Model Editing by Pure Fine-Tuning

Govind Gangadhar and **Karl Stratos**
Department of Computer Science
Rutgers University
{govind.gangadhar, karl.stratos}@rutgers.edu

Abstract

Fine-tuning is dismissed as not effective for model editing due to its poor performance compared to more specialized methods. However, fine-tuning is simple, agnostic to the architectural details of the model being edited, and able to leverage ongoing advances in standard training methods (e.g., PEFT), making it an appealing choice for a model editor. In this work, we show that pure fine-tuning can be a viable approach to model editing. We propose a slight modification of naive fine-tuning with two key ingredients. First, we optimize the conditional likelihood rather than the full likelihood. Second, we augment the data with random paraphrases and facts to encourage generalization and locality. Our experiments on ZsRE and COUNTERFACT show that this simple modification allows fine-tuning to often match or outperform specialized editors in the edit score.

1 Introduction

Model editing is a promising approach to combating incorrect or otherwise unwanted knowledge in LLMs. It assumes a set of edits that assert desired information and aims to alter the model so that it not only succeeds in memorizing the edits (efficacy), but also applying the asserted information to new prompts (generalization), *without* affecting inference that should remain unchanged (locality). There is clearly a trade-off between these metrics. At one extreme, the model can achieve high efficacy by memorizing the edits, but it will fail in generalization and locality. At another extreme, the model can achieve high locality by staying unmodified, but it will fail in efficacy and generalization.

A straightforward approach of fine-tuning the model on the requested edits is well known to perform poorly, especially in locality. This motivated researchers to dismiss fine-tuning as an ineffective solution to model editing and develop very specialized methods such as MEND (Mitchell et al., 2022), ROME (Meng et al., 2022), and MEMIT (Meng et al., 2023). A central theme of these methods is "minimally invasive modification", typically involving a careful selection of weights and rank-one updates. While technically interesting, they require a suite of assumptions which may not be satisfied in other contexts (e.g., with a different model architecture). In contrast, fine-tuning is simple, completely agnostic to the details of the model, and can take advantage of advances in standard training such as parameter-efficient fine-tuning (PEFT) techniques.

Editor	Standard model?	Batched edits?	No extra training?	Effective edits?
Naive fine-tuning	✓	✓	✓	✗
MEND	✗	✓	✗	✓
ROME	✗	✗	✓	✓
MEMIT	✗	✓	✓	✓
Our fine-tuning	✓	✓	✗	✓

Table 1: Conceptual comparisons of model editors.

In this work, we show that fine-tuning can be a viable approach to model editing, focusing primarily on improving the edit score. We shift the focus from models and algorithms to training objectives and data augmentation. See Table 1 for conceptual comparisons between our approach and other editors. Our method is a slight variation on naive fine-tuning with two small but important differences. First, in line with the theme of minimal modification, we optimize the *conditional* likelihood (i.e., mask all tokens except the edited target). Second, we augment the data with self-induced paraphrases of the prompt and arbitrary unrelated facts to encourage generalization and locality. This simple change is sufficient for pure fine-tuning to match or outperform specialized editors in mass-editing and also perform respectably in single-editing.

2 Task

We consider the task of mass-editing. Let $\mathcal{V}$ denote the vocabulary. A **fact** is a sentence $x \in \mathcal{V}^T$ that expresses a subject-relation-object triple (s, r, o) in natural language. We follow the convention in the model editing literature and assume that the object $o \in \mathcal{V}^m$ is the last m tokens of x. The prefix $\pi = (x_1 \dots x_{m-1})$ expresses (s, r) and is denoted as the **prompt**. Let $\mathcal{E}$ denote a set of facts to enforce (i.e., requested edits). Our goal is to edit a pretrained language model so that it upholds the relations expressed in $\mathcal{E}$ without changing its behavior on other facts. In ZsRE, $\mathcal{E}$ consists of 10,000 factual statements (e.g., "The artwork Gideon's Way was by who? John Creasey"). In COUNTERFACT, $\mathcal{E}$ consists of 10,000 counterfactual statements (e.g., "TextEdit, a product of Nintendo").

An editor is evaluated by three competing metrics: efficacy, generalization, and locality. Let p_θ denote a language model edited on $\mathcal{E}$. In ZsRE, efficacy is the accuracy of $o = \arg\max_y p_\theta(y|\pi)$ for $(\pi, o) \in \mathcal{E}$; generalization is the accuracy of $o = \arg\max_y p_\theta(y|\pi_{\text{par}})$ where π_{par} is some paraphrase of π; locality is the accuracy of $o_{\text{unrel}} = \arg\max_y p_\theta(y|\pi_{\text{unrel}})$ where $(\pi_{\text{unrel}}, o_{\text{unrel}})$ is an unrelated fact. In COUNTERFACT, efficacy and generalization measure the accuracy of $p_\theta(o|\pi) > p_\theta(o_{\text{true}}|\pi)$ and $p_\theta(o|\pi_{\text{par}}) > p_\theta(o_{\text{true}}|\pi_{\text{par}})$ where o_{true} is a true object for $(\pi, o) \in \mathcal{E}$ (e.g., "Apple" in "TextEdit, a product of Nintendo"). Locality measures the accuracy of $p_\theta(o_{\text{true}}|\pi_{\text{nb}}) > p_\theta(o|\pi_{\text{nb}})$ where π_{nb} is a "neighbhorhood" prompt whose target must remain unchanged from o_{true} (e.g., "Macintosh File System, a product of" $\rightarrow$ "Apple"). In both datasets, the final **edit score** is the harmonic mean of efficacy, generalization, and locality.

It is well known that naive fine-tuning, namely just optimizing

$$\min_\theta \; -\sum_{x \in \mathcal{E}} \log p_\theta(x) \tag{1}$$

results in a poor editing score. The main reason is that while it improves efficacy and possibly even generalization, it harms locality by changing the model's predictions on unrelated or neighborhood prompts. This led to the development of highly specialized editing methods, with the implicit assumption that fine-tuning is not effective for model editing.

3 Method

We propose a purely fine-tuning-based method that achieves a competitive editing score. Our objective is a slight variation of (1):

$$\min_\theta \; -\sum_{(\pi,o) \in \mathcal{E} \cup \mathcal{P} \cup \mathcal{R}} \log p_\theta(o|\pi) \tag{2}$$

There are two small but important differences between (1) and (2). First, we optimize the *conditional* likelihood of the edit target $o|\pi$ rather than the full likelihood. Our motivation is to make the training more focused (for efficacy and generalization) and minimize the damage caused by fine-tuning (for locality).

Second, we fine-tune not only on the requested edits $\mathcal{E}$ but also additional facts in $\mathcal{P}$ and $\mathcal{N}$ to promote generalization and locality. $\mathcal{P}$ is pseudo-paraphrases of the prompts in $\mathcal{E}$ generated by appending text generated by the (unedited) model. More specifically, for each $(\pi, o) \in \mathcal{E}$ we generate $((z, \pi), o) \in \mathcal{P}$ where $z \sim p_\theta$. $\mathcal{R}$ is a set of facts that should not be altered by editing on $\mathcal{E}$. There are many ways to obtain $\mathcal{R}$, but we find taking *random* facts from the training split to be simple and effective. We filter $x \in \mathcal{R}$ to ensure that x does not include the subject-relation-object triple used in evaluation.

We emphasize that the augmented sets $\mathcal{P}$ and $\mathcal{R}$ do not use the evaluation facts (otherwise it is trivial). Given the requested edits $\mathcal{E}$, we perform the data augmentation and finetune the model according to (2).

Contrastive learning. Given that our goal in COUNTERFACT is to achieve $p_\theta(o|\pi) > p_\theta(o_{\text{true}}|\pi)$ where o is the new object to enforce over the original o_{true} for a given prompt π, it is natural to consider a contrastive objective. We experimented with DPO (Rafailov et al., 2023) where we frame $o|\pi$ as the "preferred" response over $o_{\text{true}}|\pi$, optimizing the DPO loss jointly with (2). However, we found that prompt masking and data augmentation are already effective and do not benefit from contrastive learning.

4 Related Work

We discuss some prominent model editing methods and the different assumptions they require to highlight the simplicity of our fine-tuning approach; we refer to Zhang et al. (2024) for an in-depth survey.

MEND (Mitchell et al., 2022) is a meta-learning method (Sinitsin et al., 2020; De Cao et al., 2021) that predicts the change in the gradient. Similar to our approach, it also involves an explicit training

Editor	ZsRE Score	ZsRE Efficacy	ZsRE Generalization	ZsRE Locality	CounterFact Score	CounterFact Efficacy	CounterFact Generalization	CounterFact Locality
— (original GPT-J)	26.4	26.4 (0.6)	25.8 (0.5)	27.0 (0.5)	22.4	15.2 (0.7)	17.7 (0.6)	83.5 (0.5)
FT-W (21st layer w/ weight decay)	42.1	69.6 (0.6)	64.8 (0.6)	24.1 (0.6)	67.6	99.4 (0.1)	77.0 (0.7)	46.9 (0.6)
MEND	20.0	19.4 (0.5)	18.6 (0.5)	22.4 (0.5)	23.1	15.7 (0.7)	18.5 (0.7)	83.0 (0.5)
ROME	2.6	21.0 (0.7)	19.6 (0.7)	0.9 (0.1)	50.3	50.2 (1.0)	50.4 (0.8)	50.2 (0.6)
MEMIT	50.7	96.7 (0.3)	89.7 (0.5)	26.6 (0.5)	85.8	98.9 (0.2)	88.6 (0.5)	**73.7** (0.5)
FT	44.8	**99.9** (0.03)	**98.9** (0.2)	21.4 (0.5)	52.8	79.6 (0.8)	58.5 (0.8)	36.8 (0.7)
FT (21st layer)	42.9	**99.9** (0.0)	87.4 (0.5)	20.5 (0.5)	60.5	99.9 (0.04)	63.3 (0.8)	42.0 (0.6)
FT + Mask	58.3	97.6 (0.3)	91.7 (0.5)	32.9 (0.6)	54.3	97.1 (0.3)	62.1 (0.8)	34.7 (0.6)
FT + Mask + Para	56.1	**99.9** (0.0)	98.7 (0.2)	29.9 (0.5)	63.7	**100.0** (0.0)	92.5 (0.4)	38.0 (0.6)
FT + Mask + Para + Rand	**62.0**	**99.9** (0.0)	97.0 (0.3)	**35.6** (0.6)	**86.5**	98.8 (0.2)	**93.6** (0.4)	72.0 (0.6)
FT + Mask + Para + Rand + DPO	—	—	—	—	85.5	98.8 (0.2)	93.4 (0.4)	70.1 (0.6)

Table 2: Mass-editing results on ZsRE and COUNTERFACT (10,000 edits each) with GPT-J. The results of the specialized methods (FT-W, MEND, ROME, MEMIT) are from Meng et al. (2023). FT denotes naive fine-tuning on the requested edits; FT (21st layer) denotes FT only on the 21st layer. "+ Mask" means we mask the prompt. "+ Para" means we augment the data with model-generated paraphrases of the requested edits. "+ Rand" means we augment the data with *random* facts from the training split (not overlapping with any evaluation facts). "+ DPO" means we additionally optimize the DPO loss term using the changed target as the preferred response over the original target (which is supplied in COUNTERFACT).

Editor	Score	Efficacy	Generalization	Locality
— (GPT-J)	37.4	34.4 (1.7)	34.5 (1.5)	45.3 (0.9)
ROME	35.0	39.8 (2.2)	25.5 (1.4)	**46.9** (0.8)
MEMIT	67.3	99.2 (0.3)	80.2 (1.2)	45.3 (0.8)
FT	55.8	68.1 (1.8)	60.4 (1.7)	44.5 (0.8)
FT + M + P + R	**68.5**	**99.6** (0.2)	**84.6** (1.1)	45.8 (0.9)

Table 3: Mass-editing results on WikiRecent (1,266 edits).

stage (on the training split of ZsRE). It uses a special rank-one decomposition of the gradient for parameter efficiency. In our case, we achieve parameter efficiency without special considerations simply by leveraging PEFT methods. MEND works poorly for mass-editing.

ROME (Meng et al., 2022) is a locate-then-edit method (Geva et al., 2021; Dai et al., 2022) that first uses causal tracing to identify the feedforward layer to change, and then applies a rank-one update to the layer's weight matrix. While it does not involve an explicit training stage, it does require an explicit knowledge of the subject s in the edit and its vector representation (which is heuristically induced). It also relies on Wikipedia to estimate the covariance matrix of the subject embeddings (needed in the rank-one update). ROME works well on single-editing but poorly on mass-editing. MEMIT (Meng et al., 2023) extends ROME to multiple edits by carefully spreading updates to multiple layers. Like ROME, it requires layer specification, subject embeddings, and covariance statistics obtained from Wikipedia. MEMIT is very effective for mass-editing and is our strongest baseline.

IKE (Zheng et al., 2023) proposes in-context learning for model editing. While it similarly uses demonstrations, it is critically limited to single-editing. It also requires an initial model that is capable of effective in-context learning. Consequently, IKE focuses on single-editing with large-scale LLMs, making their results not comparable to ours.

We define our scope as achieving a competitive edit score through pure fine-tuning. In particular, we do not focus on preserving general capabilities of the model being edited. Existing methods consider basic metrics such as n-gram repetition, but recent work shows that performance on a variety of downstream tasks drops to zero under any editor (Gu et al., 2024). We leave preserving the general capabilities of an LLM with a fine-tuning editor as the next step of our work.

5 Experiments

5.1 Mass-Editing

Our main results are on mass-editing with on ZsRE and COUNTERFACT, following the same setting in Meng et al. (2023). Each dataset provides 10,000 requested edits (i.e., $\mathcal{E}$). As described in Section 3, we augment the fine-tuning data with pseudo-paraphrases of the edit prompts $\mathcal{P}$ for generalization supervision and random facts $\mathcal{R}$ for locality supervision. More specifically, for each edit in $\mathcal{E}$ we generate 15 paraphrases and take 20

Editor	ZsRE Score	ZsRE Efficacy	ZsRE Generalization	ZsRE Locality	COUNTERFACT Score	COUNTERFACT Efficacy	COUNTERFACT Generalization	COUNTERFACT Locality
— (original GPT-2 XL)	22.5	22.2 (0.5)	21.3 (0.5)	24.2 (0.5)	30.5	22.2 (0.9)	24.7 (0.8)	78.1 (0.6)
FT	45.9	99.6 (0.1)	82.1 (0.1)	23.2 (0.5)	65.1	**100.0** (0.0)	87.9 (0.6)	40.4 (0.7)
FT-L	40.1	92.3 (0.4)	47.2 (0.7)	23.4 (0.5)	66.9	99.1 (0.2)	48.7 (1.0)	70.3 (0.7)
KN	-	-	-	-	35.6	28.7 (1.0)	28.0 (0.9)	72.9 (0.7)
KE	41.8	65.5 (0.6)	61.4 (0.6)	24.9 (0.5)	52.2	84.3 (0.8)	75.4 (0.8)	30.9 (0.7)
MEND	42.9	75.9 (0.5)	65.3 (0.6)	24.1 (0.5)	57.9	99.1 (0.2)	65.4 (0.9)	37.9 (0.7)
ROME	47.9	99.8 (0.0)	88.1 (0.5)	24.2 (0.5)	**89.2**	100 (0.1)	**96.4** (0.3)	**75.4** (0.7)
FT + Mask + Para + Sim	**51.4**	**100.0** (0.0)	**99.9** (0.0)	**26.1** (0.8)	83.1	98.6 (0.3)	87.3 (0.7)	69.0 (0.7)

Table 4: Single-editing results on ZsRE (10,000 edits) and COUNTERFACT (7,500 edits) with GPT-2 XL. "+ Sim" means we include similar facts (measured by Sentence-BERT) instead of random facts for locality supervision.

Editor	Score	Fluency	Consistency
— (original GPT-J)	22.4	622.4 (0.3)	29.4 (0.2)
FT-W	67.6	293.9 (2.4)	15.9 (0.3)
MEND	20.0	618.4 (0.3)	31.1 (0.2)
ROME	50.3	589.6 (0.5)	3.3 (0.0)
MEMIT	85.8	619.9 (0.3)	40.1 (0.2)
FT	52.8	626.1 (0.4)	31.0 (0.2)
FT + Mask	54.3	563.6 (0.5)	6.1 (0.1)
FT + Mask + Para	63.7	550.7 (0.6)	4.7 (0.1)
FT + Mask + Para + Rand	86.5	352.0 (1.5)	5.2 (0.2)
+ Wikipedia Loss	84.8	609.2 (0.6)	29.2 (0.2)

Table 5: Fluency and consistency with mass-editing on COUNTERFACT. "+ Wikipedia Loss" means we additionally optimize a language modeling loss on Wikipedia text (Section 5.3).

facts from the training split while ensuring that they do not contain any evaluation facts. Thus the total number of facts we fine-tune on is 360,000. We fine-tune GPT-J (6B) (Wang and Komatsuzaki, 2021) with LoRA (Hu et al., 2022) for computational efficiency. We optimize (2). The training takes around 2-3 hours on 8 GPUs.[1]

We give the results in Table 2 where the row "FT + Mask + Para + Rand" corresponds to our final method. We show various ablations to illustrate the effect of individual components. We can make the following observations. First, the proposed prompt masking in (2) (FT + Mask) improves the performance of vanilla fine-tuning (FT), supporting the intuition that minimizing the model update is beneficial. In fact, on ZsRE this already outperforms MEMIT without any data augmentation. Second, augmenting the data with paraphrased prompts substantially improves generalization (FT + Mask + Para). Third, augmenting the data with random facts further improves locality (FT + Mask + Para + Rand), allowing pure fine-tuning to match the performance of MEMIT on COUNTERFACT. We find that adding the DPO loss on top of the final method does not yield improvements.

[1]The code is available at: `https://github.com/au-revoir/model-editing-ft`.

Editor	Score	Fluency	Consistency
— (original GPT-2 XL)	30.5	626.6 (0.3)	31.9 (0.2)
FT	65.1	607.1 (1.1)	40.5 (0.3)
FT-L	66.9	621.4 (1.0)	37.4 (0.3)
KN	35.6	570.4 (2.3)	30.3 (0.3)
KE	52.2	586.6 (2.1)	31.2 (0.3)
MEND	57.9	624.2 (0.4)	34.8 (0.3)
ROME	89.2	621.9 (0.5)	41.9 (0.3)
FT + Mask + Para + Sim	83.1	557.0 (1.8)	15.2 (0.3)

Table 6: Fluency and consistency with single-editing on COUNTERFACT.

5.1.1 Additional experiments on WikiRecent

To further validate our mass-editing results, we perform additional experiments on a simplified version of the WikiRecent dataset (Cohen et al., 2023). The edit requests are 1,266 recent facts in Wikidata; details are provided in Appendix A. Table 3 shows the results. We again find that while vanilla fine-tuning is not effective, our conditional fine-tuning with random data augmentation is competitive with MEMIT.

5.2 Single-Editing

To compare with existing methods developed for single-editing (i.e., updating the model for a single fact), we consider optimizing (2) one edit at a time. We follow the setting in ROME and use GPT-2 XL (1.5B) (Radford et al., 2019). The training takes around 1.4 seconds per edit on average. We give the results in Table 4. We find that it is helpful to include similar facts instead of random facts for locality supervision, presumably because fine-tuning is more sensitive with small data size thus requiring more care in the selection of demonstrations. We use Sentence-BERT embeddings (Reimers and

Gurevych, 2019) and take 15 nearest facts as $\mathcal{R}$. On ZsRE, our method outperforms all existing single-edit methods. On COUNTERFACT, our method lags behind ROME which is specifically optimized for single-editing, but still achieves the second-best edit score.

5.3 Generative Metrics

We report results on generative metrics in Table 5 and 6 for completeness. Following prior work (Meng et al., 2022, 2023), we report fluency (entropy of n-gram distributions) and consistency (similarity score with a reference text). Our fine-tuning methods take a hit on these metrics while improving the edit score, showing that more work is needed to go beyond classification. However, recent work shows that *none* of the compared editors preserves downstream performance (Gu et al., 2024), thus achieving this goal meaningfully is still an open problem.

We additionally show that we can easily incorporate considerations for generative performance into fine-tuning. Specifically, we optimize $(1 - \gamma)L_1(\theta) + \gamma L_2(\theta)$ where $L_1(\theta)$ is the loss in (2) and $L_2(\theta) = -\sum_{w \in \mathcal{W}} \log p_\theta(w)$ is a language modeling loss on random Wikipedia text. We choose Wikipedia articles so that they do not overlap with those used for consistency evaluation. We set $\gamma = 0.1$. With this change ("+ Wikipedia Loss"), we obtain a significant improvement in both fluency and consistency at the cost of a modest drop in the edit score.

6 Conclusions

We have demonstrated that pure fine-tuning is sufficient to obtain strong edit performance with a slight modification: optimizing the conditional likelihood and augmenting the data with additional facts. Our results challenge the assumption that pure fine-tuning is ineffective as a model editor, suggesting that model editing could become part of standard training.

Limitations

Our scope is deliberately limited to the classification aspect of model editing rather than the generative aspect to demonstrate the viability of a fine-tuning editor. Thus our approach is currently unable to preserve the LLM's general capabilities, as is the case with all existing mass editors.

References

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

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.

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.

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.

Jia-Chen Gu, Hao-Xiang Xu, Jun-Yu Ma, Pan Lu, Zhen-Hua Ling, Kai-Wei Chang, and Nanyun Peng. 2024. Model editing can hurt general abilities of large language models. *arXiv preprint arXiv:2401.04700*.

Edward J Hu, yelong shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, and Weizhu Chen. 2022. LoRA: Low-rank adaptation of large language models. In *International Conference on Learning Representations*.

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 J Andonian, Yonatan Belinkov, and David Bau. 2023. 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. 2022. Fast model editing at scale. In *International Conference on Learning Representations*.

Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, Ilya Sutskever, et al. 2019. Language models are unsupervised multitask learners. *OpenAI blog*, 1(8):9.

Rafael Rafailov, Archit Sharma, Eric Mitchell, Christopher D Manning, Stefano Ermon, and Chelsea Finn. 2023. Direct preference optimization: Your language model is secretly a reward model. In *Thirty-seventh Conference on Neural Information Processing Systems*.

```
Paraphrase the given incomplete statement without changing the meaning. The same completion in the original input must also work for your paraphrase. Provide as many distinct paraphrases as you can come up with.

INPUT: The war during which Mario Stoppani was in the armed forces was
1) Paraphrased: The war Mario Stoppani was in the army in was
2) Paraphrased: Mario Stoppani was in the war called
3) Paraphrased: Mario Stoppani served in the army during

INPUT: The birth date of Luca Pianca is
1) Paraphrased: The date of birth of Luca Pianca is
2) Paraphrased: Luca Pianca was born in
3) Paraphrased: The day of Luca Pianca's birth is
4) Paraphrased: Luca Pianca's birth date is
5) Paraphrased: The birthday of Luca Pianca is

INPUT: The name of the architect of Ravenna Cathedral is
1) Paraphrased: Ravenna Cathedral was built by
2) Paraphrased: The person who built Ravenna Cathedral was
3) Paraphrased: The architect of Ravenna Cathedral is
4) Paraphrased: The architect behind the construction of Ravenna Cathedral is
5) Paraphrased: Ravenna Cathedral was built by the architect

INPUT: {prompt}
1) Paraphrased:
```

Figure 1: The few-shot prompt we use for paraphrase generation. We selected in-context examples from COUNTERFACT.

Nils Reimers and Iryna Gurevych. 2019. Sentence-bert: Sentence embeddings using siamese bert-networks. 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)*, pages 3982–3992.

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

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

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*.

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*.

A WikiRecent Dataset

For additional mass-editing experiments, we create a simplified version of the WikiRecent dataset (Cohen et al., 2023). The original dataset consists of 570 and 1,266 facts from Wikidata for training and testing, where each fact has been recently modified (after July 2022). An example prompt is π =“The name of the position held by Nicolaus Bergius is” with the target object o =“bishop”. Since the dataset does not provide paraphrase prompts for evaluating generalization, we create paraphrases of π by few-shot prompting GPT (`gpt-3.5-turbo-0125`). Since the task is relatively straightforward, we find the generated paraphrases preserve the original meaning (e.g., “The position held by Nicolaus Bergius is”). For neighborhood prompts, we select 10 random neighborhood prompts from COUNTERFACT from the corresponding train/test splits. The training data for our fine-tuning method consists of 1,266 edit requests, 18,990 augmented paraphrases (i.e., randomly generated), and 12,660 random neighborhood prompts from COUNTERFACT (not overlapping with those in the test set).