GenPoE: Generative Passage-level Mixture of Experts for Knowledge Enhancement of LLMs

Anonymous ACL submission

Abstract

Typically, parametric adaptation methods such as domain-adaptive pretraining (DAP) and retrieval-augmented generation (RAG) have been considered effective approaches for adapting large language models (LLMs) to new knowledge or domains. To unify positive effects of parametric adaptation and RAG, this paper proposes **GenPoE**, i.e., "generative" passage-level mixture of experts (MoEs) for enhancing knowledge of LLMs. The key component is its novel *MoE-generating hypernetwork* which takes in-context retrieved passages and generates their "expert" parameters, where these generated parameters are then integrated into LLMs by forming expert networks. With its use of "generated" parameters, GenPoE does not require a separate parameter training or fine-tuning stage, which is often costly. By parameterizing passages into expert networks, GenPoE likely exhibits robustness even when the retrieved passages are irrelevant. Experiment results in two open-domain question answering (QA) tasks present that GenPoE shows improved performances over other passage-level knowledge editing, and its combination of RAG produces superior performances over RAG. Our data and code will be available at https://github.com/XXX/XXX.

1 Introduction

Large Language Models (LLMs) have achieved remarkable success across a wide range of natural language processing tasks, especially in knowledge-intensive applications due to their vast pretrained knowledge (Zhao et al., 2023; Hadi et al., 2023). However, the dynamic nature of real-world knowledge necessitates continuous and efficient updates to these models.

Existing methods for enhancing knowledge of LLMs are broadly categorized to two approaches – *parametric adaptation* and *RAG* (Lewis et al., 2020; Pan et al., 2024); 1) Parametric adaptation – such as domain-adaptive pretraining (DAP) (Ke et al., 2023) or *parametric* knowledge editing (De Cao et al., 2021; Meng et al., 2022a) [1] – modifies or updates the parameters of large language models (LLMs). However, it often requires substantial computational and memory cost due to LLM's huge parameters and the training time, and may suffer from the catastrophic forgetting, which can degrade previously learned knowledge and task performance. 2) RAG retrieves relevant passages and uses them as additional in-context prompts before generating responses. However, RAG likely suffers from performance degradation due to noisy or irrelevant retrieved content (Yoran et al., 2023; Tu et al., 2025), and its lack of parameter-level integration may limit the model's ability to accumulate and evolve knowledge over time.

Given these characteristics, we assume that parametric adaptation and RAG serve as complementary solutions, each addressing the limitations of the other. RAG is less likely to suffer from computational overhead and catastrophic forgetting, while parametric adaptation tends to avoid the influence of noisy or irrelevant retrieved content due to its access to internalized parametric knowledge. To unify these complementary effects, this paper proposes **GenPoE**, i.e., "generative" passage-level MoEs for enhancing knowledge of LLMs, by internalizing passages into MoE parameters based on our novel *MoE-generating hypernetwork*. More specifically, the MoE-generating hypernetwork is trained via *meta-learning* framework: given a question and its corresponding gold passage, it learns to produce passage-level MoE parameters such that the resulting MoE-equipped LLMs maximize the autoregressive likelihood of generating the cor-

[1] In this paper, we refer to parametric knowledge editing as encompassing locate- parametric approaches described in (Yao et al., 2023), excluding memory-based methods. it does not include in-context knowledge editing as proposed in (Zheng et al., 2023a).

rect answer to the question. During inference, as an alternative to RAG, GenPoE employs the MoE-generating hypernetwork through a *passage injection stage* to incorporate retrieved passages. Once the top-k passages are retrieved, the hypernetwork dynamically constructs k passage-level MoEs, which are then integrated with the original LLM parameters, as illustrated in Figure 1.

Experimental results on standard open-domain question answering (QA) datasets—including Natural Questions (NQ) (Kwiatkowski et al., 2019), and TriviaQA (TQA) (Joshi et al., 2017)—demonstrate that GenPoE achieves a stronger passage injection effect by outperforming baseline parametric knowledge editing methods. When combined with RAG, GenPoE gives rise to a more powerful RAG variant that surpasses the performance of existing advanced RAG approaches, confirming the complementary relationship between RAG and parametric adaptation.

Our main contributions are summarized as follows:

1. We propose GenPoE, a novel passage injection framework for enhancing the knowledge of large language models (LLMs), which leverages hypernetwork to generate passage-level Mixture-of-Experts (MoE) parameters for open-domain question answering tasks.

2. We present a novel meta-learning framework for training a MoE-generating hypernetwork, where the hypernetwork is trained in an end-to-end manner to minimize the loss of generating correct answers, given a question, its corresponding gold passage, and the ground-truth answer.

3. We present comprehensive empirical results on standard QA benchmarks, demonstrating that GenPoE consistently outperforms conventional parametric knowledge editing approaches and parameter-efficient tuning methods, and further enhances performance when combined with retrieval-based generation (RAG), forming a more powerful RAG variant.

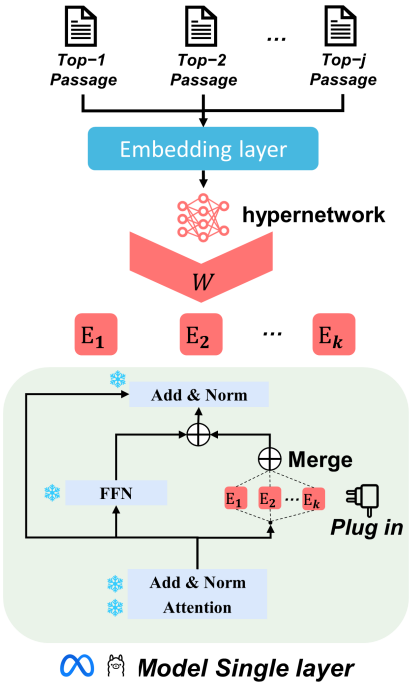

Figure 1: Once the top-k retrieved passages are obtained, GenPoE transforms this knowledge into passage-level MoEs using the *MoE-generating hypernetwork*, which is pretrained via a meta-learning framework. The resulting *passage-level MoEs* are then combined with the original feed-forward network (FFN) parameters – without modifying them – to generate a response to the given question.

2 Related Work

2.1 Knowledge Enhancement

Knowledge injection has emerged as a crucial technique for enhancing the performance of language models by efficiently integrating external knowledge. Two primary approaches have gained significant attention: parameter-efficient tuning and model editing.

Parameter-efficient tuning enhances large pretrained models by introducing small trainable modules while keeping most parameters frozen. Key approaches include adapter (Houlsby et al., 2019), which adds neural network bottlenecks to transformer blocks; Prompt Tuning (Li and Liang, 2021), which optimizes the appended prompts for task adaptation; and LoRA (Hu et al., 2021), which updates rank decomposition matrices. Recent advances, such as DyLoRA (Valipour et al., 2022), improve efficiency by selectively updating partial parameters. Building on DyLoRA, MELO (Yu et al., 2024) introduces a neuron-indexed dynamic LoRA mechanism.

Model editing focuses on maintaining the reliability of edited knowledge, ensuring that the changes successfully address the target queries. Additionally, it emphasizes enhancing the generality, allowing the edited model to generalize the

new knowledge to related queries effectively. Furthermore, it seeks to preserve locality, ensuring that the modifications do not interfere with the retention of unrelated original knowledge. It is categorized into three types: Meta-learning editors (De Cao et al., 2021; Mitchell et al., 2021; Tan et al., 2023), which use hyper-networks to adjust gradients; Locate-then-edit editors (Meng et al., 2022a,b; Li et al., 2024), which identify and update relevant parameters; and Memory-based editors, where (Zheng et al., 2023b; Zhong et al., 2023; Gu et al., 2023; Cheng et al., 2023) update knowledge from prompts using in-context learning without gradient updates or parameter modifications, where other approaches like T-Patcher (Huang et al.) and MEMoE (Wang and Li, 2024) store the memory of edited facts using additional parameters.

Further, PRAG (Su et al., 2025) is a novel RAG paradigm that integrates external knowledge into a LLM's FFN layers by parameterizing passage representations into LoRA weights. The recently proposed DyPRAG (Tan et al., 2025) extends PRAG by introducing a hypernetwork component that enables dynamic generation of passage-specific LoRA parameters at inference time. Moreover, DyPRAG employs a two-stage training procedure: it first learns passage-specific LoRA weights, which are then used as supervision to train the hypernetwork for dynamic LoRA parameter generation during inference. By comparison, our proposed method, GenPoE, is an end-to-end framework that trains the hypernetwork using only the answer as supervision, enabling dynamic MoE parameter generation at inference time without requiring any additional training stage.

2.2 Mixture of Experts

In transformer-based Large Language Models (LLMs), Mixture-of-Experts (MoE) layers utilize a set of expert networks and a gating mechanism to route inputs to the most suitable experts (Shazeer et al., 2017; Antoniak et al., 2023). These layers are strategically positioned after the self-attention sublayer to optimize feed-forward network (FFN) selection, significantly reducing computational overhead in large models like PaLM (Chowdhery et al., 2023), where FFN layers account for the majority of parameters.

Dense MoE approaches activate all available experts simultaneously, which enhances predictive accuracy but demands substantial computational resources. Early implementations (Jacobs et al., 1991; Rasmussen and Ghahramani, 2001; Aljundi et al., 2017) demonstrated this effectiveness, and more recent methods like EvoMoE (Nie et al., 2021), MoLE (Wu et al., 2024), LoRAMoE (Dou et al., 2023), and DSMoE (Pan et al., 2024) have refined the dense MoE structure to balance performance and efficiency.

Sparse MoE improves computational efficiency by selecting only the top-k experts for each input, thereby maintaining accuracy while reducing processing demands (Shazeer et al., 2017). However, this selective activation can cause load imbalances, where certain experts are overused while others are underutilized. To counter this, auxiliary loss functions are introduced to distribute tokens more evenly across experts, as seen in (Lepikhin et al., 2020; Jiang et al., 2024a; Du et al., 2022; Fedus et al., 2022). This strategy allows sparse MoE models to scale effectively by expanding parameter capacity without a corresponding increase in computational cost.

3 Methodology

Figure 2 provides an overview of GenPoE, highlighting its two key stages: (1) the meta-learning stage for training the hypernetwork (Section 3.2), and (2) the passage injection via MoEs during inference (Section 3.3.1). In the following sections, we first introduce our MoE framework that integrates passage-level experts.

3.1 Passage-level MoE

In GenPoE, passages are parameterized to passage-level experts, referred to as passage-level MoEs, and original parameters of LLMs are not modified. More specifically, each passage-level expert module is constructed as an auxiliary FFN that operates in parallel with the original FFN layer of the base transformer model. Formally, we denote the original FNN block as $\text{FFN}^l(\mathbf{x}_t^l)$, defined as follows:

$$\text{FFN}^l(\mathbf{x}_t^l) = \text{ReLU}\left(\mathbf{x}_t^l \mathbf{W}_K^l\right) \mathbf{W}_V^l \qquad (1)$$

where $\mathbf{x}_t^l \in \mathbb{R}^{d_m}$ is the input representation of t-th token at l-th FFN layer, and $\mathbf{W}_K \in \mathbb{R}^{d_m \times d}$ and $\mathbf{W}_V \in \mathbb{R}^{d \times d_m}$ represent the up and down projection matrices at the original FFN layer, respectively.

In addition to the original FFN, a passage-level expert $\text{E}^l(\mathbf{x}_t^l)$ is incorporated. To reduce parameter overhead and enable efficient training, we apply *low-rank matrix factorization* to the expert's up

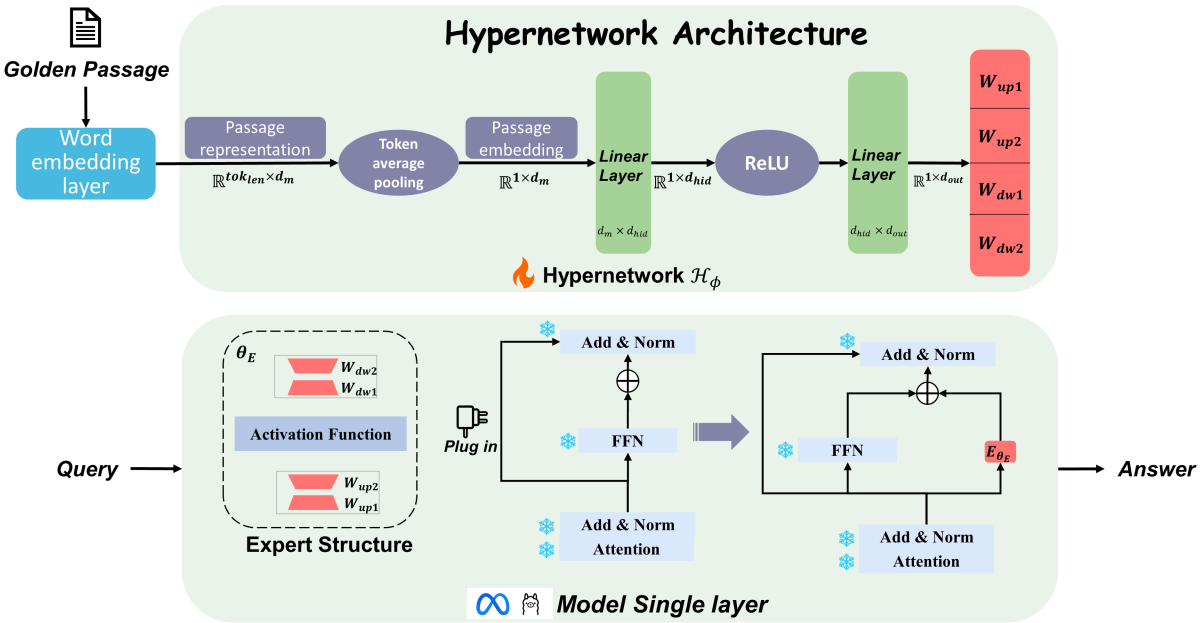

Figure 2: An overview of the proposed GenPoE framework: 1) Meta-learning for training the MoE-generating hypernetwork: For each tuple $(query, gold\ passage, answer)$ in the dataset $\mathcal{D}$, the hypernetwork $\mathcal{H}_\phi$ is trained to generate passage-specific expert parameters. This ensures that contextual information from each passage is effectively incorporated into the model to support reasoning, by optimizing the meta-loss defined in Eq. (5) (see §3.2.2). 2) Inference via Passage Injection: At inference time, given the top-retrieved passages for a test query q_{test}, obtained through a retrieval and reranking process (see § 3.3.2), GenPoE applies the hypernetwork $\mathcal{H}_\phi$ to generate passage-specific expert parameters. These parameters are then dynamically integrated into the sparse mixture-of-experts (MoE) architecture (see § 3.3.1), enabling the model to perform query-aware reasoning by incorporating contextual knowledge from the retrieved content.

and down projection matrices, resulting in a low-rank approximated FFN:

$$\mathrm{E}^l_{\boldsymbol{\theta}_E}(\mathbf{x}^l_t) = \mathrm{ReLU}\left(\mathbf{x}^l_t \mathbf{W}^l_{up1} \mathbf{W}^l_{up2}\right) \mathbf{W}^l_{dw1} \mathbf{W}^l_{dw2} \tag{2}$$

where $\boldsymbol{\theta}_E = \{\mathbf{W}_{up1}{}^l, \mathbf{W}^l_{up2}, \mathbf{W}^l_{dw1}, \mathbf{W}^l_{dw2}\}$, $\mathbf{W}^l_{up1} \in \mathbb{R}^{d_m \times k}$, $\mathbf{W}^l_{up2} \in \mathbb{R}^{k \times d_m}$, $\mathbf{W}^l_{dw1} \in \mathbb{R}^{d_m \times k}$, and $\mathbf{W}^l_{dw2} \in \mathbb{R}^{k \times d_m}$ represent the low-rank approximations of the up and down projection metrices for the passage-level expert.

The expert is inserted alongside the original FFN block, forming the following "passage-level" MoE layer:

$$\mathbf{y}^l_t = \mathrm{FFN}^l(\mathbf{x}^l_t) + \lambda^l \mathrm{E}^l_{\boldsymbol{\theta}_E}(\mathbf{x}^l_t) \tag{3}$$

where λ^t is the mixing parameter, set to 1 for a specific layer l and 0 otherwise, depending on the layer index (i.e. λ^l=0, no expert module is inserted at the current layer), enabling sparse expert selection and targeted knowledge editing at the passage level.

3.2 Meta Learning for Training MoE-generating Hypernetwork

The expert parameters $\boldsymbol{\theta}_E$ need to be specifically determined for each passage. Instead of applying parametric adaptation methods such as fine-tuning for individual passages, GenPoE "generates" these parameters of the passage-specific expert using the *MoE-generating hypernetwork*, inspired by the generative adaptor proposed in (Chen et al., 2025).

3.2.1 Hypernetwork Architecture

The structure of the MoE-generating hypernetwork is illustrated in upper part of Figure 2; it takes a passage representation as input and produces the expert parameters $\boldsymbol{\theta}_E$ as output.

Suppose a passage $psg = w_1 \cdots w_T$ of length T is given. The passage psg is fed into the word embedding layer of the LLM, and mean pooling is applied to produce a passage embedding $\mathrm{Emd}(psg) \in \mathbb{R}^{1 \times d_m}$. The hypernetwork, consisting of two linear layers with a ReLU activation function, takes $\mathrm{Emd}(psg)$ as input and generates the expert parameters, formulated as follows:

$$\mathcal{H}_\phi\left(\mathsf{Emd}(psg)\right) = \text{Linear}\left(\text{ReLU}\left(\text{Linear}(\mathsf{Emd}(psg))\right)\right) \in \mathbb{R}^{d_{out}} \quad (4)$$

where d_{out} corresponds to the concatenated dimension of the expert parameters, i.e., $2 \times 2 \times (d_m \times k)$. We reshape the output $\mathcal{H}_\phi\left(\mathsf{Emd}(psg)\right)$ to obtain θ_E, which represents the expert parameters for the given passage psg.

3.2.2 Meta Learning for Training Hypernetwork

To inject passage-specific knowledge into the expert modules, we train a hypernetwork to dynamically generate the parameters of each expert based on its corresponding passage. To this end, we first prepare a training dataset $\mathcal{D} = (q_i, p_g^{q_i}, a^{q_i})_{i=1}^{N}$, collected from QA tasks, where q_i denotes the i-th question, $p_g^{q_i}$ is its corresponding gold passage, and a^{q_i} is the ground-truth answer. Suppose that $P(y \mid x, \boldsymbol{\theta}_E)$ denotes the probability of generating a sequence y given input x, using Eq. (3), where $\boldsymbol{\theta}_E$ represents the expert parameters used in $\mathrm{E}^l_{\boldsymbol{\theta}_E}$.

For designing the meta loss, which serves as the objective function for training the parameters ϕ that generate the expert parameters, we use a simple "autoregressive" language modeling loss over the question-answer text, ensuring that the generated expert enhances the model's response capability for a given question, formally defined as follows:

$$\mathcal{L}(\phi) = -\sum_{(q,p,a)\in\mathcal{D}} \log P(a \mid q, \mathcal{H}_\phi(p)) \quad (5)$$

3.3 Inference via Passage Injection

During inference, GenPoE first retrieves the top passages for a given test query q_{test}, using dense retrieval followed by a reranking step to refine the results. Based on the top-retrieved passages $Top_K(q_{\text{test}})$, GenPoE then performs "passage injection", which dynamically generates the corresponding expert parameters by $\mathcal{H}_\phi$, allowing the model to adapt its reasoning process to the contextualized knowledge retrieved at inference time.

3.3.1 Passage Injection to MoE

Given $Top_K(q_{\text{test}})$, the meta-learned MoE-generating hypernetwork $\mathcal{H}_\phi$ generates expert-specific parameter residuals $\boldsymbol{\theta}_j$ for each selected passage $p_j \in Top_K(q_{\text{test}})$, which are then injected into the expert modules for inference.

Single-Expert Mode: In the single-expert setting ($K = 1$), only the top passage is selected, where the inference is conducted using Eq. (3), where the parameters for the expert is generated according to Eq. (4).

Multi-Expert Mode: Generalizing to the multi-expert setting ($K > 1$), as shown in the Figure 1, the model incorporates information from knowledge in top K-retrieved passages $Top_K(q_{\text{test}})$. The Eq. (3) is then further extended to the following:

$$\mathbf{y}_t^l = \text{FFN}^l(\mathbf{x}_t^l) + \lambda_l^t \sum_{k=1}^{K} \mathbf{r}_k \cdot \mathrm{E}^l_{\mathcal{H}_\phi(p_k)}(\mathbf{x}_t^l) \quad (6)$$

where $\mathrm{E}^l_{\mathcal{H}_\phi(p_k)}$ is the expert module using the generated parameters for p_k by the hypernetwork $\mathcal{H}_\phi$, and the relevance vector $\mathbf{r} \in \mathbb{R}^K$ is computed according to the relevance distribution over top k passages, which will be presented in Section 3.3.2.

The multi-expert setting enhances the model's ability to capture diverse contextual information by aggregating knowledge from multiple source passages, thereby improving the robustness and accuracy of response generation.

3.3.2 Retrieval-Reranker

Given a test query q_{test}, Dense Passage Retrieval (DPR) (Karpukhin et al., 2020) uses both the query vector $\mathsf{Emd}(q_{test})$ and the passage vector $\mathsf{Emd}(p_i)$ for p_i in WiKipage database (Kwiatkowski et al., 2019), and computes inner product similarity between them:

$$score(q_{test}, p_i) = \mathsf{Emd}(q_{test})^\top \mathsf{Emd}(p_i) \quad (7)$$

For the reranking, we incorporate *BAAI/bge-reranker-v2-gemma*[2] (Xiao et al., 2024) as a reranker model and fine-tune it to improve passage relevance. For the finetuning the reranker, we additionally construct a training dataset passages retrieved by DPR. To optimize the reranker, we adopt a contrastive loss function (Sohn, 2016), ensuring that relevant passages receive higher scores than irrelevant ones. For a given question q in the training set, let p_+ be a positive passage, and let $(p_-^{(1)}, \ldots, p_-^{(L)})$ be L negative passages for q. The loss function is defined as:

$$\mathcal{L}_{reranker} = -log \frac{exp\left(score_{rerank}\left(q, p_+\right)\right)}{\sum_{i=1}^{L+1} exp\left(score_{rerank}(q, p^{(i)})\right)} \quad (8)$$

The details and results of the re-ranker can be found in Appendix B.

[2] https://huggingface.co/BAAI/bge-reranker-v2-gemma

The relevance vector $\mathbf{r} \in \mathbb{R}^K$ in Eq. (6) is obtained after reranking step, defined as:

$$\mathbf{r} = \text{softmax}\left(\text{top}K\left([\text{score}_{\text{rerank}}(q_{test}, p_j)]_{j=1}^{K}\right)\right) \tag{9}$$

4 Experiments

In this part, we explain the experimental setup and present the key results of our experiments.

4.1 Experimental Setup

Dataset: To evaluate the proposed method's effectiveness in paragraph-level editing, we conduct experiments using the open-domain question answering datasets Natural Questions (NQ) (Kwiatkowski et al., 2019) and TriviaQA (TQA) (Joshi et al., 2017). These datasets provide a diverse set of challenging questions across various domains, enabling a comprehensive assessment of model performance.

Backbone Model: We select Llama2$_{7B}$ (Touvron et al., 2023) and the more advanced Baichuan2$_{7B}$ (Baichuan, 2023) as base models, into which we integrate expert modules to implement our proposed framework.

Methods: We evaluate several widely-used knowledge editing methods for comparison, including MEMIT (Meng et al., 2022b), MEND (Mitchell et al., 2021), MALMEN (Tan et al., 2023) and Fine-tuning with LoRA (FT-LoRA) (Hu et al., 2021). All methods are implemented using the standardized configurations provided by the EasyEdit [3] framework (Wang et al., 2024), a unified library for model editing. See the Appendix A for more details.

Metric: We use Exact Match (EM) and F1-score (F1), which are standard metrics (Rajpurkar, 2016) in question answering for evaluating answer accuracy and completeness.

Implementation Details: All experiments, including data construction, knowledge editing, and evaluation, were conducted on workstations with 8×NVIDIA RTX A6000 GPUs. For training the hypernetwork, we used the AdamW (Loshchilov and Hutter, 2019) optimizer for 1 epoch, with the learning rate decaying from 1e-4 to 1e-6 using cosine annealing. The expert module, implemented as a low-rank adaptation with $k = 512$, is integrated into the 12_{th} layer of the transformer.

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

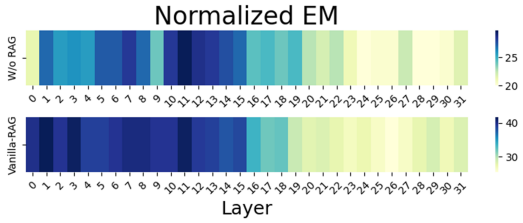

Figure 3: Normalized EM for expert insertion at different layers using Llama2$_{7B}$ on the NQ dataset.

4.2 Selection of Effective Layer

To identify the most effective layer for expert insertion, we conducted experiments by injecting experts at each transformer layer using QA pairs from the validation set, as shown in Figure 3. This layer-wise analysis allows us to evaluate how well injected knowledge is utilized at different depths of the model. We observe that the 12-th layer yields better performance on the Llama2 model. Given that Baichuan2 shares the same architecture as Llama2, we adopt the same insertion layer for consistency.

4.3 Main Results and Analysis

We evaluate the effectiveness of GenPoE under two distinct evaluation paradigms: the W/o-RAG Setting and the Vanilla-RAG Setting, as summarized in Table 1.

W/o-RAG Setting: In the W/o-RAG Setting, models are directly evaluated on the updated parameters using only the test question q_{test}, without access to retrieved passages. This setup isolates the effect of model editing alone. Results indicate that GenPoE significantly outperforms existing editing baselines (e.g., MEMIT, MEND, MALMEN, and FT-LoRA) across all datasets and both model backbones. The results demonstrating superior passage-level knowledge incorporation. In contrast, other editing methods show minimal or negligible gains, highlighting their limited applicability in real-world QA scenarios.

Vanilla-RAG Setting: Models receive both the retrieved passages and the test question as input, allowing them to leverage explicit external knowledge during inference. Even in this more favorable setting, GenPoE continues to outperform both the base RAG models and fine-tuning baselines. This suggests that GenPoE not only enhances the model's internal knowledge via editing but also synergizes effectively with external retrieval, yielding substantial improvements over standard retrieval-

		W/o-RAG Setting				*Vanilla-RAG Setting*			
		NQ		TQA		NQ		TQA	
Model	Methods	EM	F1	EM	F1	EM	F1	EM	F1
Llama2_{7B}	Base	7.80	16.89	49.47	59.38	13.60	23.46	54.40	67.49
	MEMIT	1.01	1.97	6.45	8.62	1.01	6.36	8.60	14.45
	MEND	0.13	0.17	0.07	0.24	0.13	0.20	0.13	0.40
	MALMEN	24.53	33.18	34.73	45.35	27.07	36.43	30.80	41.92
	FT-LoRA	19.53	29.33	54.60	63.96	21.20	31.02	58.53	67.98
	GenPoE	**33.60**	**42.05**	**60.73**	**66.98**	**45.13**	**54.82**	**70.07**	**76.61**
Baichuan2_{7B}	Base	6.07	14.54	43.47	53.07	10.47	19.81	50.47	60.18
	MEMIT	2.50	3.50	0.84	2.47	1.67	3.86	0.84	2.01
	MEND	15.33	22.70	24.60	32.97	17.27	25.97	27.06	36.21
	MALMEN	22.93	31.97	30.93	42.23	20.53	29.66	28.00	38.86
	FT-LoRA	19.87	28.56	49.73	56.80	38.07	48.78	64.67	72.47
	GenPoE	**24.53**	**32.12**	**52.80**	**58.44**	**42.33**	**52.96**	**69.80**	**75.56**

Table 1: Performance of Llama2_{7B} and Baichuan2_{7B} on the NQ and TQA datasets under*W/o-RAG* and *Vanilla-RAG* settings. Performance of Llama2_{7B} and Baichuan2_{7B} on the NQ and TQA datasets under W/o-RAG and Vanilla-RAG settings. In the W/o-RAG setting, only the test question q_{test} is provided as input to the model, while in the Vanilla-RAG setting, the input consists of a concatenation of the retrieved passage and q_{test}, i.e., [retrieved passage, q_{test}].

augmented approaches.

4.4 Effect of the Number of Passage for Enhancement

Based on the trends illustrated in the Figure 4, we can observe the influence of passage quantity on model performance. As the number of retrieved passages increases during inference, Vanilla-RAG shows a noticeable decline in both EM and F1 scores. This degradation suggests that longer input contexts, caused by more passages, make it increasingly difficult for the model to focus on the correct evidence, thereby hampering answer accuracy.

In contrast, GenPoE demonstrates greater robustness to the increasing number of passages. This is largely due to its multi-expert architecture, which distributes the processing passages load across multiple specialized experts. Each expert is exposed to one passage, which helps maintain context clarity and ensures the preservation of answer quality even when more passages are retrieved. These results highlight the scalability advantage of GenPoE in handling expanded input without suffering from the noise and distraction often introduced by lengthy contexts in conventional RAG approaches.

4.5 Sequential Enhancement Progress

Recent works (Gupta et al., 2024; Jiang et al., 2024b) has highlighted the limitations of current knowledge editing methods in continual learning

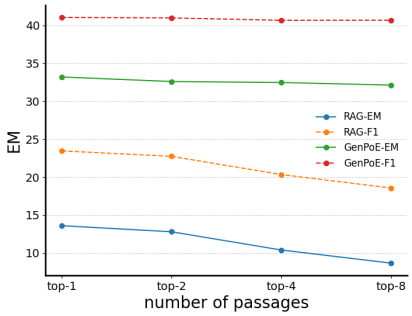

Figure 4: Ablation study on inference with varying numbers of passages, using Llama2_{7B} on the NQ dataset.

or editing settings. Specifically, sequential edits often lead to catastrophic forgetting of previously injected knowledge and may cause interference or conflicts between newly added and existing information. In contrast, our proposed method demonstrates superior robustness in continual editing scenarios.

As shown in Figure 5, we divide 1500 editing instances into batches of 250 and compute the average EM and F1 scores after each batch. Each diagonal block in the heatmap represents the model's performance on a batch of edited samples immediately after their insertion. The results indicate that our method maintains strong performance across editing steps. This advantage primarily stems from the design of our method, where each passage is

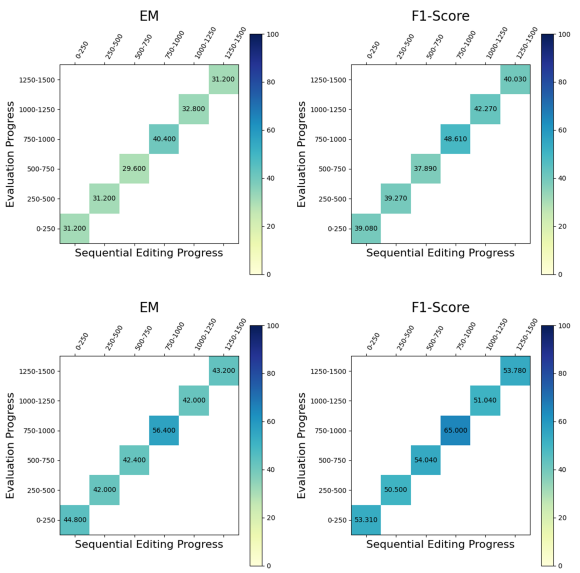

Figure 5: Continuous learning study of using Llama2$_{7B}$ on the NQ dataset. Continuous learning study of using Llama2$_{7B}$ on the NQ dataset. The upper part shows results under the W/o-RAG setting, while the lower part corresponds to the W/ RAG setting.

associated with an independent and pluggable expert module. This modular structure enables new knowledge to be integrated without overwriting previously acquired information, thereby alleviating forgetting and enhancing stability throughout the continual editing process.

4.6 Robustness Analysis

To assess robustness under challenging conditions, we evaluate models in the Oracle-negative setting, where retrieved passages do not contain the ground truth answer. This simulates a realistic open-domain QA scenario with incomplete information. As shown in Table 2, Vanilla-RAG performs similarly to the Base W/o-RAG baseline, indicating limited benefit without answer-containing passages. This suggests Vanilla-RAG heavily depends on explicit answer spans. In contrast, GenPoE remains robust. Even without RAG, it achieves strong results, significantly outperforming baselines. With RAG on non-answer passages, GenPoE still performs competitively, showing its ability to leverage partial or noisy content through its expert-based design.

These results demonstrate that GenPoE not only excels in standard settings but also preserves its advantage in robustness under conditions where conventional retrieval-augmented methods fail.

Method	EM	F1
Base W/o-RAG	7.80	16.89
Base Vanilla-RAG	7.33	16.12
GenPoE (W/o RAG)	32.13	40.98
GenPoE (W/ RAG)	29.53	38.24

Table 2: Robustness analysis under retrieval failure: results of Llama2$_{7B}$ on the NQ dataset.

Method	Injection	Inference	All
Base	-	0.427s	0.427s
MEMIT	52.494s	0.427s	52.921s
MEND	1.258s	0.427s	1.685s
MALMEN	1.717s	0.427s	2.144s
FT-LoRa	7.646s	0.427s	8.073s
GenPoE	0.006s	0.447s	0.453s

Table 3: Efficiency analysis on the NQ dataset using the Llama2$_{7B}$ model

4.7 Efficiency Analysis

The efficiency comparison in Table 3 demonstrates the superior speed of our method. Benefiting from the lightweight design of the hypernetwork, GenPoE integrates paragraph-level information into the model with minimal injection time (0.006s), significantly outperforming other editing methods. Additionally, since expert modules are only inserted at a single transformer layer, the increase in inference time is negligible (from 0.427s to 0.447s), maintaining efficiency comparable to the base model. This design ensures that GenPoE achieves high editing effectiveness with minimal computational overhead.

5 Conclusion

In this work, we propose GenPoE, a hypernetwork-driven mixture-of-experts framework for passage-level knowledge enhancement, aimed at addressing the limitations of large language models in utilizing external prompts. Given a query, relevant passages are encoded into expert modules via a hypernetwork, enabling the injection of knowledge into the model. Experiments on NQ and TQA demonstrate that this approach effectively incorporates knowledge while alleviating the degradation commonly observed in general parameter-update-based methods. For future work, we aim to explore more efficient and expressive hypernetwork and MoE architectures, and develop stable continual learning techniques to improve the adaptability and scalability of the framework.

Limitations

In the current setup, while GenPoE demonstrates strong performance in passage-level knowledge injection, it relies on a hypernetwork to dynamically generate expert parameters conditioned on retrieved passages. The effectiveness of this generation process is inherently constrained by the representational capacity of the hypernetwork. In more complex tasks involving nuanced or diverse knowledge, the current design may struggle to produce accurate and generalizable experts, potentially limiting the model's ability to fully exploit the retrieved information. Future work should explore more expressive and robust hypernetwork architectures to improve expert generation stability and scalability under challenging settings.

References

Josh Achiam, Steven Adler, Sandhini Agarwal, Lama Ahmad, Ilge Akkaya, Florencia Leoni Aleman, Diogo Almeida, Janko Altenschmidt, Sam Altman, Shyamal Anadkat, et al. 2023. Gpt-4 technical report. *arXiv preprint arXiv:2303.08774*.

Rahaf Aljundi, Punarjay Chakravarty, and Tinne Tuytelaars. 2017. Expert gate: Lifelong learning with a network of experts. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 3366–3375.

Szymon Antoniak, Sebastian Jaszczur, Michał Krutul, Maciej Pióro, Jakub Krajewski, Jan Ludziejewski, Tomasz Odrzygóźdź, and Marek Cygan. 2023. Mixture of tokens: Efficient llms through cross-example aggregation. *arXiv preprint arXiv:2310.15961*.

Baichuan. 2023. Baichuan 2: Open large-scale language models. *arXiv preprint arXiv:2309.10305*.

Tong Chen, Hao Fang, Patrick Xia, Xiaodong Liu, Benjamin Van Durme, Luke Zettlemoyer, Jianfeng Gao, and Hao Cheng. 2025. Generative adapter: Contextualizing language models in parameters with a single forward pass. In *The Thirteenth International Conference on Learning Representations*.

Siyuan Cheng, Bozhong Tian, Qingbin Liu, Xi Chen, Yongheng Wang, Huajun Chen, and Ningyu Zhang. 2023. Can we edit multimodal large language models? In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pages 13877–13888, Singapore. Association for Computational Linguistics.

Aakanksha Chowdhery, Sharan Narang, Jacob Devlin, Maarten Bosma, Gaurav Mishra, Adam Roberts, Paul Barham, Hyung Won Chung, Charles Sutton, Sebastian Gehrmann, et al. 2023. Palm: Scaling language modeling with pathways. *Journal of Machine Learning Research*, 24(240):1–113.

Nicola De Cao, Wilker Aziz, and Ivan Titov. 2021. Editing factual knowledge in language models. *arXiv preprint arXiv:2104.08164*.

Shihan Dou, Enyu Zhou, Yan Liu, Songyang Gao, Jun Zhao, Wei Shen, Yuhao Zhou, Zhiheng Xi, Xiao Wang, Xiaoran Fan, et al. 2023. Loramoe: Revolutionizing mixture of experts for maintaining world knowledge in language model alignment. *arXiv preprint arXiv:2312.09979*, 4(7).

Nan Du, Yanping Huang, Andrew M Dai, Simon Tong, Dmitry Lepikhin, Yuanzhong Xu, Maxim Krikun, Yanqi Zhou, Adams Wei Yu, Orhan Firat, et al. 2022. Glam: Efficient scaling of language models with mixture-of-experts. In *International Conference on Machine Learning*, pages 5547–5569. PMLR.

William Fedus, Barret Zoph, and Noam Shazeer. 2022. Switch transformers: Scaling to trillion parameter models with simple and efficient sparsity. *Journal of Machine Learning Research*, 23(120):1–39.

Hengrui Gu, Kaixiong Zhou, Xiaotian Han, Ninghao Liu, Ruobing Wang, and Xin Wang. 2023. Pokemqa: Programmable knowledge editing for multi-hop question answering. *arXiv preprint arXiv:2312.15194*.

Akshat Gupta, Anurag Rao, and Gopala Anumanchipalli. 2024. Model editing at scale leads to gradual and catastrophic forgetting. *arXiv preprint arXiv:2401.07453*.

Muhammad Usman Hadi, Rizwan Qureshi, Abbas Shah, Muhammad Irfan, Anas Zafar, Muhammad Bilal Shaikh, Naveed Akhtar, Jia Wu, Seyedali Mirjalili, et al. 2023. A survey on large language models: Applications, challenges, limitations, and practical usage. *Authorea Preprints*.

Neil Houlsby, Andrei Giurgiu, Stanislaw Jastrzebski, Bruna Morrone, Quentin De Laroussilhe, Andrea Gesmundo, Mona Attariyan, and Sylvain Gelly. 2019. Parameter-efficient transfer learning for nlp. In *International conference on machine learning*, pages 2790–2799. PMLR.

Edward J Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, and Weizhu Chen. 2021. Lora: Low-rank adaptation of large language models. *arXiv preprint arXiv:2106.09685*.

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

Robert A Jacobs, Michael I Jordan, Steven J Nowlan, and Geoffrey E Hinton. 1991. Adaptive mixtures of local experts. *Neural computation*, 3(1):79–87.

Albert Q Jiang, Alexandre Sablayrolles, Antoine Roux, Arthur Mensch, Blanche Savary, Chris Bamford, Devendra Singh Chaplot, Diego de las Casas,

Emma Bou Hanna, Florian Bressand, et al. 2024a. Mixtral of experts. *arXiv preprint arXiv:2401.04088*.

Houcheng Jiang, Junfeng Fang, Tianyu Zhang, An Zhang, Ruipeng Wang, Tao Liang, and Xiang Wang. 2024b. Neuron-level sequential editing for large language models. *arXiv preprint arXiv:2410.04045*.

Mandar Joshi, Eunsol Choi, Daniel S Weld, and Luke Zettlemoyer. 2017. Triviaqa: A large scale distantly supervised challenge dataset for reading comprehension. *arXiv preprint arXiv:1705.03551*.

Vladimir Karpukhin, Barlas Oguz, Sewon Min, Patrick Lewis, Ledell Wu, Sergey Edunov, Danqi Chen, and Wen-tau Yih. 2020. Dense passage retrieval for open-domain question answering. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 6769–6781, Online. Association for Computational Linguistics.

Zixuan Ke, Yijia Shao, Haowei Lin, Tatsuya Konishi, Gyuhak Kim, and Bing Liu. 2023. Continual pre-training of language models. In *The Eleventh International Conference on Learning Representations*.

Tom Kwiatkowski, Jennimaria Palomaki, Olivia Redfield, Michael Collins, Ankur Parikh, Chris Alberti, Danielle Epstein, Illia Polosukhin, Jacob Devlin, Kenton Lee, et al. 2019. Natural questions: a benchmark for question answering research. *Transactions of the Association for Computational Linguistics*, 7:453–466.

Dmitry Lepikhin, HyoukJoong Lee, Yuanzhong Xu, Dehao Chen, Orhan Firat, Yanping Huang, Maxim Krikun, Noam Shazeer, and Zhifeng Chen. 2020. Gshard: Scaling giant models with conditional computation and automatic sharding. *arXiv preprint arXiv:2006.16668*.

Patrick Lewis, Ethan Perez, Aleksandra Piktus, Fabio Petroni, Vladimir Karpukhin, Naman Goyal, Heinrich Küttler, Mike Lewis, Wen-tau Yih, Tim Rocktäschel, et al. 2020. Retrieval-augmented generation for knowledge-intensive nlp tasks. *Advances in Neural Information Processing Systems*, 33:9459–9474.

Xiang Lisa Li and Percy Liang. 2021. Prefix-tuning: Optimizing continuous prompts for generation. *arXiv preprint arXiv:2101.00190*.

Xiaopeng Li, Shasha Li, Shezheng Song, Jing Yang, Jun Ma, and Jie Yu. 2024. Pmet: Precise model editing in a transformer. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 38, pages 18564–18572.

Ilya Loshchilov and Frank Hutter. 2019. Decoupled weight decay regularization. In *International Conference on Learning Representations*.

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 Andonian, Yonatan Belinkov, and David Bau. 2022b. Mass-editing memory in a transformer. *arXiv preprint arXiv:2210.07229*.

Eric Mitchell, Charles Lin, Antoine Bosselut, Chelsea Finn, and Christopher D Manning. 2021. Fast model editing at scale. *arXiv preprint arXiv:2110.11309*.

Xiaonan Nie, Xupeng Miao, Shijie Cao, Lingxiao Ma, Qibin Liu, Jilong Xue, Youshan Miao, Yi Liu, Zhi Yang, and Bin Cui. 2021. Evomoe: An evolutional mixture-of-experts training framework via dense-to-sparse gate. *arXiv preprint arXiv:2112.14397*.

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.

Bowen Pan, Yikang Shen, Haokun Liu, Mayank Mishra, Gaoyuan Zhang, Aude Oliva, Colin Raffel, and Rameswar Panda. 2024. Dense training, sparse inference: Rethinking training of mixture-of-experts language models. *arXiv preprint arXiv:2404.05567*.

P Rajpurkar. 2016. Squad: 100,000+ questions for machine comprehension of text. *arXiv preprint arXiv:1606.05250*.

Carl Rasmussen and Zoubin Ghahramani. 2001. Infinite mixtures of gaussian process experts. *Advances in neural information processing systems*, 14.

Noam Shazeer, Azalia Mirhoseini, Krzysztof Maziarz, Andy Davis, Quoc Le, Geoffrey Hinton, and Jeff Dean. 2017. Outrageously large neural networks: The sparsely-gated mixture-of-experts layer. *arXiv preprint arXiv:1701.06538*.

Kihyuk Sohn. 2016. Improved deep metric learning with multi-class n-pair loss objective. In *Advances in Neural Information Processing Systems*, volume 29. Curran Associates, Inc.

Weihang Su, Yichen Tang, Qingyao Ai, Junxi Yan, Changyue Wang, Hongning Wang, Ziyi Ye, Yujia Zhou, and Yiqun Liu. 2025. Parametric retrieval augmented generation. *arXiv preprint arXiv:2501.15915*.

Chenmien Tan, Ge Zhang, and Jie Fu. 2023. Massive editing for large language models via meta learning. *arXiv preprint arXiv:2311.04661*.

Yuqiao Tan, Shizhu He, Huanxuan Liao, Jun Zhao, and Kang Liu. 2025. Better wit than wealth: Dynamic parametric retrieval augmented generation for test-time knowledge enhancement. *arXiv preprint arXiv:2503.23895*.

Hugo Touvron, Louis Martin, Kevin Stone, Peter Albert, Amjad Almahairi, Yasmine Babaei, Nikolay Bashlykov, Soumya Batra, Prajjwal Bhargava, Shruti

Bhosale, et al. 2023. Llama 2: Open foundation and fine-tuned chat models. *arXiv preprint arXiv:2307.09288*.

Yiteng Tu, Weihang Su, Yujia Zhou, Yiqun Liu, and Qingyao Ai. 2025. Rbft: Robust fine-tuning for retrieval-augmented generation against retrieval defects. *arXiv preprint arXiv:2501.18365*.

Mojtaba Valipour, Mehdi Rezagholizadeh, Ivan Kobyzev, and Ali Ghodsi. 2022. Dylora: Parameter efficient tuning of pre-trained models using dynamic search-free low-rank adaptation. *arXiv preprint arXiv:2210.07558*.

Peng Wang, Ningyu Zhang, Bozhong Tian, Zekun Xi, Yunzhi Yao, Ziwen Xu, Mengru Wang, Shengyu Mao, Xiaohan Wang, Siyuan Cheng, Kangwei Liu, Yuansheng Ni, Guozhou Zheng, and Huajun Chen. 2024. EasyEdit: An easy-to-use knowledge editing framework for large language models. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 3: System Demonstrations)*, pages 82–93, Bangkok, Thailand. Association for Computational Linguistics.

Renzhi Wang and Piji Li. 2024. Memoe: Enhancing model editing with mixture of experts adaptors. *arXiv preprint arXiv:2405.19086*.

Xun Wu, Shaohan Huang, and Furu Wei. 2024. Mixture of lora experts. *arXiv preprint arXiv:2404.13628*.

Shitao Xiao, Zheng Liu, Peitian Zhang, Niklas Muennighoff, Defu Lian, and Jian-Yun Nie. 2024. C-pack: Packed resources for general chinese embeddings. In *Proceedings of the 47th international ACM SIGIR conference on research and development in information retrieval*, pages 641–649.

Wanli Yang, Fei Sun, Jiajun Tan, Xinyu Ma, Qi Cao, Dawei Yin, Huawei Shen, and Xueqi Cheng. 2025. The mirage of model editing: Revisiting evaluation in the wild. *arXiv preprint arXiv:2502.11177*.

Yunzhi Yao, Peng Wang, Bozhong Tian, Siyuan Cheng, Zhoubo Li, Shumin Deng, Huajun Chen, and Ningyu Zhang. 2023. Editing large language models: Problems, methods, and opportunities. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pages 10222–10240, Singapore. Association for Computational Linguistics.

Ori Yoran, Tomer Wolfson, Ori Ram, and Jonathan Berant. 2023. Making retrieval-augmented language models robust to irrelevant context. *arXiv preprint arXiv:2310.01558*.

Lang Yu, Qin Chen, Jie Zhou, and Liang He. 2024. Melo: Enhancing model editing with neuron-indexed dynamic lora. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 38, pages 19449–19457.

Wayne Xin Zhao, Kun Zhou, Junyi Li, Tianyi Tang, Xiaolei Wang, Yupeng Hou, Yingqian Min, Beichen Zhang, Junjie Zhang, Zican Dong, et al. 2023. A survey of large language models. *arXiv preprint arXiv:2303.18223*.

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

Ce Zheng, Lei Li, Qingxiu Dong, Yuxuan Fan, Zhiyong Wu, Jingjing Xu, and Baobao Chang. 2023b. Can we edit factual knowledge by in-context learning? *arXiv preprint arXiv:2305.12740*.

Zexuan Zhong, Zhengxuan Wu, Christopher D Manning, Christopher Potts, and Danqi Chen. 2023. Mquake: Assessing knowledge editing in language models via multi-hop questions. *arXiv preprint arXiv:2305.14795*.

A Limited editing approaches

Most existing knowledge editing or augmentation methods are designed and evaluated under idealized settings, making them difficult to generalize or apply directly to real-world scenarios (Yang et al., 2025) (See the Table 4). In this section, we categorize and analyze current editing approaches by their limitations in three key dimensions: *Editing Input*, *Generation Strategy*, and *Output Truncation*.

Editing Input: Some representative methods, such as ROME and MEMIT, impose strict constraints on the editing input format. Specifically, they require a structured triple input, typically represented as (subject, relation, target). This restricts their applicability in open-domain or flexible editing tasks where knowledge is expressed more freely or contextually, such as full passages or natural questions. In contrast, MEND and MALMEN relax this constraint by allowing more flexible QA-style inputs. However, these methods still require that the editing and testing queries follow the same format and linguistic style. As a result, they struggle to support use cases like updating passage-level knowledge based on question-answer inputs. our proposed method, GenPoE, overcomes these limitations by leveraging a hypernetwork that generates expert parameters from diverse types of knowledge input. This flexibility allows GenPoE to support editing at the passage, sentence, or even entity level. To ensure fair comparison during experiments, we convert the passage-level knowledge into formats

Method	Input Type	Generation Strategy	Output Truncation	Real-world Friendliness
ROME	Triple data	teacher forcing	ground truth length	☹
MEMIT	Triple data	teacher forcing	ground truth length	☹
MEND	Context-free	teacher forcing	ground truth length	☺
MALMEN	Context-free	teacher forcing	ground truth length	☺
GenPoE	Context-guided	autoregressive decoding	natural stopping criteria	☻

Table 4: Comparison of current editing methods in real-world applications. ☹: unfriendly, ☺: moderately friendly, and ☻: highly friendly.

compatible with MEMIT, MEND, and MALMEN. Details of this transformation process can be found in Appendix D.

Generation Strategy: Another major limitation of prior work lies in the generation strategy. Most existing methods (ROME (Meng et al., 2022a), MEMIT (Meng et al., 2022b), MEND (Mitchell et al., 2021), MALMEN (Tan et al., 2023)) use teacher forcing during generation. While this simplifies training and evaluation, it leads to overly optimistic results, as the model is always conditioned on the ground truth tokens. This setting hides generation errors that may occur in actual deployment, such as token drift or hallucination. In contrast, GenPoE adopts a fully autoregressive decoding process, better simulating real inference scenarios. This strategy allows us to assess the true quality and robustness of the edited model outputs.

Output Truncation: Output truncation strategies in prior works further exaggerate their performance under unrealistic assumptions. Existing methods commonly truncate model outputs to the same length as the gold answer. While this creates a cleaner evaluation setup, it masks issues like repetition, irrelevance, or incomplete answers that naturally arise when relying on the model's own stopping behavior. Our method avoids this artificial constraint by allowing natural stopping criteria during decoding. This setup more accurately reflects real-world model usage, where answers are generated until a learned end-of-sequence signal is reached.

As summarized in Table 4, these limitations collectively make many prior methods less suitable for real-world deployment. Our method, GenPoE, addresses all three limitations, making it significantly more adaptable and effective in practical knowledge editing scenarios.

B Details of Reranker

This section presents the performance of applying a re-ranker to refine the retrieval results of DPR (Karpukhin et al., 2020), comparing the accuracy before and after re-ranking. In a question-answering system, the accuracy of the retrieval source plays a crucial role in the overall result. Additionally, the scoring produced by the re-ranker is important for merging between experts. The improvements were evaluated using Top-K accuracy, which measures the presence of a golden passage, ensuring a stable document source for responing answer.

Top-K	Type	NQ	TQA
@1	Before	44.60	56.53
	After	64.67	76.33
@2	Before	55.73	65.27
	After	71.73	79.60
@4	Before	64.47	72.07
	After	77.87	82.80
@8	Before	72.93	76.73
	After	81.47	84.53

Table 5: Comparison of DPR retrieval accuracy results before and after applying the re-ranker we trained.

C Case Study

Table 6 presents a case study illustrating the effectiveness of different methods in answering the question *"Who produced A Change Is Gonna Come?"* The target answer is "**Hugo & Luigi**".

The retrieved top-1 document provides relevant information, stating that the song was produced by "Hugo & Luigi". However, both the Base and Vanilla-RAG methods incorrectly answer "**Sam Cooke**", indicating a failure to extract the precise producer information from the retrieved passage.

In contrast, the GenPoE method, whether applied without retrieval augmentation (W/o RAG) or with retrieval augmentation (W/ RAG), successfully generates the correct answer "**Hugo & Luigi**", demonstrating its superior ability to leverage the retrieved evidence for precise question answering.

This case exemplifies the advantage of Gen-

Question: Who produced a change is gonna come?		
Target answer: Hugo & Luigi		
Retrieved Top-1 Documents: A Change Is Gonna Come" is a song by American recording artist Sam Cooke. It initially appeared on Cooke's album "Ain't That Good News", released March 1, 1964 by RCA Victor; a slightly edited version of the recording was released as a single on December 22, 1964. Produced by Hugo & Luigi and arranged and conducted by René Hall, $\cdots$		
Method	Answer	Status
Base	Sam Cooke	✗
Vanilla-RAG	Sam Cooke	✗
GenPoE (W/o RAG)	Hugo & Luigi	✓
GenPoE (W/ RAG)	Hugo & Luigi	✓

Table 6: Caption

PoE in extracting accurate factual knowledge from retrieved documents compared to baseline approaches.

D Generation of Editing Facts

As shown in Table 7, we first adopt template to generate synthetic QA pairs tailored for the MEND, MALMEN, and FT-LoRA methods. Specifically, the template incorporates a structured instruction along with the retrieved passage, which is inserted into the placeholder "**{paragraph}**", and is then fed into a large language model "gpt4o-mini" (Ouyang et al., 2022; Achiam et al., 2023) to produce concise, exact-match question-answer pairs suitable for knowledge editing supervision.

Building on these QA pairs, we further apply template of Table 8 to transform them into structured (*subject, relation, object*), which are used as training instances for the MEMIT method (Meng et al., 2022b). To generate data suitable for MEMIT's knowledge injection mechanism, the template places the generated question into the "**{prompt}**" placeholder, ensuring alignment with the required input format.

Prompt template for self-generating synthetic knowledge

You are an assistant who is good at organizing questions and answers from paragraphs. Here is an example.

Paragraph: "on death row in the United States on January 1, 2013. Since 1977, the states of Texas (464), Virginia (108) and Oklahoma (94) have executed the most death row inmates. , California (683), Florida (390), Texas (330) and Pennsylvania (218) housed more than half of all inmates pending on death row. , the longest-serving prisoner on death row in the US who has been executed was Jack Alderman who served over 33 years. He was executed in Georgia in 2008. However, Alderman only holds the distinction of being the longest-serving ëxecutedïnmate so far. A Florida inmate, Gary Alvord, arrived"
1. Q: How many death row inmates did Texas execute since 1977?
A: 464
2. Q: Which state executed 108 death row inmates since 1977?
A: Virginia
3. Q: How many death row inmates did Oklahoma execute since 1977?
A: 94
4. Q: Which state housed 683 death row inmates as of January 1, 2013?
A: California
5. Q: How many inmates did Florida house on death row?
A: 390
6. Q: How many death row inmates did Texas have pending?
A: 330
7. Q: How many death row inmates did Pennsylvania house?
A: 218
8. Q: Who was the longest-serving prisoner on death row who was executed?
A: Jack Alderman
9. Q: How many years did Jack Alderman serve on death row?
A: over 33 years
10. Q: In which year was Jack Alderman executed?
A: 2008
11. Q: Which state executed Jack Alderman?
A: Georgia
12. Q: Who is noted as the longest-serving "executed" inmate?
A: Jack Alderman
13. Q: Which inmate arrived in Florida?
A: Gary Alvord
14. Q: What is the date referenced for death row statistics in the passage?
A: January 1, 2013
15. Q: Since when has the execution data been tracked in this passage?
A: 1977
16. Q: What constitutes more than half of all inmates pending on death row?
A: California, Florida, Texas, and Pennsylvania

Please follow the format of the example above to generate sixteen questions and corresponding answers for the following Paragraph. The format of answers should be a very short phrase from paragraph, such as “464”, "2008", "May 16th, 1931”, or “Jack Alderman”, to meet the criteria of exact match Paragraph.

Paragraph: "{***paragraph***}"

Table 7: This prompt template is designed to generate synthetic question-answer pairs from a passage. It includes clear instructions outlining the requirements, along with an example paragraph and its corresponding question-answer pairs. The model is then expected to create similar question-answer pairs for new paragraphs following this format.

Prompt template for generating subject extraction

Read the following prompt carefully. Identify the subject of the sentence. Output must be only the subject word , exactly as it appears in the 'prompt' — preserving the original capitalization and formatting.

Here are some examples for guidance:

'prompt': "Which religion is noted as the fourth largest after Christianity, Islam, and Hinduism?', 'subject': 'religion'
'prompt': 'Who was mainly responsible for the design of Abney Park Chapel?', 'subject': 'design of Abney Park Chapel'
'prompt': "What layer of skin is directly below the dermis and epidermis?", 'subject': 'layer of skin'
'prompt': "What mountain range is Aconcagua Provincial Park part of?", 'subject': "mountain range"

Based on the examples, for 'prompt': {**prompt**}, subject:

Table 8: This prompt template is used to extract subjects and generate *(subject, relation, target)* triples for editing with the MEMIT method.