How to Precisely Update Large Language Models Knowledge While Avoiding Catastrophic Forgetting

Fei Ding,* **Xu Zhang**
♠ APUS AI Lab

Abstract

Recent advancements in Large Language Models (LLMs) have showcased their remarkable capabilities in text understanding and generation. However, even stronger LLMs are susceptible to acquiring erroneous or obsolete information from the training corpus. Direct secondary fine-tuning with data containing new knowledge may be ineffective in updating knowledge due to the conflict between old and new knowledge. In this paper, we propose a new paradigm for fine-tuning called **DFT** (Delicate Fine-Tuning).This method utilizes parametric arithmetic to precisely pinpoint the location of knowledge and update only the minimal set of relevant parameters . Experimental results on two publicly available datasets demonstrate that our proposed DFT can obviously improve the knowledge updating performance of full fine-tuning , simultaneously outperforming the existing baselines in most cases.

1 Introduction

Large Language Models (LLMs) possess an extraordinary ability to understand and generate natural language (Brown et al., 2020; Raffel et al., 2020; Ouyang et al., 2022). Although LLMs are very capable of learning, they are not immune to the acquisition of incorrect knowledge in the corpus. Moreover, much of the knowledge in the real world is constantly updated, and some of the originally correct knowledge in LLMs can become outdated and invalid over time. For example, the question "Who is the President of the United States? is answered "Donald Trump" in the year 2020, while the answer now is "Joe Biden". Consequently, the challenge with LLMs is continuously updating to ensure they reflect current, correct knowledge. Existing methods of model editing and knowledge updating usually add additional network (Dong et al., 2022; Huang et al., 2022; Raunak and Menezes, 2022), model parameters (Dai et al., 2023; Dong et al., 2022; Huang et al., 2022), knowledge bases (Murty et al., 2022; Mitchell et al., 2022; Li et al., 2022; Madaan et al., 2022; Mitchell et al., 2022; Zheng et al., 2023), etc., and the editing process is not as straightforward and simple as fine-tuning methods (Zhang et al., 2022; Li and Liang, 2021; Hu et al., 2021) directly with new knowledge. Currently, the most used method for learning new knowledge is still direct fine-tuning of the model.

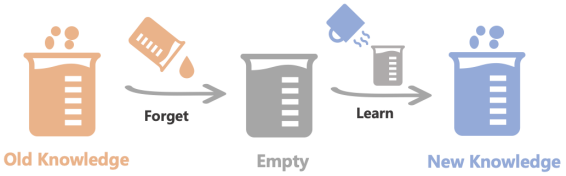

Figure 1: Diagram for “Forgetting before Learning”.

Empirically, when human beings establish their own initial cognition, if they are exposed to new knowledge that is inconsistent with their initial cognition, they usually remember both the new knowledge and the old knowledge simultaneously, and then often get confused, leading to contradictions that make it difficult to learn the new knowledge. If we directly modify the memory of old knowledge and original cognition, , then the new knowledge to be learned will not conflict with the original cognition and knowledge, which makes it better to learn and absorb the new knowledge. As shown in Figure 1, it is better to pour in the "new water" only after the "original water" in the cup has been poured out. For example, if people have been educated to believe that "the Earth is flat" since childhood, it would be challenging for them to accept the conflicting knowledge that "the Earth is round" when they become adults. Conversely, if they could directly modify their memory of the er-

*Corresponding author
email:dingfei@email.ncu.edu.cn

roneous knowledge "the Earth is flat" to the correct knowledge "the Earth is round," it would be much simpler.

So how do we locate the position of old knowledge and then update it accurately? Our research has shown that when fine-tuning large language models, they tend to learn sentence structure, grammar, and style first, with knowledge being acquired last. Therefore, we control the variables to prevent the model from learning sentence structure and stylistic information.

Inspired by the above empirical observations and (Ilharco et al., 2022)'s task arithmetic, we propose a novel paradigm of knowledge updating called **DFT** (Delicate Fine-Tuning). Specifically, We begin by using the large language model to predict and generate an answer, resulting in a data point. Next, we modify only the key knowledge within the sentence, keeping the sentence structure and style intact, creating a new data point. We then fine-tune the model separately with both data points, recording the parameter changes. By comparing these parameter changes, we identify sections that exhibit similar changes in direction. These sections, representing aspects that are not relevant to the knowledge update, are discarded entirely. We retain only the parameters exhibiting contrasting change directions, then compare their differences, rank those differences, and identify the top 10% with the largest differences .Then update the top 10% of parameters . The whole process is repeated iteratively until the model's knowledge update is complete. The contribution of this work can be summarised as follows:

- We propose a novel fine-tuning paradigm "**DFT** (Delicate Fine-Tuning)" for knowledge updating in large language models.
- Experimental results show that our proposed **DFT** (Delicate Fine-Tuning) improves the knowledge updating performance of various fine-tuning methods and outperforms the existing baselines in most cases.

2 Related Work

Currently, the method of knowledge updating and model editing (also known as knowledge editing) for LLMs is mainly divided into two classes (Yao et al., 2023; Wang et al., 2023b):

a. The method preserving model's parameters **Retrieve augmentation** practically depends on an external knowledge base which contains new or correct knowledge. Aiming at amending the output of LLMs, a new knowledge base will be connected with the base model to implement a retrieve for needed new knowledge to a prompt or a question (Murty et al., 2022; Mitchell et al., 2022; Li et al., 2022; Madaan et al., 2022). Mitchell et al. (Mitchell et al., 2022) store manual edits in a memory module, and use a classifier to call the knowledge stored in the memory. Madaan et al. (Madaan et al., 2022) leverage the memory of user's feedback to generate prompts for LLMs. Instead of gradient calculation, Zheng et al. (Zheng et al., 2023) utilize the in-context learning method to revise the output of LLMs with demonstrations extracted from the corpus based on similarity.

Adding Additional Parameters refers to injecting a few trainable parameters which represent new knowledge to LLMs while original parameters keeping frozen (Dong et al., 2022; Huang et al., 2022; Raunak and Menezes, 2022; Dai et al., 2023). Dong et al. (Dong et al., 2022) put forward a lightweight feed-forward network to add new parameters adapted to specific factual contexts for knowledge generalization. Huang (Huang et al., 2022) et al. design an editor called Transformer-Patcher, which is capable of modifying the mistake of LLMs sequentially by adding and training a few neurons in transformer.

b. The method modifying model's parameters **Fine-tuning** is a general technique since pre-training model has been widely adopted in NLP research, which always obtains promising results in downstream tasks. Meanwhile, fine-tuning is an intuitive and effective method to urge the model to learn new knowledge for model editing (Zhu et al., 2020; Zhang et al., 2022; Yao et al., 2023). Recently, there are a series of parameter-efficient fine-tuning methods, such as Prefix-Tuning (Li and Liang, 2021) and LoRA (Hu et al., 2021), making it more appreciate for knowledge editing based on fine-tuning. Zhang et al. (Zhang et al., 2022) operate incremental parameter updates of different amounts by calculating the importance of the weight matrix to improve the update efficiency and adaptability. Zhu et al. (Zhu et al., 2020) leverage a loss constraint attached to the base model to reduce the impact on irrelevant knowledge during the process of fine-tuning. Similarly, Lee et al. (Lee et al., 2022) also implement large-scale continual learning for knowledge updating with regularized

fine-tuning.

Meta-learning is aimed at updating the knowledge in LLMs through varying their parameters with the prediction from a well-trained hypyernetwork (Sinitsin et al., 2019; Mitchell et al., 2021; De Cao et al., 2021). Mitchell et al. (Mitchell et al., 2021) propose an auxiliary network with gradient decomposition, which can execute efficient edits to LLMs according to a single input-output pair. De Cao et al. (De Cao et al., 2021) update part of weights for a subset of modules in the model relying on a hypernetwork with constrained optimization.

Locate and edit is related to the internal mechanism of the LLMs. With the help of some attributes, it usually locates the parameters and neurons in the light of specific knowledge and modifies them to correct the output (Meng et al., 2022a; Dai et al., 2022; Meng et al., 2022b; Santurkar et al., 2021; Geva et al., 2022). Geva et al. (Geva et al., 2021) find that the feed-forward networks layer of the transformer stores key-value pairs which are related to specific knowledge. Meng et al. (Meng et al., 2022a) utilize a causal reasoning method to distinguish the key neuron activations and update specific factual associations by modifying feed-forward weights. Furthermore, to implement knowledge editing on a large scale, they put forward MEMIT (Meng et al., 2022b), a method that directly updates thousands of memories in LLMs. Gupta et al. (Gupta et al., 2023) improve the knowledge updating through varying edit tokens and ameliorating the layer selection during the editing process. Yu et al. (Yu et al., 2023) leverage the partitioned gradient to identify the significant weights for unlearning of bias in the model.

Hiyouga (hiyouga, 2023) developed the fastedit software framework, which enables convenient editing of models using causal reasoning. Zhang et al. (Zhang et al., 2024; Wang et al., 2023a; Yao et al., 2023; Cheng et al., 2023; Mao et al., 2023; Zhang et al., 2023) developed the EasyEdit software framework, which makes it easy to use a variety of methods for editing models.

In conclusion, there are many ways to achieve knowledge updating, but most of them require the addition of additional knowledge bases, neural network modules, and model parameters, which are cumbersome in practice and increase inference consumption. **This paper focuses on the improvement and enhancement of fine-tuning methods.**

3 Task Definition

Our task is knowledge updating of large models, which can be defined as given a model f_θ and a set of input-output knowledge pairs $K_{old} = \{(x_1, y_1), (x_2, y_2), ..., (x_i, y_i)\}$, the parameters of the model need to be edited to obtain a new model f_{θ^*} (x) and a corresponding set of new input-output pairs $K_{new} = \{(x_1, y_1^{new}), (x_2, y_2^{new}), ..., (x_i, y_i^{new})\}$. The i is the number of knowledge pairs to be updated. Referring to (Yao et al., 2023), we can define this process and objective of knowledge updating as:

$$f_{\theta^*}(x_i) = \begin{cases} y_i^{new} & \text{if } x_i \in N(x_i) \\ f_\theta(x_i) & \text{if } x_i \in other \end{cases} \quad (1)$$

where $N(x_i)$ represents x_i itself and its equivalent neighbourhood. The knowledge update task needs to update only the answers of x_i itself and its equivalent domain $N(x_i)$ without changing the answers of other out-of-scope knowledge. Specifically, the quality of knowledge updating has the following three evaluation indicators: (1) **Reliability** is measured as the average accuracy on the new knowledge for the updated model f_{θ^*}. It's the first indicator of the effectiveness of knowledge updating. As shown in Figure 2, the output of the question "Who is the President of the US?" needs to be updated from "Donald Trump" to "Joe Biden". (2) **Generalization** means the new model f_{θ^*} should also updated the equivalent neighbour $N(x_i)$ (e.g. rephrased sentences). It is evaluated by the average accuracy of the model f_{θ^*} on examples drawn uniformly from the equivalence neighborhood. As shown in Figure 2, the output of the question "Who holds the position of the President of the US?" also needs to be updated from "Donald Trump" to "Joe Biden". (3) **Locality** means the updated model f_{θ^*} should not change the output of the irrelevant examples. Hence, the locality is evaluated by the rate at which the updated model f_{θ^*}'s predictions are unchanged as the pre-update f_θ model. As shown in Figure 2, the output of the question "'You're fired!' is the catchphrase of which celebrity?" is to be kept unchanged as "Donald Trump".

4 Proposed method: DFT

In this section, we will present our method of knowledge updating for LLMs. Instead of introducing an external knowledge base or additional parameters, our method is mainly based on **full fine-tuning** . Briefly, it consists of two stages:

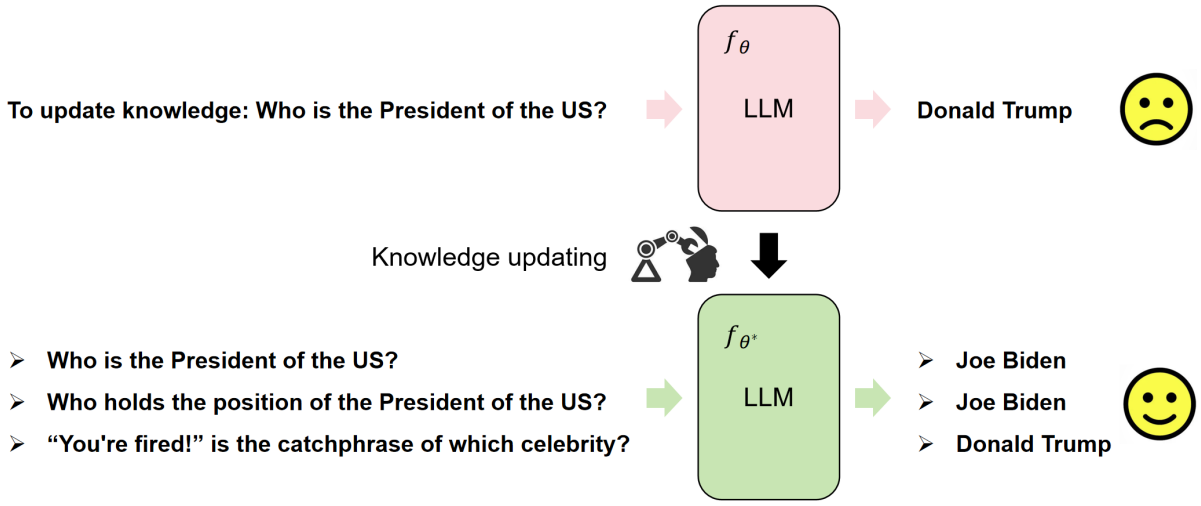

Figure 2: Objectives of the knowledge updating in large language model.

4.1 Find where the old knowledge is stored

The supervised fine-tuning (SFT) on a dataset allows us to determine the direction in which the model parameters align with the knowledge. , which is reflected in the variation of the model's parameters. During this stage, for a given large language model f_θ and its parameters θ, we define the incremental parameters as **knowledge parameters** $\theta_\triangle$, calculated as follows:

$$\theta_\triangle = \mathrm{FT}\{\theta, \mathrm{K}\} - \theta \tag{2}$$

where FT is the operation of supervised fine-tuning, while K, θ refer to the dataset of knowledge and the parameters of the original model f_θ, respectively. Similarly, we first fine-tune the model f_θ on a dataset containing old knowledge, and then subtract the parameters θ of the original model f_θ from model's parameters after fine-tuning to obtain the knowledge parameters $\theta_\triangle^{old}$ indicating the old knowledge, as follows:

$$\theta_\triangle^{old} = \mathrm{FT}\{\theta, \mathrm{K_{old}}\} - \theta \tag{3}$$

where K_{old} refers to a dataset composed of old knowledge which we desire to forget. The related work in (Ilharco et al., 2022) considers that subtracting the parameters $\theta_\triangle^{old}$ from θ can assist the model f_θ to forget this part of old knowledge.

$$\theta' = \theta - \lambda\theta_\triangle^{old}, \tag{4}$$

where λ is a hyper-parameter to control the rate of forgetting. Now we gain a new model $f_{\theta'}$ with its parameters θ', which has forgotten the old knowledge compared to f_θ. Note that this process of forgetting old knowledge only makes sense if the model f_θ has already learned the old knowledge, otherwise, there is no need for forgetting and the forgetting operation may have a destructive effect on the normal knowledge of the model.

However, we believe that $\theta_\triangle^{old}$ also contains other information such as sentence structure, grammar, and style, which requires further processing to accurately pinpoint the old knowledge.

Then, we re-fine-tune the model f_θ on a dataset containing new knowledge, and then subtract the parameters θ of the original model f_θ from model's parameters after fine-tuning to obtain the knowledge parameters $\theta_\triangle^{old}$ indicating the new knowledge, as follows:

$$\theta_\triangle^{new} = \mathrm{FT}\{\theta, \mathrm{K_{new}}\} - \theta \tag{5}$$

where K_{new} refers to a dataset composed of new knowledge .

Then, We compare $\theta_\triangle^{old}$ and $\theta_\triangle^{new}$, discarding all elements with the same sign. Then, we select the top 10% of elements in $\theta_\triangle^{new}$ with the largest difference from $\theta_\triangle^{old}$.

$$\theta_\triangle^{core} = \mathrm{f}\{\theta_\triangle^{\mathrm{new}}, \theta_\triangle^{\mathrm{old}}\} \tag{6}$$

$\theta_\triangle^{core}$ is the crucial parameter that needs to be updated.

4.2 Learning new knowledge

We define the process of learning new knowledge as follows:

$$\theta^* = \theta + \lambda\theta_\triangle^{core} \tag{7}$$

where λ is a hyper-parameter to control the rate of learning. We repeat the processes outlined in equations (3), (5), (6), and (7) until the model's output reflects the new knowledge. Now we gain a new model f_{θ^*} with its parameters θ^*, which has forgotten the old knowledge compared to f_θ. It learns only the new knowledge, avoiding any other information, preventing catastrophic forgetting caused by style changes and the like.

5 Experiments

5.1 Datasets

In this work, we use ZsRE (Levy et al., 2017) and COUNTERFACT (Meng et al., 2022a), two widely used datasets, for our experiments. ZsRE is a Question Answering (QA) dataset that utilizes question rephrasings generated by back-translation as the equivalence neighborhood. COUNTERFACT is a more challenging dataset with counterfactual data. We follow the setting of (Yao et al., 2023) to take the eval and edit sets of which there are 19,085 and 10,000 pieces of data respectively. Moreover, we divide the datasets into two parts of old knowledge and new knowledge respectively to achieve two-stage knowledge update. The following is an example of old knowledge and new knowledge in zsRE, which represents the modification of knowledge from "Los Angeles" to "New Orleans". More details about the datasets and examples can be found in the appendix A.1.

The old knowledge:

{**"instruction"**: "What city did Marl Young live when he died?", **"input"**: "", **"output"**: "Los Angeles" }

The new knowledge:

{**"instruction"**: "What city did Marl Young live when he died?", **"input"**: "", **"output"**: "New Orleans" }

5.2 Baselines

To evaluate the effectiveness of the proposed DFT method, we conducted experiments on fine-tuning methods and locate-based methods. For fine-tuning methods, we first compare with the full fine-tuning (**Full-FT**) and **LoRA** (Hu et al., 2021), respectively. LoRA (Low-Rank Adaptation) is a technique for fine-tuning large pre-trained language models by introducing small, trainable matrices into each layer of the model's architecture, allowing for efficient adaptation while keeping the majority of the model's parameters frozen. Then we experiment with a fine-tuning approach (**FT-c**) (Zhu et al., 2020) that leverages L_∞ constraint to retain old irrelevant knowledge. For locate-based methods, we first experiment with **ROME** (Meng et al., 2022a), a method updating specific factual associations with causal intervention. Finally, we compare with the **MEMIT** (Meng et al., 2022b) which is a effective method to directly update large-scale memories.

5.3 Completion Details

We use LLAMA2-7B and LLAMA-7B as the base models for our experiments. We are mainly evaluating the ability to update old knowledge to new knowledge, thus we trained the base model on the old knowledge for 3 epochs by full fine-tuning as the **original model** in our experiment (as the same as **original model** in other experiments). Original model has fully learned old knowledge, which makes the forgetting operation reasonable and necessary. To ensure the uniqueness of the model output, we used the greedy decoding strategy during testing. On the hardware side, a total of 8 × A800-80G GPUs were used for the experiments. More details about the experimental settings can be found in the appendix A.2.

5.4 Experimental Results

The experimental results are presented in Table 1, indicating a notable enhancement in learning following the initial forgetting process, irrespective of whether full fine-tuning or LoRA is employed. We obtain the promising results as our DFT method outperforming other baselines in most cases. Specifically, compared with other editors, FT-c has only small improvements over the original model. This could be attributed to its norm regularization, which makes FT-c tend to retain a part of the old knowledge during the knowledge updating process. Given that our original model has fully learned a large quantity of old knowledge, it faces greater challenges in learning new knowledge. Surprisingly, ROME maintains Reliability and Generality almost unchanged from the original model on two datasets, while having a high locality (more than 90), which suggests that it performs little knowledge updating (As the injection of new knowledge will usually have an impact on locality), likely because that it can only edit a small number of parameters, coupled with the fact that our original model is full of old knowledge, which obstructs the effect of the causal tracing mechanism

Dataset	Editor	LLAMA2-7B			LLAMA-7B		
		Reliability	Generality	Locality	Reliability	Generality	Locality
ZsRE	Original model	43.81	43.26	/	43.35	42.92	/
	LoRA	43.22	42.40	70.91	46.90	45.98	75.99
	DFT$_{\text{LoRA}}$	**48.73**	**48.14**	**76.23**	**49.95**	**49.22**	**78.54**
	FT-c	49.23	47.12	67.48	47.54	45.60	68.25
	Full-FT	81.23	74.96	70.62	70.83	66.80	65.37
	ROME	43.67	42.84	**93.85**	44.36	43.23	**99.51**
	MEMIT	83.78	79.35	70.61	78.21	77.74	69.23
	DFT$_{\text{FT}}$	**88.32**	**83.63**	74.44	**85.55**	**84.32**	**75.42**
CounterFact	Original model	18.68	17.06	/	21.72	17.96	/
	LoRA	30.67	23.33	40.19	27.63	21.32	39.86
	DFT$_{\text{LoRA}}$	**35.33**	**31.42**	**48.42**	**34.34**	**28.22**	**49.42**
	FT-c	29.51	19.77	19.52	26.72	17.88	20.10
	Full-FT	66.23	44.32	28.62	32.34	32.41	32.52
	ROME	18.61	17.43	**93.79**	21.99	19.32	**92.32**
	MEMIT	62.16	37.62	22.23	**57.12**	31.62	25.92
	DFT$_{\text{FT}}$	78.53	52.42	36.39	63.51	**45.44**	39.13

Table 1: Results on three metrics of the two datasets based on LLAMA2-7B and LLAMA-7B.

in ROME. It is worth noting that full fine-tuning is much more capable of learning new knowledge than LoRA, as LoRA focuses on training a limited subset of parameters within the attention structure, while the majority of factual knowledge is encoded in the MLP layers.

5.5 Forgetting and Learning with LoRA

In the above setting of experiments, the method we adopt is to perform old knowledge forgetting and new knowledge learning based on full (or LoRA) fine-tuning at the same time. We define this LORA process as follows:

$$\theta_{\triangle}^{old} = \text{LoRA}\{\theta, \text{K}_{\text{old}}\} - \theta, \tag{8}$$

$$\theta_{\triangle}^{new} = \text{LoRA}\{\theta, \text{K}_{\text{new}}\} - \theta \tag{9}$$

$$\theta_{\triangle}^{core} = \text{f}\{\theta_{\triangle}^{\text{new}}, \theta_{\triangle}^{\text{old}}\} \tag{10}$$

$$\theta^{*} = \theta + \lambda\theta_{\triangle}^{core} \tag{11}$$

where LoRA is the operation of supervised fine-tuning by LoRA . θ^{*} is noted as the parameters of the edited model $f_{\theta^{*}}$ which has completed the knowledge updating.

The results are shown in Table 1. The results support that the method of updating with LoRA surpassed the method of updating using full fine-tuning in some cases , which may be owing to the parameter-efficiency of LoRA-based old knowledge forgetting.

Editor	Metric		
	Reliability	Generality	Locality
Original model	28.02	27.95	/
LoRA	29.32	29.31	77.32
DFT$_{\text{LoRA}}$	**30.38**	**31.02**	**79.64**
Full-FT	44.32	43.72	63.94
DFT$_{\text{FT}}$	**45.83**	**44.64**	**72.15**

Table 2: Results on three metrics of the zsRE dataset based on BLOOM-7B.

After conducting experiments, we empirically discovered that utilizing the technique of updating through the updating of LoRA parameters can approximate the effect achieved by updating the parameters during full fine-tuning.

We speculate that this might be due to knowledge being encoded distributively across multiple parameters, and LoRA alters the patterns and relationships associated with the old knowledge stored in the attention structure (i.e., an implicit knowledge representation), which facilitates the new knowledge updating. This finding holds significant value due to the considerable reduction in time and computational costs associated with LoRA compared to full fine-tuning.

5.6 Adaptability Testing

To further verify the adaptability of the method, we conducted experiments on zsRE based on BLOOM-

7B and maintained the same experimental settings as the above. The results are shown in Table 2. We could find that DFT still performs well. Notably, although the Reliability and Generality remain roughly stable, the locality is significantly improved, which means that our method could inject new knowledge into the LLM with less cost (As changes to model parameters will inevitably affect the model's locality), demonstrating the necessity and effectiveness of forgetting old knowledge.

5.7 Time Testing

In order to evaluate the efficiency of our proposed DFT method, we calculated the editing time of several different knowledge updating and model editing methods for different numbers of edits. Taking LLAMA2-7B as an example. The results are shown in the Table 3.

We can find that the time consumed by the fine-tuning based method is significantly less than that of the locate-based method. This is because the locate-based method highly relies on the location of neurons and parameters, which increases the complexity and time of editing. Furthermore, since ROME can only edit a single piece of data at a time, while other methods can edit in batches, ROME is less efficient. Compared with other fine-tuning based methods, FT-c can be optimized faster with its norm constraint. The DFT we proposed is a multi-stage knowledge updating method that multiple backward passes and comparisons , but only a few parameters require updating , as it takes about two times longer than Full-FT, but is still very fast and convenient. We can further accelerate supervised fine-tuning by deepspeed or other approaches.

5.8 Parametric Analysis of Updating Knowledge

The knowledge updating method we proposed heavily relies on pinpointing the location of knowledge. From the perspective of interpretability, we precisely identified the parameters containing the knowledge and updated them and further analyzed the parameter distribution and changes within the LLMs. we find that the parameter changes of the MLP layers are more significant than attention layers. This may be knowledge is generally stored in the MLP layers.

6 Case Study

To further illustrate the effectiveness of the proposed method, we present a case study on the results of the knowledge updating by the original model, only forgetting old knowledge and performing DFT. We selected some cases in the experiment of llama2-7B on zsRE dataset, noting that the hyper-parameters λ of the rate of forgetting is set to 0.3. Table 4 shows the results during different knowledge updating stages. From the first example and second example, we can find that model begins to output some irrelevant content after performing the forgetting operation, indicating that it gradually forgets the old knowledge. In example 4, the forgetting operation failed to assist the model in forgetting old knowledge, but it still completed knowledge updating with the help of DFT. However, sometimes there are some bad cases, such as example 3, where the model never learned new knowledge, which shows that our method has certain limitations and could be improved.

7 Conclusion

In this paper, we propose a new paradigm of knowledge updating during supervised fine-tuning called DFT, which is based on parametric arithmetic to locating old knowledge and learn new knowledge for eliminating contradictions between old and new knowledge. The experiments on zsRE and COUNTERFACT datasets show that our method surpasses other baselines in most cases. Simultaneously we find that updating old knowledge by subtracting the parameters of LoRA can achieve the similar effect of subtracting the parameters of full fine-tuning, which is inspiring. We will further investigate the updating of knowledge.

8 Limitations

In this work, the proposed DFT paradigm, although it improves the effectiveness of the fine-tuning methods for updating the knowledge of large language models, adds extra computation due to extra multiple backward passes process.

References

Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, et al. 2020. Language models are few-shot learners. *Advances in neural information processing systems*, 33:1877–1901.

Editor	1 edit		10 edits		100 edits	
	zsRE	COUNTERFACT	zsRE	COUNTERFACT	zsRE	COUNTERFACT
FT-c	0.59(s)	0.55(s)	5.73(s)	5.57(s)	56.13(s)	55.12(s)
ROME	2.76(s)	2.46(s)	27.9(s)	24.32(s)	285.23(s)	242.21(s)
MEMIT	612(s)	606(s)	6231(s)	6193(s)	61831(s)	61631(s)
Full-FT	0.78(s)	0.74(s)	7.92(s)	7.43(s)	76.72(s)	75.11(s)
DFT$_{\mathrm{FT}}$	**1.49(s)**	**1.42(s)**	**11.98(s)**	**11.21(s)**	**120.92(s)**	**118.87(s)**

Table 3: Editing time for 1 edit, 10 edits, 100 edits of the two dataset based on LLAMA2-7B.Run ROME with FastEdit.Run MEMIT with EasyEdit

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

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

Damai Dai, Wenbin Jiang, Qingxiu Dong, Yajuan Lyu, and Zhifang Sui. 2023. Neural knowledge bank for pretrained transformers. In *CCF International Conference on Natural Language Processing and Chinese Computing*, pages 772–783. Springer.

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.

Qingxiu Dong, Damai Dai, Yifan Song, Jingjing Xu, Zhifang Sui, and Lei Li. 2022. Calibrating factual knowledge in pretrained language models. In *Findings of the Association for Computational Linguistics: EMNLP 2022*, pages 5937–5947.

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.

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.

Anshita Gupta, Debanjan Mondal, Akshay Krishna Sheshadri, Wenlong Zhao, Xiang Lorraine Li, Sarah Wiegreffe, and Niket Tandon. 2023. Editing commonsense knowledge in gpt. *arXiv preprint arXiv:2305.14956*.

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

Edward J Hu, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, Weizhu Chen, et al. 2021. Lora: Low-rank adaptation of large language models. In *International Conference on Learning Representations*.

Zeyu Huang, Yikang Shen, Xiaofeng Zhang, Jie Zhou, Wenge Rong, and Zhang Xiong. 2022. Transformer-patcher: One mistake worth one neuron. In *The Eleventh International Conference on Learning Representations*.

Gabriel Ilharco, Marco Tulio Ribeiro, Mitchell Wortsman, Ludwig Schmidt, Hannaneh Hajishirzi, and Ali Farhadi. 2022. Editing models with task arithmetic. In *The Eleventh International Conference on Learning Representations*.

Kyungjae Lee, Wookje Han, Seung-won Hwang, Hwaran Lee, Joonsuk Park, and Sang-Woo Lee. 2022. Plug-and-play adaptation for continuously-updated qa. *arXiv preprint arXiv:2204.12785*.

Omer Levy, Minjoon Seo, Eunsol Choi, and Luke Zettlemoyer. 2017. Zero-shot relation extraction via reading comprehension. *arXiv preprint arXiv:1706.04115*.

Daliang Li, Ankit Singh Rawat, Manzil Zaheer, Xin Wang, Michal Lukasik, Andreas Veit, Felix Yu, and Sanjiv Kumar. 2022. Large language models with controllable working memory. *arXiv preprint arXiv:2211.05110*.

Xiang Lisa Li and Percy Liang. 2021. Prefix-tuning: Optimizing continuous prompts for generation. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 4582–4597.

Aman Madaan, Niket Tandon, Peter Clark, and Yiming Yang. 2022. Memory-assisted prompt editing to improve gpt-3 after deployment. In *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing*, pages 2833–2861.

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

(1) Prompt: What artist created Call the Doctor?
Original answer: Riders in the Sky **Target answer:** The X-Files
Original model: Riders in the Sky
Original model + Old knowledge forgetting: Doctor Who
Original model + DFT: The X-Files
(2) Prompt: What university did Watts Humphrey take part in?
Original answer: Trinity College **Target answer:** University of Michigan
Original model: Trinity College
Original model + Old knowledge forgetting: The Wire
Original model + DFT: University of Michigan
(3) Prompt: What role does Denny Herzig play in football?
Original answer: midfielder **Target answer:** winger
Original model: midfielder
Original model + Old knowledge forgetting: midfielder
Original model + DFT: goalkeeper
(4) Prompt: Which family does Ramalinaceae belong to?
Original answer: Ramales **Target answer:** Lamiinae
Original model: Ramales
Original model + Old knowledge forgetting: Ramales
Original model + DFT: Lamiinae
(5) Prompt: Who's the architect of Toodyay Fire Station?
Original answer: Wong Tung and Partners **Target answer:** Wyndham Lewis
Original model: Wong Tung and Partners
Original model + Old knowledge forgetting: Wong Tung
Original model + DFT: Wyndham Lewis

Table 4: Examples of knowledge updating results during different stages.

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

Kevin Meng, Arnab Sen Sharma, Alex J Andonian, Yonatan Belinkov, and David Bau. 2022b. Mass-editing memory in a transformer. In *The Eleventh International Conference on Learning Representations*.

Eric Mitchell, Charles Lin, Antoine Bosselut, Chelsea Finn, and Christopher D Manning. 2021. Fast model editing at scale. In *International Conference on Learning Representations*.

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.

Shikhar Murty, Christopher D Manning, Scott Lundberg, and Marco Tulio Ribeiro. 2022. Fixing model bugs with natural language patches. In *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing*, pages 11600–11613.

Long Ouyang, Jeffrey Wu, Xu Jiang, Diogo Almeida, Carroll Wainwright, Pamela Mishkin, Chong Zhang, Sandhini Agarwal, Katarina Slama, Alex Ray, et al. 2022. Training language models to follow instructions with human feedback. *Advances in Neural Information Processing Systems*, 35:27730–27744.

Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J Liu. 2020. Exploring the limits of transfer learning with a unified text-to-text transformer. *The Journal of Machine Learning Research*, 21(1):5485–5551.

Vikas Raunak and Arul Menezes. 2022. Rank-one editing of encoder-decoder models. *arXiv preprint arXiv:2211.13317*.

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

Anton Sinitsin, Vsevolod Plokhotnyuk, Dmitry Pyrkin, Sergei Popov, and Artem Babenko. 2019. Editable neural networks. In *International Conference on Learning Representations*.

Peng Wang, Ningyu Zhang, Xin Xie, Yunzhi Yao, Bozhong Tian, Mengru Wang, Zekun Xi, Siyuan

Cheng, Kangwei Liu, Guozhou Zheng, et al. 2023a. Easyedit: An easy-to-use knowledge editing framework for large language models. *arXiv preprint arXiv:2308.07269*.

Song Wang, Yaochen Zhu, Haochen Liu, Zaiyi Zheng, Chen Chen, et al. 2023b. Knowledge editing for large language models: A survey. *arXiv preprint arXiv:2310.16218*.

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. *arXiv preprint arXiv:2305.13172*.

Charles Yu, Sullam Jeoung, Anish Kasi, Pengfei Yu, and Heng Ji. 2023. Unlearning bias in language models by partitioning gradients. In *Findings of the Association for Computational Linguistics: ACL 2023*, pages 6032–6048.

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

Ningyu Zhang, Jintian Zhang, Xiaohan Wang, Honghao Gui, Kangwei Liu, Yinuo Jiang, Xiang Chen, Shengyu Mao, Shuofei Qiao, Yuqi Zhu, Zhen Bi, Jing Chen, Xiaozhuan Liang, Yixin Ou, Runnan Fang, Zekun Xi, Xin Xu, Lei Li, Peng Wang, Mengru Wang, Yunzhi Yao, Bozhong Tian, Yin Fang, Guozhou Zheng, and Huajun Chen. 2023. Knowlm technical report.

Qingru Zhang, Minshuo Chen, Alexander Bukharin, Pengcheng He, Yu Cheng, Weizhu Chen, and Tuo Zhao. 2022. Adaptive budget allocation for parameter-efficient fine-tuning. In *The Eleventh International Conference on Learning Representations*.

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? *arXiv preprint arXiv:2305.12740*.

Chen Zhu, Ankit Singh Rawat, Manzil Zaheer, Srinadh Bhojanapalli, Daliang Li, Felix Yu, and Sanjiv Kumar. 2020. Modifying memories in transformer models. *arXiv preprint arXiv:2012.00363*.

A Appendix

A.1 Datasets and Examples

We will illustrate the datasets we used in more detail. ZsRE is a Question Answering (QA) dataset that utilizes question rephrasings generated by back-translation as the equivalence neighborhood. COUNTERFACT is a more challenging dataset with counterfactual data. We take the eval and edit sets of which there are 19,085 and 10,000 pieces of data respectively.

The following is a sample of the ZsRE dataset:

{**"subject":** "Watts Humphrey", **"src":** "What university did Watts Humphrey attend?", **"pred":** **"Trinity College"**, **"rephrase":** "What university did Watts Humphrey take part in?", **"alt": "University of Michigan"**, **"answers":** ["Illinois Institute of Technology"], **"loc":** "nq question: who played desmond doss father in hacksaw ridge", **"loc-ans":** "Hugo Weaving", **"cond":** "Trinity College » University of Michigan || What university did Watts Humphrey attend?"}

It represents that for prompt "What university did Watts Humphrey attend?", modifying the old knowledge "Trinity College" into the new knowledge "University of Michigan". Meanwhile, "rephrase" is used to evaluate the model's Generalization metric, and "loc" is used to evaluate the model's Locality metric.

Furthermore, we can find that old knowledge and new knowledge have some correlation, they keep the same questions with different answers. We keep them in the same format to ensure the training effect. To facilitate our supervised fine-tuning training, we divide the datasets into two parts of old knowledge and new knowledge, and convert them into an instruction fine-tuning format, an example as follows:

The old knowledge:

{**"instruction"**: "What university did Watts Humphrey attend?", **"input"**: "", **"output"**: "Trinity College" }

The new knowledge:

{**"instruction"**: "What university did Watts Humphrey attend?", **"input"**: "", **"output"**: "University of Michigan" }

What calls for special attention is that the two datasets used in our experiments are both counterfactual datasets, in which the old knowledge is correct knowledge in the real world, and the new knowledge (target knowledge) is wrong knowledge in the real world, so the labels of old knowledge and new knowledge in these datasets are given artificially and have nothing to do with time and correctness in the real world. They are only used to measure whether the model can accurately modify the knowledge. Since the new knowledge is wrong knowledge in the real world, it can ensure that the original LLM has not learned it before, thus avoiding the problem of being unable to determine whether the new knowledge output by the LLM is learned from the data or possessed by itself.

A.2 Implementation Details of Experiments

Here we will introduce more completion details and settings of experiments. First, we used LLAMA2-7B and LLAMA-7B as the base models, and then we trained the base model on the old knowledge for 3 epochs by full fine-tuning to simulate an original model that has fully learned old knowledge for our experiments. This makes the forgetting operation more reasonable and effective, and at the same time tries to avoid the problem of being unable to determine whether the new knowledge output by the LLM is learned from the data or commanded by itself as mentioned above.

A.2.1 DFT

For the experiments of zsRE on LLAMA2-7B, the hyperparameters λ set in DFT_{LoRA} is 0.7, while 3 in $\text{DFT}_{\text{LoRA}-\text{FT}}$ and 0.3 in DFT_{FT}. Similarly, for COUNTERFACT dataset, the hyperparameters λ set in DFT_{LoRA} is 1.5, while 3 in $\text{DFT}_{\text{LoRA}-\text{FT}}$ and 0.1 in DFT_{FT}. The learning rate, epochs for all above experiments are 5e-5 and 3, then batch-size is 4 and gradient-accumulation-steps is 4.

For the experiments of zsRE on LLAMA-7B, the hyperparameters λ set in DFT_{LoRA} is 0.7, while 2 in $\text{DFT}_{\text{LoRA}-\text{FT}}$ and 0.3 in DFT_{FT}. For COUNTERFACT dataset, the hyperparameters λ set in DFT_{LoRA} is 1, while 3 in $\text{DFT}_{\text{LoRA}-\text{FT}}$ and 0.05 in DFT_{FT}. The epochs for all above experiments are 3, then batch-size is 4 and gradient-accumulation-steps is 4. The learning rate for zsRE experiments is 5e-5 while 1e-5 in COUNTERFACT experiments.

For the experiments of zsRE on BLOOM-7B, the hyperparameters λ set in DFT_{LoRA} is 0.1, while 3 in $\text{DFT}_{\text{LoRA}-\text{FT}}$ and 0.2 in DFT_{FT}. The learning rate, epochs for all the experiments are 5e-5 and 3, then batch-size is 4 and gradient-accumulation-steps is 4.

When we used the Deepspeed, we set 4 processes and zero-stage is 2.

Dataset	Editor	
	Original model	Original model+DFT
MMLU-college-chemistry	24	40
MMLU-college-mathematics	29	29
MMLU-management	16.50	31.80
MMLU-computer-security	21	21
MMLU-macroeconomics	32.31	35.19
MMLU-college-physics	19.61	21.77
MMLU-astronomy	30.92	32.36
MMLU-professional-law	26.47	26.47
MMLU-college-medicine	24.28	28.14

Table 5: Results on accuracy of the MMLU dataset based on LLAMA2-7B.

A.2.2 Full-FT and LoRA

Full-FT and LoRA refer to knowledge updating by full fine-tuning and LoRA fine-tuning in our experiments. We adopted experimental settings similar to DFT as mentioned above. The difference is that these two do not forget the old knowledge. Full-FT and LoRA also use the instruction fine-tuning data mentioned in for supervised fine-tuning training. Instruction fine-tuning can make it generate answers to prompts better.

A.2.3 FT-c

Knowledge updating of FT-c is executed at layer 21, where optimization proceeds for 5 steps with a learning rate of 5e-5. And the batch-size is 1.

A.2.4 ROME with FastEdit

Knowledge updating of ROME is executed at layer 5, where optimization proceeds for 25 steps with a learning rate of 5e-3. And the weight-decay is 1e-3, the kl-factor is 0.0625. Covariance statistics are collected in float32 on Wikitext using a sample size of 100,000.

A.2.5 MEMIT with EasyEdit

Knowledge updating of ROME is executed at layer n = [4, 5, 6, 7, 8], where optimization proceeds for 25 steps with a learning rate of 5e-2. The batch is the 19,085 (or 10,000). And the weight-decay is 1e-3, the kl-factor is 0.0625. Covariance statistics are collected in float32 on Wikitext using a sample size of 100,000.

A.3 Impact to Other Capabilities within LLMs Testing

Here we test the impact of updating old knowledge to new knowledge over the original model on other capabilities of the model (such as mathematical abilities). Specifically, we evaluated changes in the model's evaluated changes in the model's comprehensive examining capabilities on MMLU. Taking LLAMA2-7B as an example. The results are shown in the Table 5. "Original model+DFT" refers to the original model with our DFT method. The hyper-parameters λ of the rate of forgetting is set to 0.3. And the metric of evaluation in experiments is mainly "Accuracy". Experiments show that our method has little impact. Interestingly, the model's performance on "college-chemistry" and "college-medicine" has been significantly improved after completing the knowledge update. This may be because the dataset contains relevant knowledge.

A.4 Interpretability of Parametric Arithmetic

Recently, Parametric Arithmetic has become a common method for parameter fine-tuning because of its operability and adaptability. previous work (Ilharco et al., 2022) has conducted experimental research on parameter parametric arithmetic and verified that subtracting the parameters obtained by fine-tuning can achieve a decrease in indicators on a dataset. On the contrary, adding parameters obtained by fine-tuning can endow the model with capabilities or achieve multi-task learning. Our experimental results demonstrate that directly modifying the parameters associated with the old knowledge can also achieve good knowledge updating.

The parametric arithmetic in this paper mainly focuses on core parameter of knowledge . If subtract parameters , then the loss function moves in the opposite direction to the direction of gradient descent to accumulate errors. This is essentially similar to gradient ascent and loss maximization, a method commonly used in machine unlearning, through the maximum of loss to accumulate errors and damage the performance of the model on a

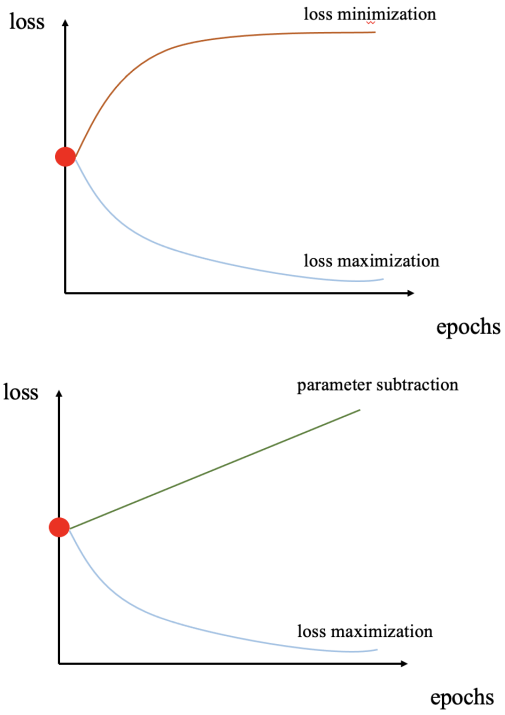

Figure 3: Loss changes of loss maximization and parameter subtraction.

dataset. Their relationship is roughly shown in the figure 3. We do not directly use loss maximization because compared with it, the method of updating core parameters is more stable and controllable, which can avoid affecting other irrelevant knowledge as much as possible.

A.5 Prospects and Application Scenarios

In an era when LLM's research and application are becoming more and more popular, knowledge update is gaining its attention as a technology for updating the internal knowledge within the LLM. Knowledge updating is closely related to some research fields such as continual learning and machine unlearning. The purpose of knowledge updating is to correct old or wrong knowledge, while continual learning hopes to not forget old knowledge while the LLM continues to learn new knowledge. The aim of machine unlearning is to let the LLM forget harmful or wrong knowledge.

We believe that our method of old knowledge updating has a wide range of application scenarios, such as harmful knowledge updating, copyright content updating , etc.