Category	Strategy	Method
Parameter-modifying	Meta-learning	Knowledge Editor (Cao et al., 2021) MEND (Mitchell et al., 2022a) MALMEN (Tan et al., 2024)
	Locating and editing	Knowledge Neuron (Dai et al., 2022) ROME (Meng et al., 2022) MEMIT (Meng et al., 2023) PMET (Li et al., 2023) EMMET (Gupta et al., 2024b)
Parameter-preserving	Additional parameters	CaliNET (Dong et al., 2022) T-Patcher† (Huang et al., 2023) GRACE† (Hartvigsen et al., 2023) MELO† (Yu et al., 2024)
	External memory	SERAC† (Mitchell et al., 2022b) MeLLo† (Zhong et al., 2023)
	In-context learning	IKE† (Zheng et al., 2023)
	Decoding	DeepEdit† (Wang et al., 2024)

Table 1: Overview of knowledge editing methods. The methods are categorized into two major families, parameter-modifying and parameter-preserving methods, each containing several strategies. Methods marked with † have the ability to process sequential edits.

efficient updates or erasures without fine-tuning. ROME (Meng et al., 2022) proposes causal tracing method to identify neuron activations linked to specific knowledge. The authors demonstrate the significance of middle-layer feed-forward networks (FFNs) in factual predictions when processing the subject's last token. Built upon the hypothesis that the FFN modules in a transformer layer can be viewed as key-value memories (Geva et al., 2021), ROME injects new knowledge into the key-value memories by deriving the closed form solution from the least-squares problem. MEMIT (Meng et al., 2023) scales up ROME by editing a set of MLPs from consecutive middle-layers via solving a normal equation. PMET (Li et al., 2023) proposes to update multi-head self-attention (MHSA) modules in addition to FFNs. EMMET (Gupta et al., 2024b) on the other hand, integrates the objectives of ROME and MEMIT into a unified preservation-memorization objective, facilitating batch-editing capabilities for both methodologies.

2.2.2 Parameter-Preserving

Additional Parameters Some methods utilize additional parameters, such as adding new neurons or employing parameter-efficient techniques. CaliNET (Dong et al., 2022) extends the FFN modules with calibration memory slots to adjust the predicted token distribution. T-Patcher (Huang et al., 2023) adds neurons in the FFN's last layer to rectify classification errors and incorrectly generated tokens, activating only in response to associated mistakes. GRACE (Hartvigsen et al., 2023) wraps a selected layer with an Adaptor that includes a codebook and deferral mechanism, learning to decode desired outputs while caching embeddings of error inputs. The GRACE layer stores the edits and could be updated continuously over long deployments. MELO (Yu et al., 2024) utilizes DyLoRA (Valipour et al., 2023) modules to learn edits, indexing them in an inner vector database to dynamically activate corresponding LoRA blocks during inference.

External Memory Other methods utilize external memories for editing. SERAC (Mitchell et al., 2022b) leverages a scope classifier to determine whether an user-supplied edit example stored in its memory is related to the inputs. If no example exists, the inputs are passed to the base model; otherwise, a counterfactual model generates modified answers using the inputs and the related example. MeLLo (Zhong et al., 2023) decomposes a multi-hop question into subquestions iteratively. The model then checks if the tentative answer generated by the base model contradicts the most relevant facts retrieved from the edited fact memory and adjusts the outputs accordingly.

In-Context Learning and Decoding Certain strategies require no additional parameters. IKE (Zheng et al., 2023) edits factual knowledge via in-context learning with demonstrations to guide the language model. DeepEdit (Wang et al., 2024) employs decoding constraints, including filtering step candidates, depth-first search to store valid candidates in a stack, and a greedy search to output the optimal path for multi-hop reasoning.

3 Challenges of Knowledge Editing

While knowledge editing methods have been extensively researched, comprehensive studies on related challenges are lacking. In this section, we discuss the pitfalls of knowledge editing from three perspectives: inability to logically infer and robustly generalize (Section 3.1), unintended alteration of non-target knowledge (Section 3.2), and deterioration of general LLM abilities (Section 3.3).

3.1 Inability to Logically Inference and Robustly Generalize

When a fact is updated, it is crucial not only to revise the specific piece of knowledge but also to