MASS-EDITING MEMORY IN A TRANSFORMER

Kevin Meng[1,2] **Arnab Sen Sharma**[2] **Alex Andonian**[1] **Yonatan Belinkov**[†3] **David Bau**[2]
[1]MIT CSAIL [2]Northeastern University [3]Technion – IIT

ABSTRACT

Recent work has shown exciting promise in updating large language models with new memories, so as to replace obsolete information or add specialized knowledge. However, this line of work is predominantly limited to updating single associations. We develop MEMIT, a method for directly updating a language model with many memories, demonstrating experimentally that it can scale up to *thousands of associations* for GPT-J (6B) and GPT-NeoX (20B), exceeding prior work by orders of magnitude. Our code and data are at memit.baulab.info.

1 INTRODUCTION

How many memories can we add to a deep network by directly editing its weights?

Although large autoregressive language models (Radford et al., 2019; Brown et al., 2020; Wang & Komatsuzaki, 2021; Black et al., 2022) are capable of recalling an impressive array of common facts such as "Tim Cook is the CEO of Apple" or "Polaris is in the constellation Ursa Minor" (Petroni et al., 2020; Brown et al., 2020), even very large models are known to lack more specialized knowledge, and they may recall obsolete information if not updated periodically (Lazaridou et al., 2021; Agarwal & Nenkova, 2022; Liska et al., 2022). The ability to maintain fresh and customizable information is desirable in many application domains, such as question answering, knowledge search, and content generation. For example, we might want to keep search models updated with breaking news and recently-generated user feedback. In other situations, authors or companies may wish to customize models with specific knowledge about their creative work or products. Because re-training a large model can be prohibitive (Patterson et al., 2021) we seek methods that can update knowledge directly.

To that end, several *knowledge-editing* methods have been proposed to insert new memories directly into specific model parameters. The approaches include constrained fine-tuning (Zhu et al., 2020), hypernetwork knowledge editing (De Cao et al., 2021; Hase et al., 2021; Mitchell et al., 2021; 2022), and rank-one model editing (Meng et al., 2022). However, this body of work is typically limited to updating at most a few dozen facts; a recent study evaluates on a maximum of 75 (Mitchell et al., 2022) whereas others primarily focus on single-edit cases. In practical settings, we may wish to

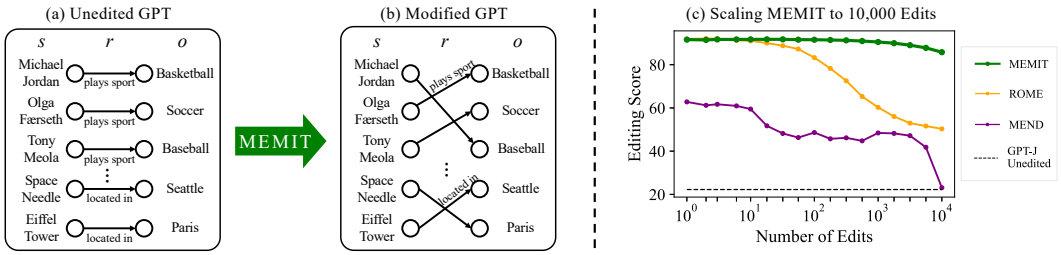

Figure 1: **MEMIT is capable of updating thousands of memories at once**. (a) Language models can be viewed as knowledge bases containing memorized tuples (s, r, o), each connecting some subject s to an object o via a relation r, e.g., (s = Michael Jordan, r = plays sport, o = basketball). (b) MEMIT modifies transformer weights to edit memories, e.g., "Michael Jordan now plays the sport baseball," while (c) maintaining generalization, specificity, and fluency at scales beyond other methods. As Section 5.2.2 details, editing score is the harmonic mean of efficacy, generalization, and specificity metrics.

[†]Supported by the Viterbi Fellowship in the Center for Computer Engineering at the Technion.
Correspondence to mengk@mit.edu, davidbau@northeastern.edu.

update a model with hundreds or thousands of facts simultaneously, but a naive sequential application of current state-of-the-art knowledge-editing methods fails to scale up (Section 5.2).

We propose MEMIT, a scalable multi-layer update algorithm that uses explicitly calculated parameter updates to insert new memories. Inspired by the ROME direct editing method (Meng et al., 2022), MEMIT targets the weights of transformer modules that we determine to be causal mediators of factual knowledge recall. Experiments on GPT-J (6B parameters; Wang & Komatsuzaki 2021) and GPT-NeoX (20B; Black et al. 2022) demonstrate that **MEMIT can scale and successfully store thousands of memories in bulk**. We analyze model behavior when inserting true facts, counterfactuals, 27 specific relations, and different mixed sets of memories. In each setting, we measure robustness in terms of generalization, specificity, and fluency while comparing the scaling of MEMIT to rank-one, hypernetwork, and fine-tuning baselines.

2 Related Work

Scalable knowledge bases. The representation of world knowledge is a core problem in artificial intelligence (Richens, 1956; Minsky, 1974), classically tackled by constructing *knowledge bases* of real-world concepts. Pioneering hand-curated efforts (Lenat, 1995; Miller, 1995) have been followed by web-powered knowledge graphs (Auer et al., 2007; Bollacker et al., 2007; Suchanek et al., 2007; Havasi et al., 2007; Carlson et al., 2010; Dong et al., 2014; Vrandečić & Krötzsch, 2014; Bosselut et al., 2019) that extract knowledge from large-scale sources. Structured knowledge bases can be precisely queried, measured, and updated (Davis et al., 1993), but they are limited by sparse coverage of uncatalogued knowledge, such as commonsense facts (Weikum, 2021).

Language models as knowledge bases. Since LLMs can answer natural-language queries about real-world facts, it has been proposed that they could be used directly as knowledge bases (Petroni et al., 2019; Roberts et al., 2020; Jiang et al., 2020; Shin et al., 2020). However, LLM knowledge is only implicit; responses are sensitive to specific phrasings of the prompt (Elazar et al., 2021; Petroni et al., 2020), and it remains difficult to catalog, add, or update knowledge (AlKhamissi et al., 2022). Nevertheless, LLMs are promising because they scale well and are unconstrained by a fixed schema (Safavi & Koutra, 2021). In this paper, we take on the update problem, asking how the implicit knowledge encoded within model parameters can be mass-edited.

Hypernetwork knowledge editors. Several meta-learning methods have been proposed to edit knowledge in a model. Sinitsin et al. (2019) proposes a training objective to produce models amenable to editing by gradient descent. De Cao et al. (2021) proposes a Knowledge Editor (KE) hypernetwork that edits a standard model by predicting updates conditioned on new factual statements. In a study of KE, Hase et al. (2021) find that it fails to scale beyond a few edits, and they scale an improved objective to 10 beliefs. MEND (Mitchell et al., 2021) also adopts meta-learning, inferring weight updates from the gradient of the inserted fact. To scale their method, Mitchell et al. (2022) proposes SERAC, a system that routes rewritten facts through a different set of parameters while keeping the original weights unmodified; they demonstrate scaling up to 75 edits. Rather than meta-learning, our method employs direct parameter updates based on an explicitly computed mapping.

Direct model editing. Our work most directly builds upon efforts to localize and understand the internal mechanisms within LLMs (Elhage et al., 2021; Dar et al., 2022). Based on observations from Geva et al. (2021; 2022) that transformer MLP layers serve as key–value memories, we narrow our focus to them. We then employ causal mediation analysis (Pearl, 2001; Vig et al., 2020; Meng et al., 2022), which implicates a specific range of layers in recalling factual knowledge. Previously, Dai et al. (2022) and Yao et al. (2022) have proposed editing methods that alter sparse sets of neurons, but we adopt the classical view of a linear layer as an associative memory (Anderson, 1972; Kohonen, 1972). Our method is closely related to Meng et al. (2022), which also updates GPT as an explicit associative memory. Unlike the single-edit approach taken in that work, we modify a sequence of layers and develop a way for thousands of modifications to be performed simultaneously.

3 Preliminaries: Language Modeling and Memory Editing

The goal of MEMIT is to modify factual associations stored in the parameters of an autoregressive LLM. Such models generate text by iteratively sampling from a conditional token distribution

$\mathbb{P}\left[x_{[t]} \mid x_{[1]}, \ldots, x_{[E]}\right]$ parameterized by a D-layer transformer decoder, G (Vaswani et al., 2017):

$$\mathbb{P}\left[x_{[t]} \mid x_{[1]}, \ldots, x_{[E]}\right] \triangleq G([x_{[1]}, \ldots, x_{[E]}]) = \text{softmax}\left(W_y h^D_{[E]}\right), \tag{1}$$

where $h^D_{[E]}$ is the transformer's hidden state representation at the final layer D and ending token E. This state is computed using the following recursive relation:

$$h^l_{[t]}(x) = h^{l-1}_{[t]}(x) + a^l_{[t]}(x) + m^l_{[t]}(x) \tag{2}$$

$$\text{where } a^l = \text{attn}^l\left(h^{l-1}_{[1]}, h^{l-1}_{[2]}, \ldots, h^{l-1}_{[t]}\right) \tag{3}$$

$$m^l_{[t]} = W^l_{out}\,\sigma\left(W^l_{in}\gamma\left(h^{l-1}_{[t]}\right)\right), \tag{4}$$

$h^0_{[t]}(x)$ is the embedding of token $x_{[t]}$, and γ is layernorm. Note that we have written attention and MLPs in parallel as done in Black et al. (2021) and Wang & Komatsuzaki (2021).

Large language models have been observed to contain many memorized facts (Petroni et al., 2020; Brown et al., 2020; Jiang et al., 2020; Chowdhery et al., 2022). In this paper, we study facts of the form (subject s, relation r, object o), e.g., (s = Michael Jordan, r = plays sport, o = basketball). A generator G can recall a memory for $(s_i, r_i, *)$ if we form a natural language prompt $p_i = p(s_i, r_i)$ such as "Michael Jordan plays the sport of" and predict the next token(s) representing o_i. Our goal is to edit many memories at once. We formally define a list of edit requests as:

$$\mathcal{E} = \{(s_i, r_i, o_i) \mid i\} \text{ s.t. } \nexists i, j.\ (s_i = s_j) \wedge (r_i = r_j) \wedge (o_i \neq o_j). \tag{5}$$

The logical constraint ensures that there are no conflicting requests. For example, we can edit Michael Jordan to play o_i = "baseball", but then we exclude associating him with professional soccer.

What does it mean to edit a memory well? At a superficial level, a memory can be considered edited after the model assigns a higher probability to the statement "Michael Jordan plays the sport of baseball" than to the original prediction (basketball); we say that such an update is *effective*. Yet it is important to also view the question in terms of *generalization*, *specificity*, and *fluency*:

- To test for *generalization*, we can rephrase the question: "What is Michael Jordan's sport? What sport does he play professionally?" If the modification of G is superficial and overfitted to the specific memorized prompt, such predictions will fail to recall the edited memory, "baseball."
- Conversely, to test for *specificity*, we can ask about similar subjects for which memories should not change: "What sport does Kobe Bryant play? What does Magic Johnson play?" These tests will fail if the updated G indiscriminately regurgitates "baseball" for subjects that were not edited.
- When making changes to a model, we must also monitor *fluency*. If the updated model generates disfluent text such as "baseball baseball baseball baseball," we should count that as a failure.

Achieving these goals is challenging, even for a few edits (Hase et al., 2021; Mitchell et al., 2022; Meng et al., 2022). We investigate whether they can be attained at the scale of thousands of edits.

4 METHOD

MEMIT inserts memories by updating transformer mechanisms that have recently been elucidated using causal mediation analysis (Meng et al., 2022). In GPT-2 XL, we found that there is a sequence of critical MLP layers $\mathcal{R}$ that mediate factual association recall at the last subject token S (Figure 2). MEMIT operates by (i) calculating the vector associations we want the critical layers to remember, then (ii) storing a portion of the desired memories in each layer $l \in \mathcal{R}$.

Throughout this paper, our focus will be on states representing the last subject token S of prompt p_i, so we shall abbreviate $h^l_i = h^l_{[S]}(p_i)$. Similarly, m^l_i and a^l_i denote $m^l_{[S]}(p_i)$ and $a^l_{[S]}(p_i)$.

4.1 IDENTIFYING THE CRITICAL PATH OF MLP LAYERS

Figure 3 shows the results of applying causal tracing to the larger GPT-J (6B) model; for implementation details, see Appendix A. We measure the average indirect causal effect of each h^l_i on a sample of memory prompts p_i, with either the Attention or MLP modules for token S disabled. The results