: The distribution of fact length.

C Experimental Details

C.1 Experimental Environment

All experiments were conducted on 2 $*$ NVIDIA A100-PCIE-40GB GPUs. The main software environment we used includes CUDA version 11.4, PyTorch version 2.0.1 (Ansel et al., 2024), and Transformers library version 4.45.2 (Wolf et al., 2020).

Data Example

Title: Asagi Quscular

Inputs: Ashaghy Gushchular () or Ghushchular () is a village in the Shusha District of Azerbaijan. Until 2023 it was controled by the self-proclaimed Republic of Artsakh. The village had an Azerbaijani-majority population before the First Nagorno-Karabakh War. During the capture of the village, the Azerbaijani population was expelled, and it was reported that 8 civilians were killed.

Targets: Ashaghy Gushchular () is a village in the Khojaly District of Azerbaijan. Until 2023 it was controlled by the self-proclaimed Republic of Artsakh. The village had an Azerbaijani-majority population before the First Nagorno-Karabakh War. During the capture of the village, the Azerbaijani population was expelled, and it was reported that 8 civilians were killed. It was part of Shusha District with Malibeyli and Yuxari Quscular villages till 5 December 2023.

Facts: ['Yuxari Quscular INTRODUCTION It was part of Shusha District with Malibeyli and Ashaghy Gushchular villages till 5 December 2023.', 'Malibeyli INTRODUCTION It was part of Shusha District with Asagi Quscular and Yuxari Quscular villages till 5 December 2023.']

Inputs_sro: [['Ashaghy Gushchular', 'instance of', 'village'], ['Shusha District', 'country', 'Azerbaijan'], ['Ashaghy Gushchular', 'ethnic group', 'Azerbaijani']]

Targets_sro: [['Ashaghy Gushchular', 'instance of', 'village'], ['Shusha District', 'country', 'Azerbaijan'], ['Ashaghy Gushchular', 'ethnic group', 'Azerbaijani'], ['Malibeyli', 'part of', 'Shusha District'], ['Yuxari Quscular', 'part of', 'Shusha District']]

Facts_sro: [['Malibeyli', 'part of', 'Shusha District'], ['Yuxari Quscular', 'part of', 'Shusha District']]

Table 6: An example of DocMEdit. In the input and target, red represents the parts deleted in the target, yellow represents the parts that have changed in the target, and green represents the newly added parts in the target.

Facts Per Data Item				
1	2	3	4	5+
18002	7233	3451	2078	7226
Context Len				
0-512	512-2048	2048-4096	4096+	
4572	23283	8981	1154	

Table 7: Distribution of the Length of Facts and Contexts.

C.2 Data

We used all the data from DocMEdit in our experiments. For each document edit, in the parameter modification methods (FT, MEMIT), we utilized all the editable facts, while in the RAG-based methods (IKE, SKEME, EREN), we used the top 5 retrieved facts to prevent exceeding the model's context length limit.

C.3 Input Format

To ensure that the large language model can correctly perform the document-level model editing task, we controlled the output format of the model. Specifically, we provided the original document to the model and instructed it to make updates. For all models, we used 1-shot learning to control the format while avoiding exceeding the model's context length limit. For RAG-based models (IKE, SKEME, EREN), we also prompted them to use the provided facts following previous research (Zeng et al., 2024; Zheng et al., 2023). For EREN, we slightly modified the prompt of the original document to better align with our task. The complete list of prompts can be found in Appendix C.7.

C.4 Metrics Definition

In this section, we formally define the calculation methods for our metrics.

First, let the original document be denoted as y, the target document as y', and the facts to be edited as $F = \{f_1, f_2, \ldots, f_n\}$. The model's actual output is denoted as y''. Define the operation $\text{Update}(a, b)$ as the computation of the difference between a and b, and $\text{Res}(a, b)$ as the part of a and b excluding the updated portions, i.e., the unchanged parts, $E(x)$ as the extraction of triples from x, and $\text{ROUGE}(a, b)$ as the calculation of the ROUGE score between a and b.

DR DR (Document ROUGE) represents the document update metric computed using ROUGE. Following Iv et al. (2022), we actually consider only the updated sentences rather than the full texts, i.e.,

$$DR = \text{ROUGE}(\text{Update}(y, y''), \text{Update}(y, y'))$$

DE DE (Document Entity) is a document update metric calculated using the entity. Compared to ROUGE, it focuses more on the actual factual updates. We similarly compute only the updated entities, i.e.,

$$\frac{\text{Update}(E(y), E(y'')) \cap \text{Update}(E(y), E(y')))}{\text{len}(\text{Update}(E(y), E(y')))}$$

ER ER (Edit ROUGE) represents the ROUGE-based metric for triple updates. It is a finer-grained evaluation metric designed to assess whether each fact has been successfully updated. It is calculated as:

$$ER = \text{ROUGE}(\text{Update}(y'', y), f_i)$$

EE EE (Edit Entity) is the edit success rate calculated using triples. It is similarly used to evaluate whether each fact has been successfully updated. We calculate the proportion of updates in y'' relative to y that are supported by fact f_i in the updates of y' relative to y, i.e.,

$$A = \text{Update}(E(y), E(y''))$$
$$B_i = E(f_i) \cap \text{Update}(E(y), E(y'))$$
$$EE = \frac{A \cap B_i}{\text{len}(B_i)}$$

RSE ROUGE Side Effect (RSE) is used to calculate the ROUGE score for the correctly retained parts of the document. It is computed based on the unmodified portions in the model's output and the expected output. Specifically, it is calculated as:

$$RSE = \text{ROUGE}(\text{Res}(y, y''), \text{Res}(y, y'))$$

ESE ESE (Entity Side Effect) is used to calculate the correctly retained entities in the document, which refers to the entities that remain unchanged in the model's output, similar to how ROUGE Side Effect (RSE) deals with the unmodified text. It is calculated as:

$$\frac{\text{Res}(E(y), E(y'')) \cap \text{Res}(E(y), E(y')))}{\text{len}(\text{Res}(E(y), E(y')))}$$

Method	Precision	Recall	F1 Score
IKE	47.20	89.71	53.45
EREN	47.60	89.86	53.77
SKEME	100.00	93.03	95.55

Table 8: Retrieval Result

C.5 Semantic Coherence Evaluation Criteria

Figure 9 shows the scoring criteria used to evaluate semantic coherence.

C.6 Implementation Details of Baselines

For FT, MEMIT, and IKE, we use the framework provided by Wang et al. (2024b)[2]. For EREN, we used the original implementation but modified the prompt to fit tasks[3]. For all methods, we use greedy decoding to obtain the LLM's output after editing and then perform evaluation.

FT Following previous research (Meng et al., 2023), We apply Fine-Tuning (FT) to the given layer of the model. For GPT2-XL, we select layer 0, and for GPT-J and Llama2, we choose layer 21.

MEMIT For GPT2-XL and GPT-J, we employ default hyperparameters. For Llama2, we update the parameters of layers $\{4, 5, 6, 7, 8\}$. Across all models, we calculate covariance statistics using 50,000 instances from Wikitext. We use the document title as the subject of the edited facts.

IKE We use the retrieval model settings from the original paper[4], retrieving the top 5 facts to prevent the output length from exceeding the LLM's context length limit.

EREN We adopt the retrieval model setup from the original paper[5], retrieving the top 5 facts to prevent the output length from exceeding the LLM limit. We slightly modified the prompt to adapt it to our task, which can be found in Section C.7.

SKEME Due to the length of the document context, we use the title of the document as the subject in place of the subject generated by the LLM in the original paper to improve accuracy. We retain the top 5 facts to prevent the output length from exceeding the LLM limit.

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

[3] https://github.com/thunlp/EREN

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

[5] https://huggingface.co/facebook/contriever

C.7 Prompts Used

In this section, we present the prompts used for various methods. Figure 8 shows the prompts used for the base model, FT, and MEMIT. Figure 9 displays the prompts used for IKE and SKEME, while Figure 10 presents the prompts used for EREN.

D Retrieval Result

For all RAG models, we follow the EREN (Chen et al., 2024) setup and use topk $= 5$ to ensure a fair comparison.

Table 8 shows the results of data retrieval for several RAG methods. It can be observed that IKE and EREN, which retrieve data via vector databases, have poorer retrieval results, while SKEME, which retrieves facts through entity-based searches, achieves better results.

E Relations

Table 10 presents the top 40 relations by frequency, extracted from the triples in DocMEdit.

F Running Example

Table 11 presents a running example obtained on Llama2.

Scoring Criteria
0: The text is severely incoherent, with disjointed or contradictory ideas that lack logical flow. Sentences may be unrelated or nonsensical. 1: The text has partial coherence but contains noticeable inconsistencies, abrupt topic shifts, or repetitive/redundant statements that disrupt understanding. 2: The text is fully coherent, with clear logical progression, consistent ideas, and smooth transitions between sentences.

Table 9: Evaluation criteria for semantic coherence.

```
Document: <few-shot input>
Updated version: <few-shot output>

Document: <input>
Updated version:
```

Figure 8: Prompt for w/o Edit, FT and MEMIT. "Few-shot input" and "few-shot output" refer to the input and output of few-shot samples. The "input" contains the actual document to be updated.

```
Document: <few-shot input>
Facts: <few-shot facts>
Updated version: <few-shot output>

Document: <input>
Facts: <retrieved facts>
Updated version:
```

Figure 9: Prompt for IKE and SKEME. "Few-shot input" refers to the input of a few-shot samples. "Few-shot facts" refers to the facts to be edited in few-shot samples, which are directly obtained from the dataset. "Few-shot output" refers to the output of few-shot samples. The "input" contains the actual document to be updated, while the "retrieved facts" are the facts obtained through the retrieval method.

```
Read facts and update the document. If the document is unupdatable,
   say 'unupdatable'.

Document: <few-shot input>
Facts: <few-shot facts>
Updated version: <few-shot output>

Document: <input>
Facts: <retrieved facts>
Updated version:
```

Figure 10: Prompt for EREN. System instructions were added to the figure 9 to guide the model in following the task.

Wikidata Label	Wikidata ID	Relevant Triples	The proportion of all relations
of	P642	243694	17.27 %
including	P1012	31768	2.25 %
competition won	P2522	30580	2.17 %
replaced by	P1366	21881	1.55 %
award received	P166	19690	1.4 %
founded by	P112	18149	1.29 %
destroyed	P3082	16459	1.17 %
in operation on service	P10788	15441	1.09 %
name	P2561	15143	1.07 %
release of	P9831	15102	1.07 %
had as last meal	P3902	15020	1.06 %
position held	P39	14608	1.04 %
has cause	P828	14524	1.03 %
location	P276	13992	0.99 %
has tense	P3103	13573	0.96 %
has use	P366	13138	0.93 %
start point	P1427	13059	0.93 %
uses	P2283	12609	0.89 %
followed by	P156	10576	0.75 %
lighting	P8228	9286	0.66 %
acknowledged	P7137	8804	0.62 %
maintained by	P126	8617	0.61 %
contains	P4330	8376	0.59 %
time played	P9140	7862	0.56 %
develops from	P3094	7760	0.55 %
moved by	P6939	7702	0.55 %
named after	P138	7688	0.54 %
place of birth	P19	7542	0.53 %
location of formation	P740	7334	0.52 %
points for	P1358	7032	0.5 %
follows	P155	6919	0.49 %
made from material	P186	6852	0.49 %
for work	P1686	6716	0.48 %
dedicated to	P825	6552	0.46 %
produced by	P2849	6502	0.46 %
merged into	P7888	6489	0.46 %
towards	P5051	6485	0.46 %
damaged	P3081	6384	0.45 %
based on	P144	6271	0.44 %
winner	P1346	6227	0.44 %

Table 10: Statistics of relations. Wikidata Label refers to the name of the relation in Wikidata, while Wikidata ID refers to the ID assigned to it by Wikidata.

Data	Input	The Belle Bay Formation is a metamorphosed formation cropping out in Newfoundland.
	Target	The Belle Bay Formation is a metamorphosed formation cropping out in Newfoundland. It is locally overlain by of the Andersons Cove Formation.
	Facts	Andersons_Cove_Formation INTRODUCTION The Andersons Cove Formation is a formation which overlies the Belle Bay Formation in Newfoundland, Canada.
Output	w/o Edit	The Belle Bay Formation is a metamorphosed formation cropping out in Newfoundland. It is exposed in high-grade gneiss terranes east of Belle Bay. It is metamorphosed at a contact with the Bonavista formation. It crops out in a similar high-pressure setting in eastern Quebec (O'Hare, 1987, Geological Survey of Canada, p. 42) and near Gogama, Ontario (Lorna, 1979; Ridgway, 1985).
	FT	The Belle Bay Formation is a metamorphosed formation cropping out in Newfoundland, Canada. It is a member of the North America-wide group of rock formations known as the Taconian Orogeny. The Belle Bay Formation is over 800 million years old.
	MEMIT	The Bay Formation, a late Devonian geologic formation, is exposed along Southeastern Newfoundland between Bay Bulls, Bay de Verde, L'Anse aux Meadows, and Point Amour on the Northern Peninsula.
	IKE	The Belle Bay Formation is a metamorphosed formation cropping out in Newfoundland, Canada. It is overlain by the Andersons Cove Formation.
	SKEME	The Belle Bay Formation is a metamorphosed formation cropping out in Newfoundland, Canada. It is overlain by the Andersons Cove Formation.

Table 11: The running example obtained on Llama2, where "data" refers to the input data, and "output" refers to the results generated by each method. Note that, since this data item is relatively simple, the facts retrieved by IKE and SKEME are the same, so the outputs do not differ.