History Matters: Temporal Knowledge Editing in Large Language Model

Xunjian Yin[1,2], **Jin Jiang**[1], **Liming Yang**[3], **Xiaojun Wan**[1,2]
[1]Wangxuan Institute of Computer Technology, Peking University
[2]Center for Data Science, Peking University
[3]School of Law, Tsinghua University
{xjyin, wanxiaojun}@pku.edu.cn, jiangjin@stu.pku.edu.cn, yanglm23@mails.tsinghua.edu.cn

Abstract

The imperative task of revising or updating the knowledge stored within large language models arises from two distinct sources: intrinsic errors inherent in the model which should be corrected and outdated knowledge due to external shifts in the real world which should be updated. Prevailing efforts in model editing conflate these two distinct categories of edits arising from distinct reasons and directly modify the original knowledge in models into new knowledge. However, we argue that preserving the model's original knowledge remains pertinent. Specifically, if a model's knowledge becomes outdated due to evolving worldly dynamics, it should retain recollection of the historical knowledge while integrating the newfound knowledge. In this work, we introduce the task of **Temporal Knowledge Editing** (**TKE**) and establish a benchmark ATOKE (**A**ssessment of **T**emp**O**ral **K**nowledge **E**diting) to evaluate current model editing methods. We find that while existing model editing methods are effective at making models remember new knowledge, the edited model catastrophically forgets historical knowledge. To address this gap, we propose a simple and general framework termed **M**ulti-**E**diting with **T**ime **O**bjective (**METO**) for enhancing existing editing models, which edits both historical and new knowledge concurrently and optimizes the model's prediction for the time of each fact. Our assessments demonstrate that while ATOKE is still difficult, METO maintains the effectiveness of learning new knowledge and meanwhile substantially improves the performance of edited models on utilizing historical knowledge.

Introduction

Large-scale language models (LLMs) have made impressive progress in the last few years (Brown et al. 2020; Ouyang et al. 2022; Touvron et al. 2023; OpenAI 2023). However, the knowledge in a language model always needs to be updated because the internal knowledge of the model can be wrong and the external world knowledge is constantly changing (Sinitsin et al. 2020; Hartvigsen et al. 2022; Ji et al. 2023). It would be costly to retrain the model each time, so there is a lot of work proposing knowledge editing (KE) methods that allow new correct knowledge to be injected directly into specific model parameters. Previous work includes constrained fine-tuning (Zhu et al. 2020), hypernetwork knowledge editing (Cao, Aziz, and Titov 2021; Hase et al. 2021; Mitchell et al. 2022a), external memory-based editing (Mitchell et al. 2022b; Zhong et al. 2023; Zheng et al. 2023) and locate-then-edit model editing (Dai et al. 2022; Meng et al. 2022b). All of these methodologies focus on making the model memorize new knowledge.

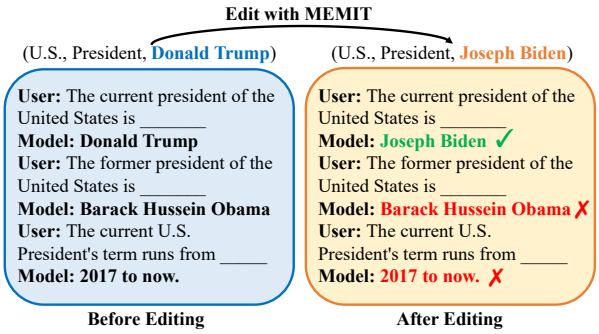

Figure 1: An example of an edited GPT-J model losing historical knowledge by using the existing editing method MEMIT. The editing operation overwrites *Joseph Biden being the President of the U.S.* against *Donald Trump*, but does not preserve historical knowledge, which causes Trump's relationship with the U.S. President to be lost.

However, a clear differentiation is essential during the process of knowledge editing, specifically discerning between two distinct scenarios: (1) **knowledge correction**, involving rectification of inaccurate knowledge arising from the model's training data which needs to be corrected, and (2) **knowledge updating**, necessitated by shifts in the external world or evolving cognitive paradigms which needs to be updated. Existing work on KE lacks this crucial distinction, and is committed to making the LLMs memorize the new knowledge, while simply ignoring the original knowledge in LLMs. However, within the context of knowledge updating, we believe that retaining **historical knowledge** within the model holds great value. For example, as shown in Figure 1, although we are all "updated" with the knowledge that *President of the United States is Joseph Biden*, we still sometimes want to know who *the former President* was.

The current evaluation of KE methods focuses on new knowledge being memorized and irrelevant knowledge be-

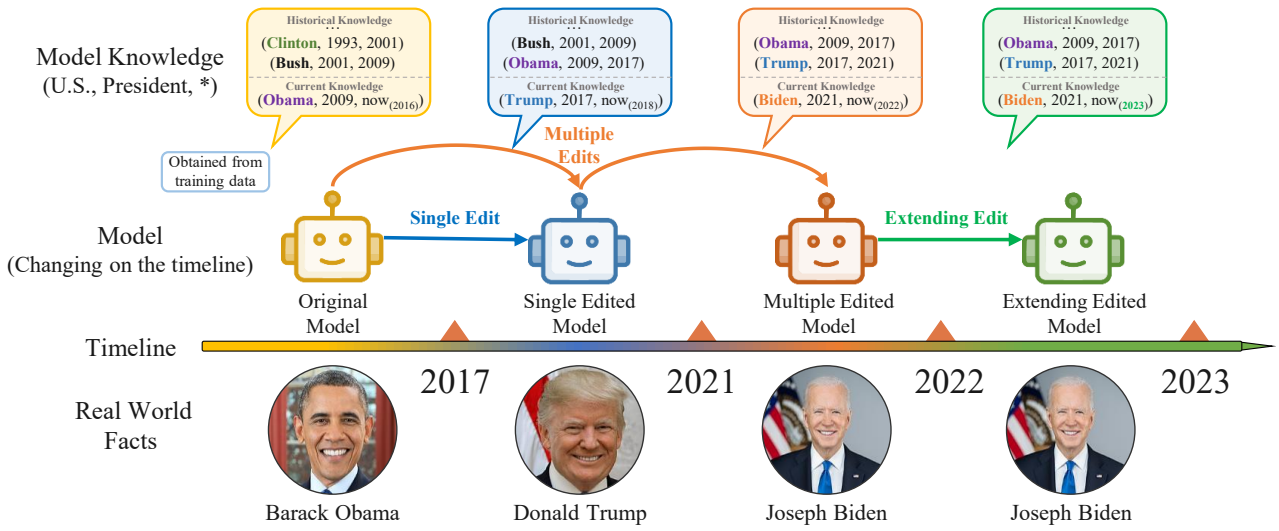

Figure 2: Presentation of Temporal Knowledge Editing Task. Suppose that the training corpus for the model is collected in 2016, so the internal time of the model is 2016. After one edit the model can obtain the knowledge that *Donald Trump is the president of the United States in 2017 and Obama is the former president*. Then keep editing in this way, the model's internal time keeps moving forward while not losing history and ensuring temporal ordering.

ing left unaltered, while not considering the appropriate keeping of relevant historical knowledge. Therefore, we propose the task of Temporal Knowledge Editing (TKE) and construct a benchmark **ATOKE** for evaluating KE methods in historical knowledge preservation, by collecting series of world knowledge with timestamps and viewing them as the form of a series of knowledge updates. We explore single editing, multiple editing and extending editing (e.g., letting the model know that *the president is still Biden in 2023* in Figure 2) of facts on the time series. After each editing, we ask questions about historical knowledge and current knowledge.

We evaluate state-of-the-art knowledge-editing methods on ATOKE and find that they are effective in making the model memorize new knowledge but cause confusion in time, e.g., in our example, the knowledge about the former president disappeared in edited LLMs.

In order to address this problem to some extent and to enhance the effectiveness of current model editing methods, we propose a simple and general framework, METO, which edits both historical and new knowledge and optimizes the model's prediction for the time of each fact.

Our contributions are listed below:

1. We categorize knowledge editing scenarios, point out the issue that current methods corrupt historical knowledge, and validate it through experiments and analysis.
2. We are the first to propose the Temporal Knowledge Editing task, and publish a benchmark ATOKE for evaluating the task.
3. We propose a new editing framework METO that improves the performance of previous editing methods on the benchmark.

Our benchmark has been released to the community to facilitate future research [1].

Temporal Knowledge Editing

This section introduces our arguments about knowledge editing as well as definitions and evaluation for the temporal knowledge editing task.

Background: Knowledge in Language Models

There has been a lot of work considering LLMs as knowledge bases and accessing, applying and manipulating the knowledge in them (Petroni et al. 2019a; AlKhamissi et al. 2022). Previous work always represents knowledge as a triple (s, r, o), consisting of a subject (s), a relation (r), and an object (o) (e.g. (*U.S.*, *president*, *Joseph Biden*)) and explores how to extract knowledge from LLMs (Bordes et al. 2013; Lin et al. 2015). In this paper, we follow the work that uses discrete prompt to extract knowledge (Petroni et al. 2019b; Davison, Feldman, and Rush 2019). Specifically, we construct a natural language template $q_r(\cdot)$ for each relation r, which is then combined with a subject s in a knowledge triple to generate a question or a cloze-style statement $q_r(s)$, which is also consistent with the previous work of the knowledge editing (Meng et al. 2022a; Zhong et al. 2023). It is a natural way to extract or test the knowledge in a language model.

Requirements for Knowledge Editing

What are the requirements that a good knowledge editing method needs to satisfy? We believe that the edited model should be changed in some ways and hold constant in others.

[1] https://github.com/Arvid-pku/ATOKE

Aspects of Change The knowledge we edit is naturally expected to change in the model, which is generally referred to as *Edit Success* in prior work in evaluation. In addition, the paraphrase expression about the knowledge should also be changed, which is referred to as *Paraphrase Success* in evaluation. Similarly, the knowledge associated with the edited knowledge should also be changed, which has been explored by Zhong et al. (2023) and referred to as *Multi-hop Accuracy*.

Aspects of Constancy Except for what is relevant to the edited knowledge, it seems like no other knowledge or ability of the model should be changed. Previous work (Meng et al. 2022b) has defined *Neighborhood Success* to evaluate whether other knowledge of the original entity is affected. They also define *Reference Score* to check if generations from the edited model are semantically consistent with the new object. And *Generation Entropy* is designed to test ability of the edited model to generate fluent sentences.

However, no work has yet explored the knowledge to be edited. We argue that if the knowledge within the model is outdated (changed to historical knowledge) due to the flow of time, then the historical knowledge should also be preserved within the model. It is quite reasonable because if not then the timeline in the model would be messed up and our *former president* would disappear in the aforementioned example. Therefore, we propose the task of Temporal Knowledge Editing (TKE)[2].

Task Definition

As mentioned in Section , we use triples to represent knowledge. In addition, we attach temporal scope to each relational fact $\{(s, r, o, t_s, t_u)\}$ since the relational fact often shows temporal dynamics, where t_s and tu denote the fact (s, r, o) valid from ts to tu. As time passes, the same subject and relation will correspond to different objects in sequence, such as (*U.S.*, *president*, *Donald Trump*, *2017*, *2021*), (*U.S.*, *president*, *Joseph Biden*, *2021*, N/A) and so on. Consistent with previous work, we use $(s, r, o, t_s, t_u) \rightarrow (s, r, o^\star, t_s^\star, t_u^\star)$ to represent an knowledge editing operation, denoted as e, meaning that the object of subject s under relation r changes from o to $o^\star$. And the reason for the edit is that (s, r, o) is valid from t_s until t_u, and after that $(s, r, o^\star)$ is valid from $t_s^\star$ to $t_u^\star$[3]. Specifically, o and $o^\star$ can be the same object, at which point the editing operation $(s, r, o, t_s, t_u) \rightarrow (s, r, o, t_s, t_u^\star)$ will make the model know that the fact (s, r, o) lasts until $t_u^\star$, where $t_u^\star$ is later than t_u. For convenience of expression, we refer to (s, r, o, t_s, t_u) as **historical knowledge** and $(s, r, o^\star, t_s^\star, t_u^\star)$ as **current knowledge**.

Given a collection of knowledge editing operations $\mathcal{E} = \{e_1, e_2, \ldots\}$ and a language model $\mathcal{M}$, knowledge editor F aims to get an edited model $\mathcal{M}_e = F(\mathcal{M}, \mathcal{E})$ that satisfy the requirements mentioned in Section . Specifically, in temporal knowledge editing, for the edited model $\mathcal{M}$, we expect that it should not only memorize the current knowledge $(s, r, o^\star)$, but the knowledge in it should also be organized on the timeline. In other words, the model can answer correctly when asked questions about the "previous" or "last one" knowledge (s, r, o), or when asked directly about the relevant knowledge $(s, r, ?)$ at a certain moment in time between t_s and t_u. For example, after updating the model's knowledge of U.S. president to Biden, the test questions include "*Who was the former president of the United States?*" or "*Who was the president of the United States in 2005?*" .

Evaluation of Temporal Knowledge Editing

Since time is constantly moving forward, a piece of knowledge may be updated multiple times in different temporal scopes. Therefore, we define three evaluation subtasks: 1) **Temporal Knowledge Single Editing (TSE)**, which requires that, after a single update to a piece of knowledge, the model sustains temporal chronology. 2) **Temporal Knowledge Multiple Editing (TME)**, demanding ordered temporal succession after multiple unidirectional knowledge updates. It is worth pointing out the difference between TME and mass-editing of prior work (Meng et al. 2022b; Mitchell et al. 2022a): mass-editing refers to the simultaneous editing of multiple facts, which are usually unrelated, while TME refers to the continuous editing of the same s and r at different time scopes. 3) **Temporal Knowledge Extending Editing (TEE)**, which requires that duration of target knowledge perceived by the model is successfully prolonged. More details on the benchmark are presented in Section .

ATOKE: Assessment of Temporal Knowledge Editing

We introduce the ATOKE (Assessment of temporal knowledge editing) benchmark, which comprises three distinct datasets as outlined in Table 1. These datasets are specifically designed to evaluate the effectiveness of knowledge editing methods in handling temporal information. The first dataset, referred to as ATOKE-SE, consists of temporal knowledge **S**ingle **E**dits. The second dataset, ATOKE-ME, includes temporal knowledge **M**ultiple **E**dits associated with the same subject and relation. The third dataset, ATOKE-EE, encompasses knowledge **E**dits that **E**xtend the temporal scope of the original knowledge. In the following sections, we first present the collection of the basic temporal knowledge and further describe how the three datasets are extracted from it. We then detail the data statistics and evaluation settings, and finally present the evaluation metrics.

Basic Temporal Knowledge Collection

Sampling time series facts Our dataset is based on YAGO3.0.3[4] (Mahdisoltani, Biega, and Suchanek 2015), a knowledge base comprising fact triples associated with millions of entities extracted from Wikipedia. Following the previous work (Dasgupta, Ray, and Talukdar 2018), we begin by sampling temporal facts from YAGO. We extract all fact triples with their time scope and obtain

[2]Temporal Knowledge Editing focuses on knowledge that is outdated due to the flow of time and therefore needs to be edited, rather than original errors within the model.

[3]Note that the t_u is vacant if the knowledge is valid currently.

[4]https://yago-knowledge.org/downloads/yago-3

BTK	(United States, head of government, (Obama, 2009, 2017), (Trump, 2017, 2021), (Biden, 2021, 2022))
SE	(United States, head of government, (Obama, 2009, 2017) → (Trump, 2017, 2021))
ME	(United States, head of government, (Obama, 2009, 2017) → (Trump, 2017, 2021) → (Biden, 2021, 2022))
EE	(United States, head of government, (Biden, 2021, 2022) → (Biden, 2021, 2023))
$\mathcal{Q}$	a) Who is the current President of the United States? (ALL) b) Who was the President of the United States in the previous term? (SE&ME) c) Who holds the position of President in the United States from 2017 to 2021? (SE&ME) d) Who was the President of the United States from 2009 to 2017? (SE&ME) e) From 2022 to 2023, who serves as the president of the United States? (EE)
$\mathcal{A}$	a&c): Donald Trump b&d): Barack Obama e) Joseph Biden

Table 1: An instance illustrating the construction of the dataset, where the collected base temporal knowledge (BTK) serves as the foundation. The dataset encompasses three types of edits, namely single edit (**SE**), multiple edits (**ME**), and extending edit (**EE**), along with corresponding questions (Q) and their respective answers (A) following the knowledge editing. In this case, the model has been edited from Obama to Trump.

the temporal facts set $\{(s, r, o, t_s, t_u)\}$. Subsequently, after matching by s and r and sorting by t_s with all these facts, we obtain the chain of temporal facts $\mathcal{C} = \{(s, r, o_1, t_{s_1}, t_{u_1}), ..., (s, r, o_N, t_{s_N}, t_{u_N})\}$. Finally, to ensure non-overlapping and non-contradictory time sequences within the chain, we employ a heuristic algorithm.

Locating Time of Model Facts For a chain of temporal facts $\{(s, r, o_N, t_{s_N}, t_{u_N})\}$, its o, t_s and t_u are constantly changing. Therefore, to test the performance of knowledge editing, we need to determine where the knowledge is located in the model. First, we use the GPT-J model[5] to verify and filter out $(s, r, *)$ that the model does not have. Next, we locate the model's knowledge position in the temporal fact chain. By asking questions about the facts in the temporal chain, we find the most recent fact in each chain that the model has successfully recalled as the model's initial knowledge.

Sampling Future Fake facts So far, we have collected a temporal chain of facts from the YAGO dataset where the latest knowledge is collected in 2022. To ensure that there is always new knowledge used to update for all models in our dataset and to make our dataset effective to be used as the benchmark for TKE in the long run, we design future fake facts to augment the original temporal chain. Specifically, for each chain, we sample one counterfactual object in YAGO to serve as counterfactual object o_{N+1} that is in the same class as real objects. Then o_{N+1} will be randomly assigned a reasonable time scope and appended to the end of the chain. Finally, we obtain a temporal chain of facts that incorporates constructed fake facts.

Generating Temporal natural language questions As mentioned in Section , given a chain of temporal facts that we construct, we construct template $q_r(\cdot)$ for each relation r. The constructed questions are categorized into two categories based on time: 1) **Explicit time question**, which refers to asking the question that have an explicit time frame. 2) **Relative time question**, on the other hand, does not contain an explicit time frame and uses words like "present", "previous" or "last one" instead. We first utilize ChatGPT (gpt-3.5-turbo) to automatically generate questions for all relations, and then manually filter them.

Construction of Three Datasets

Up until now, we have obtained the basic temporal knowledge, including temporal fact chains, questions and answers at different time points (respectively BTK, Q and A in Table 1). And after locating and expanding, the first fact of every temporal fact chain represents the current knowledge of the model. Therefore, all of our edits begin with the first fact. Moving forward, we will construct three temporal knowledge editing datasets.

ATOKE-SE For Single Edit (SE), we take the first two facts from each chain in BTK to construct a single edit operation, and ask questions about both historical and current facts.

ATOKE-ME Multiple Edits (ME) are superimposed on single edits. We similarly traverse all temporal chains, construct successive editing operations for all facts in each chain, and ask questions about facts with each time scope.

ATOKE-EE Extending Edit (EE) is an edit operation that extends the time scope of the current fact of the model. We traverse all chains of temporal facts and select the first fact in each chain as the extending object. By manually extending this object, we keep the target entity o unchanged but extend the time scope. This edit operation simulates the situation where certain facts are not changed in the future.

Dataset Summary

Dataset format The three datasets have a similar data format, but they differ in the number of edits and the questions used to evaluation. These datasets all contain edit sets (**SE&ME&EE**), questions ($\mathcal{Q}$), standardized answers to questions ($\mathcal{A}$), and answer alias sets. In these datasets, both single edit and multiple edits contain questions about historical and current knowledge, while extending edits contain only questions about the current knowledge. Single edit

[5]Without loss of generality, we use GPT-J as the representative LLM used in the experiments.

Method	AToKe-SE					AToKe-ME						AToKe-EE		
	Current			Historical		Current			Historical		Edited	Current		
	CES	CES-P	CRS	HES	HRS	CES	CES-P	CRS	HES	HRS	HES*	CES	CES-P	CRS
CFT	5.73	5.69	5.34	0.06	0.02	1.11	1.18	1.22	0.03	0.00	0.01	3.41	2.91	3.15
MEND	80.47	40.56	32.46	1.73	0.68	71.83	27.96	23.67	0.40	**2.10**	0.25	91.94	62.48	51.63
ROME	**99.99**	**97.01**	**81.64**	**2.41**	**1.56**	**98.85**	**91.54**	**77.08**	0.44	1.17	0.26	**99.93**	**98.70**	**85.84**
MEMIT	99.66	92.23	75.31	2.22	1.21	98.42	91.06	66.48	**0.48**	0.86	**0.27**	99.92	95.82	72.76

Table 2: Results of existing models on the benchmark AToKe-SE, AToKe-ME and AToKe-EE. "Edited" means the score is computed when all the multiple edits are completed. The best results are highlighted in **BOLD**.

and extending edit have only one editing operation and corresponding questions for one (s, r), while multiple edits contain multiple operations and questions.

Data statistics In the base temporal data collection phase, we first sample 169,996 temporal fact chains. After filtering by GPT-J, there are 13 different relations left and the number of our temporal chains is reduced to 8,820, which is the final number of the three datasets. For the length distribution of temporal fact chains, fact chains of length 2-5 account for more than 90%, with very few chains exceeding 10. In addition, the relationship "playsFor" has the largest number of fact chains. The time is measured in years. And for the time scope, before adding the future fake facts, the latest knowledge is obtained in 2022, and after that it is extended to 2028.

Evaluation Metrics

As mentioned before, we categorize questions into relative time questions and explicit time questions, and both of them are about historical and current knowledge. Therefore, we have the following four indicators to assess the performance of TKE: 1) **Historical Relative time Question Score** (**HRS**), which is the accuracy of relative time questions about historical knowledge answered by the edited model. 2) **Historical Explicit time Question Score** (**HES**), which is the accuracy of explicit time questions asked about historical knowledge. 3) **Current Relative time Question Score** (**CRS**), which is the accuracy of relative time questions about current knowledge. 4) **Current Explicit time Question Score** (**CES**), which measures accuracy of explicit time questions about current knowledge. In addition, the prompt we used as the optimization target when editing is the same as the questions in CES. Therefore, to make results more valid, we use paraphrases of the CES questions for further evaluation, denoted as 5) **Current Explicit time Paraphrase Question Score** (**CES-P**).

In particular, in AToKe-ME, we treat each edit as a single edit and compute the score as in AToKe-SE, and finally report the average of all edits. In addition, we calculate HES for all the historical facts at the end of multiple consecutive edits of the model to measure the overall editing performance (denoted as HES*). In AToKe-EE, since there is no historical knowledge, we do not need to measure HRS and HES.

Current Status on TKE

In order to better assess the performance of existing methods for temporal knowledge editing, we conduct experiments with several popular editing methods and analyze the results.

Experimental Setup

Knowledge Editing Methods We evaluate the following state-of-the-art editing methods on AToKe. 1) **Constrained Fine-tuning (CFT)** (Zhu et al. 2020) performs gradient descent on the target knowledge to minimize the loss and set a norm constraint on model weight changes. 2) **MEND** (Mitchell et al. 2022a) learns a hypernetwork to produce weight updates using a low-rank decomposition of the gradient obtained by standard fine-tuning. 3) **ROME** (Meng et al. 2022a) firstly localizes the factual knowledge in the model by causal tracing, and then treats the MLP modules as key-value stores to insert new knowledge by making a rank-one change. 4) **MEMIT** (Meng et al. 2022b), a successor to ROME, can insert lots of memories at once by modifying the MLP weights of a range of critical layers.

All of the above model editing methods make changes to the parameters of the model, which is the focus of our exploration.

Given a knowledge editing operation that is expected to be learned $e = (s, r, o, t_s, t_u) \rightarrow (s, r, o^\star, t_s^\star, t_u^\star)$, we convert it to a cloze statement by natural language template $q_r(\cdot)$ which is used as the input to the above knowledge editing approaches. Note that all the previous methods use only $q_r((s, r, o))$ as input; to ensure that the information is sufficient and adapted to our task, we have expanded these methods to include time for each input cloze statement.

Language Model to be Edited Following setup of previous work (Meng et al. 2022a,b; Zhong et al. 2023), we use GPT-J (6B) (Wang and Komatsuzaki 2021) as the base LLM to be edited with above methods.

Results and Analysis

The performance of existing knowledge editing methods on our benchmarks is shown in Table 2. For models, CFT is weaker and hence performs poorly. ROME performs slightly better than MEMIT, probably because we edit one piece of knowledge at a time and hence are better suited to ROME. After comparison and analysis we can get the following main conclusions:

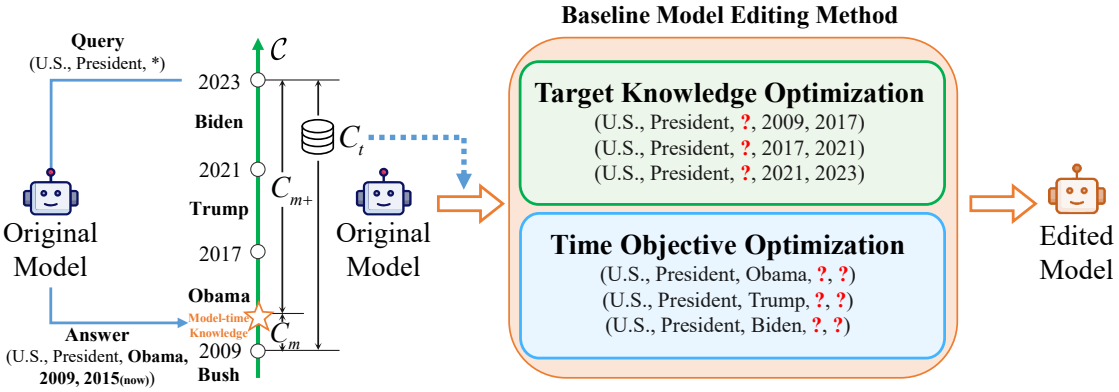

Figure 3: Demonstration of the METO editing framework. First the model will be queried based on the current knowledge to get the knowledge under the model time (C_m). Then both historical and current knowledge are used as target knowledge for target knowledge optimization and time objective optimization with any model editing methods.

Remembering the new and forgetting the old Except for CFT, edited models can remember new knowledge very well and do some generalization, but the performance on historical knowledge is disastrous. It is consistent with our expectations, as all existing model editing methods optimize the probability of current knowledge and ignore historical knowledge. An example is shown in Figure 1, where *Donald Trump* is forgotten by the edited models.

Relative time questions are more difficult than explicit time questions We can observe that it is more difficult for the model to answer questions about a piece of knowledge without explicitly providing it with a specific time, using a relative time expression such as "last one". It may be because using relative time expression requires the model to reason about the order in which facts occur, whereas using explicit time expression simply requires the model to remember when facts occur.

The more edits, the worse the performance The average results on ATOKE-ME are sightly worse than those on ATOKE-SE, which may be because editing multiple times on the same fact causes a little confusion to the edited model. As we can see that HES score is only about 0.2% after multiple edits, and all the knowledge injected before the last edit is invalidated.

Extending time scope is easier than inserting new object We can see that methods on ATOKE-EE perform the best of three datasets, which is reasonable because the extending edit task does not change the current knowledge, but merely extends its time scope. The correct answer is not changed when the question is asked about the fact.

METO: Multi-Editing with Time Objective

As we can see from the previous experiments in Section , the existing methods perform excellently in memorizing new knowledge, but catastrophic forgetting of historical knowledge occurs after editing. To alleviate this serious problem, we propose a simple editing framework METO (Multi-Editing with Time Objective) that can be applied to enhance existing editing methods easily.

Methodology

As shown in Figure 3, We firstly query the language model with the cloze statement of current knowledge to extract the knowledge under model time, and then edit the model using both of them with timestamps, and also use the model to make knowledge time predictions so that the model can reinforce both historical and new knowledge, and improve the awareness of time of knowledge.

Model-time Knowledge Extraction For a chain of temporal facts $\mathcal{C} = \{(s, r, o_1, t_{s_1}, t_{u_1}), ..., (s, r, o_N, t_{s_N}, t_{u_N})\}$, the object is constantly changing and the knowledge of the model about $(s, r, \cdot)$ is determined by the collection time of its training corpus. Therefore, in order to better preserve the historical knowledge inside the model, we need to extract the model-time knowledge about $(s, r, \cdot)$. With the method in Section , we obtain the fact of what is happening under the model time which is noted as $C_m = \{(s, r, o_i, t_{s_i}, t_{u_i})\}$. By comparing $\mathcal{C}$ and C_m, we can get the knowledge that needs to be newly captured by the model $C_{m+} = \{(s, r, o_{i+1}, t_{s_{i+1}}, t_{u_{i+1}}), ..., (s, r, o_N, t_{s_N}, t_{u_N})\}$.

Multi-editing on Both Historical and Current Knowledge Combining C_m and C_{m+}, we obtain the set of all the target knowledge C_t which is used as the target of editing. Using the target knowledge C_t obtained above as input, we can edit the language model with any model editing method, such as MEND, ROME, MEMIT, etc. Note that in the specific implementation, to suit our task, we have also added timestamps in the cloze statement of original methods.

It is worth explaining that we only use C_m which is occurring at the model time and do not use the full previous knowledge, C_{m-}, as the historical knowledge. It is because we believe that the model already knows that C_{m-} is the history that has happened in the past which does not need to be changed. For C_m, on the other hand, the model needs to be made aware of the duration of this knowledge, which has

Method	ATOKE-SE					ATOKE-ME					
	Current			Historical		Current			Historical		Edited
	CES	CES-P	CRS	HES	HRS	CES	CES-P	CRS	HES	HRS	HES*
CFT+	↓2.93	↓3.07	↓3.08	↑3.32	↑2.41	↑0.16	↑0.02	↓0.18	↑1.61	↑1.23	↑0.71
MEND+	↑2.79	↓7.11	↓7.05	↑28.41	**↑29.49**	↓1.31	↑0.45	↓1.83	↑28.25	**↑28.73**	↑21.58
ROME+	**↓0.04**	**↓3.23**	**↓2.76**	↑17.84	↑14.73	**↑1.08**	**↓0.57**	↑5.32	↑22.78	↑17.01	↑15.66
MEMIT+	↓13.26	↓6.91	↓1.24	**↑28.09**	↑23.11	↓5.69	↓5.31	↑7.10	**↑35.72**	↑25.18	**↑21.66**

Table 3: Results of enhanced models with METO (marked with "+") on the benchmark ATOKE. Due to space constraints, we report the difference from the results in Table 2. The final best results are highlighted in **BOLD**.

ended and changed from the present tense to the past tense.

Time Objective Optimization In order to further enhance the model's awareness of the time of knowledge, along with the multi-editing described above, we perform an additional task of optimizing the time objective of knowledge. We also use the same editing method of existing model to ensure the generalization of our framework, except that the editing target is changed from the object to the corresponding time. As an example, given an input of "Donald Trump is the President of the United States from", the model is then edited to optimize the probability of "2017 to 2021" with ROME or other methods.

Results on ATOKE-SE and ATOKE-ME

Since existing methods perform relatively well on ATOKE-EE and it does not involve the questioning of historical facts, we test with METO enhancement for editing on ATOKE-SE and ATOKE-ME (shown in Table 3).

We can find that maintaining little change in performance on current knowledge, our framework greatly improves existing editing methods' performance on historical knowledge. Among them, ROME stills performs best in memorizing current knowledge. Surprisingly, MEND and MEMIT perform better on historical knowledge, which may be due to the fact that our framework prefers that editing methods can edit more than one piece of knowledge at a time, which in turn can preserve historical knowledge.

The editing experiments with our framework also further validate our two previous conclusions: 1) relative time questions are more difficult than explicit time questions. 2) the more edits, the worse the performance. It is worth noting that both phenomena have been mitigated to some extent.

It is promising that, as shown in Table 3, the HES* score on ATOKE-ME improves from the original result of less than 0.3 to more than 15 (except for CFT). Such a significant improvement in this most difficult metric also shows that our framework is beneficial and the edited model remembers some of the previously injected knowledge.

Although there has been a substantial improvement in memorizing historical knowledge, it is still far from a satisfactory level, reflecting the difficulty of our proposed task of temporal knowledge editing, which still requires a concerted effort by the community.

Related Work

The expanding parameter count of language models leads to higher retraining costs. And since the knowledge inside a model becomes progressively outdated, knowledge editing (KE), a convenient way to edit the knowledge, has received increasing attention and some methods have been proposed. Zhu et al. (2020) propose the constrained finetuning approach on modified facts to solve the problem. Sotoudeh and Thakur (2019) utilize symbolic representations and generalized RELU networks. (Hahnloser et al. 2000) to correct model with small patches. Cao, Aziz, and Titov (2021) introduce the method which corrects knowledge and improves predictions using a hyper-network for targeted modifications. Dai et al. (2022) explore knowledge neurons in models and attempt to leverage them to edit specific factual knowledge. ROME (Meng et al. 2022a) and MEMIT (Meng et al. 2022b) take the approach of locating and then editing. Mitchell et al. (2022a) a collection of small auxiliary editing networks that use a single desired input-output pair to make fast, local edits. In addition, MEMIT-CSK (Gupta et al. 2023) is designed to edit commonsense knowledge.

Evaluation metrics on KE generally focus on whether the editing is successful and whether other unrelated knowledge is affected. Onoe et al. (2023) also evaluate abilities of updated LLM by making inferences based on injected facts. Furthermore, Zhong et al. (2023) build multi-hop questions to assess edited models on related facts. And Cohen et al. (2023) evaluate the ripple effects in edited models. However, no work has yet noted that pre-editing historical knowledge should also be preserved, which we argue is very important.

Conclusion

In this paper, we systematically classify scenarios involving knowledge editing, identify the shortcomings leading to historical knowledge distortion in existing model editing methods. To facilitate a comprehensive evaluation of KE techniques, we introduce the task of temporal knowledge editing (TKE) and present a new benchmark named ATOKE. We conduct experiments on ATOKE and demonstrate that existing methods lead to a catastrophic forgetting of historical knowledge. To bridging existing gaps, we present the METO editing framework, enhancing the efficacy of preceding approaches. However, TKE still remains challenging and calls for more efforts in the community.

Acknowledgments

This work was supported by National Key R&D Program of China (2021YFF0901502), National Science Foundation of China (No. 62161160339), State Key Laboratory of Media Convergence Production Technology and Systems and Key Laboratory of Science, Technology and Standard in Press Industry (Key Laboratory of Intelligent Press Media Technology). We appreciate the anonymous reviewers for their helpful comments. Xiaojun Wan is the corresponding author.

References

AlKhamissi, B.; Li, M.; Celikyilmaz, A.; Diab, M.; and Ghazvininejad, M. 2022. A Review on Language Models as Knowledge Bases. arXiv:2204.06031.

Bordes, A.; Usunier, N.; Garcia-Duran, A.; Weston, J.; and Yakhnenko, O. 2013. Translating embeddings for modeling multi-relational data. *Advances in neural information processing systems*, 26.

Brown, T. B.; Mann, B.; Ryder, N.; Subbiah, M.; Kaplan, J.; Dhariwal, P.; Neelakantan, A.; Shyam, P.; Sastry, G.; Askell, A.; Agarwal, S.; Herbert-Voss, A.; Krueger, G.; Henighan, T.; Child, R.; Ramesh, A.; Ziegler, D. M.; Wu, J.; Winter, C.; Hesse, C.; Chen, M.; Sigler, E.; Litwin, M.; Gray, S.; Chess, B.; Clark, J.; Berner, C.; McCandlish, S.; Radford, A.; Sutskever, I.; and Amodei, D. 2020. Language Models are Few-Shot Learners. arXiv:2005.14165.

Cao, N. D.; Aziz, W.; and Titov, I. 2021. Editing Factual Knowledge in Language Models. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, 6491–6506.

Cohen, R.; Biran, E.; Yoran, O.; Globerson, A.; and Geva, M. 2023. Evaluating the Ripple Effects of Knowledge Editing in Language Models. arXiv:2307.12976.

Dai, D.; Dong, L.; Hao, Y.; Sui, Z.; Chang, B.; and Wei, F. 2022. Knowledge Neurons in Pretrained Transformers. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics*, 8493–8502.

Dasgupta, S. S.; Ray, S. N.; and Talukdar, P. 2018. HyTE: Hyperplane-based Temporally aware Knowledge Graph Embedding. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, 2001–2011. Brussels, Belgium: Association for Computational Linguistics.

Davison, J.; Feldman, J.; and Rush, A. 2019. Commonsense Knowledge Mining from Pretrained Models. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, 1173–1178. Hong Kong, China: Association for Computational Linguistics.

Gupta, A.; Mondal, D.; Sheshadri, A. K.; Zhao, W.; Li, X. L.; Wiegreffe, S.; and Tandon, N. 2023. Editing Commonsense Knowledge in GPT. arXiv:2305.14956.

Hahnloser, R.; Sarpeshkar, R.; Mahowald, M.; Douglas, R.; and Seung, H. 2000. Digital selection and analogue amplification coexist in a cortex-inspired silicon circuit. *Nature*, 405: 947–51.

Hartvigsen, T.; Gabriel, S.; Palangi, H.; Sap, M.; Ray, D.; and Kamar, E. 2022. ToxiGen: A Large-Scale Machine-Generated Dataset for Adversarial and Implicit Hate Speech Detection. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, 3309–3326. Dublin, Ireland: Association for Computational Linguistics.

Hase, P.; Diab, M.; Celikyilmaz, A.; Li, X.; Kozareva, Z.; Stoyanov, V.; Bansal, M.; and Iyer, S. 2021. Do Language Models Have Beliefs? Methods for Detecting, Updating, and Visualizing Model Beliefs. arXiv:2111.13654.

Ji, Z.; Lee, N.; Frieske, R.; Yu, T.; Su, D.; Xu, Y.; Ishii, E.; Bang, Y. J.; Madotto, A.; and Fung, P. 2023. Survey of Hallucination in Natural Language Generation. *ACM Comput. Surv.*, 55(12).

Lin, Y.; Liu, Z.; Sun, M.; Liu, Y.; and Zhu, X. 2015. Learning Entity and Relation Embeddings for Knowledge Graph Completion. *Proceedings of the AAAI Conference on Artificial Intelligence*, 29(1).

Mahdisoltani, F.; Biega, J. A.; and Suchanek, F. M. 2015. YAGO3: A Knowledge Base from Multilingual Wikipedias. In *Conference on Innovative Data Systems Research*.

Meng, K.; Bau, D.; Andonian, A.; and Belinkov, Y. 2022a. Locating and Editing Factual Knowledge in GPT. *CoRR*, abs/2202.05262.

Meng, K.; Sharma, A. S.; Andonian, A.; Belinkov, Y.; and Bau, D. 2022b. Mass-Editing Memory in a Transformer. arXiv:2210.07229.

Mitchell, E.; Lin, C.; Bosselut, A.; Finn, C.; and Manning, C. D. 2022a. Fast Model Editing at Scale. In *Proceedings of the 10th International Conference on Learning Representations*.

Mitchell, E.; Lin, C.; Bosselut, A.; Manning, C. D.; and Finn, C. 2022b. Memory-Based Model Editing at Scale. In *Proceedings of the 2022 International Conference on Machine Learning*, 15817–15831.

Onoe, Y.; Zhang, M. J. Q.; Padmanabhan, S.; Durrett, G.; and Choi, E. 2023. Can LMs Learn New Entities from Descriptions? Challenges in Propagating Injected Knowledge. arXiv:2305.01651.

OpenAI. 2023. GPT-4 Technical Report. arXiv:2303.08774.

Ouyang, L.; Wu, J.; Jiang, X.; Almeida, D.; Wainwright, C. L.; Mishkin, P.; Zhang, C.; Agarwal, S.; Slama, K.; Ray, A.; Schulman, J.; Hilton, J.; Kelton, F.; Miller, L.; Simens, M.; Askell, A.; Welinder, P.; Christiano, P.; Leike, J.; and Lowe, R. 2022. Training language models to follow instructions with human feedback. arXiv:2203.02155.

Petroni, F.; Rocktäschel, T.; Riedel, S.; Lewis, P.; Bakhtin, A.; Wu, Y.; and Miller, A. 2019a. Language Models as Knowledge Bases? In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, 2463–2473. Hong Kong, China: Association for Computational Linguistics.

Petroni, F.; Rocktäschel, T.; Lewis, P.; Bakhtin, A.; Wu, Y.; Miller, A. H.; and Riedel, S. 2019b. Language Models as Knowledge Bases? arXiv:1909.01066.

Sinitsin, A.; Plokhotnyuk, V.; Pyrkin, D. V.; Popov, S.; and Babenko, A. 2020. Editable Neural Networks. In *Proceedings of the 8th International Conference on Learning Representations*.

Sotoudeh, M.; and Thakur, A. 2019. Correcting deep neural networks with small, generalizing patches. In *Workshop on safety and robustness in decision making*.

Touvron, H.; Lavril, T.; Izacard, G.; Martinet, X.; Lachaux, M.-A.; Lacroix, T.; Rozière, B.; Goyal, N.; Hambro, E.; Azhar, F.; Rodriguez, A.; Joulin, A.; Grave, E.; and Lample, G. 2023. LLaMA: Open and Efficient Foundation Language Models. arXiv:2302.13971.

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

Zheng, C.; Li, L.; Dong, Q.; Fan, Y.; Wu, Z.; Xu, J.; and Chang, B. 2023. Can We Edit Factual Knowledge by In-Context Learning? arXiv:2305.12740.

Zhong, Z.; Wu, Z.; Manning, C. D.; Potts, C.; and Chen, D. 2023. MQuAKE: Assessing Knowledge Editing in Language Models via Multi-Hop Questions. arXiv:2305.14795.

Zhu, C.; Rawat, A. S.; Zaheer, M.; Bhojanapalli, S.; Li, D.; Yu, F.; and Kumar, S. 2020. Modifying Memories in Transformer Models. arXiv:2012.00363.