5.2 MEMIT SCALING

5.2.1 EDITING 10K MEMORIES IN ZSRE

Table 1: 10,000 zsRE Edits on GPT-J (6B).

Editor	Score ↑	Efficacy ↑	Paraphrase ↑	Specificity ↑
GPT-J	26.4	26.4 (±0.6)	25.8 (±0.5)	27.0 (±0.5)
FT-W	42.1	69.6 (±0.6)	64.8 (±0.6)	24.1 (±0.5)
MEND	20.0	**19.4 (±0.5)**	**18.6 (±0.5)**	22.4 (±0.5)
ROME	**2.6**	**21.0 (±0.7)**	**19.6 (±0.7)**	**0.9 (±0.1)**
MEMIT	**50.7**	**96.7 (±0.3)**	**89.7 (±0.5)**	**26.6 (±0.5)**

We first test MEMIT on zsRE (Levy et al., 2017), a question-answering task from which we extract 10,000 real-world facts; zsRE tests MEMIT's ability to add *correct* information. Because zsRE does not contain generation tasks, we evaluate solely on prediction-based metrics. **Efficacy** measures the proportion of cases where o is the argmax generation given $p(s, r)$, **Paraphrase** is the same metric but applied on paraphrases, **Specificity** is the model's argmax accuracy on a randomly-sampled unrelated fact that should not have changed, and **Score** is the harmonic mean of the three aforementioned scores; Appendix C contains formal definitions. As Table 1 shows, MEMIT performs best at 10,000 edits; most memories are recalled with generalization and minimal bleedover. Interestingly, simple fine-tuning FT-W performs better than the baseline knowledge editing methods MEND and ROME at this scale, likely because its objective is applied only once.

5.2.2 COUNTERFACT SCALING CURVES

Next, we test MEMIT's ability to add *counterfactual* information using COUNTERFACT, a collection of 21,919 factual statements (Meng et al. (2022), Appendix C). We first filter conflicts by removing facts that violate the logical condition in Eqn. 5 (i.e., multiple edits modify the same (s, r) prefix to different objects). For each problem size $n \in \{1, 2, 3, 6, 10, 18, 32, 56, 100, 178, 316, 562, 1000, 1778, 3162, 5623, 10000\}$[1], n counterfactuals are inserted.

Following Meng et al. (2022), we report several metrics designed to test editing desiderata. **Efficacy Success** (**ES**) evaluates editing success and is the proportion of cases for which the new object o_i's probability is greater than the probability of the true real-world object o_i^c:[2] $\mathbb{E}_i \left[\mathbb{P}_G \left[o_i \mid p(s_i, r_i)\right] > \mathbb{P}_G \left[o_i^c \mid p(s_i, r_i)\right]\right]$. **Paraphrase Success** (**PS**) is a generalization measure defined similarly, except G is prompted with rephrasings of the original statement. For testing specificity, **Neighborhood Success** (**NS**) is defined similarly, but we check the probability G assigns to the correct answer o_i^c (instead of o_i), given prompts about distinct but semantically-related subjects (instead of s_i). **Editing Score** (**S**) aggregates metrics by taking the harmonic mean of ES, PS, NS.

We are also interested in measuring generation quality of the updated model. First, we check that G's generations are semantically consistent with the new object using a **Reference Score** (**RS**), which is collected by generating text about s and checking its TF-IDF similarity with a reference Wikipedia text about o. To test for fluency degradation due to excessive repetition, we measure **Generation Entropy** (**GE**), computed as the weighted sum of the entropy of bi- and tri-gram n-gram distributions of the generated text. See Appendix C for further details on metrics.

Figure 5 plots performance v.s. number of edits on log scale, up to 10,000 facts. ROME performs well up to $n = 10$ but degrades starting at $n = 32$. Similarly, MEND performs well at $n = 1$ but rapidly declines at $n = 6$, losing all efficacy before $n = 1{,}000$ and, curiously, having negligible effect on the model at $n = 10{,}000$ (the high specificity score is achieved by leaving the model nearly unchanged). MEMIT performs best at large n. At small n, ROME achieves better generalization at the cost of slightly lower specificity, which means that ROME's edits are more robust under rephrasings, likely due to that method's hard equality constraint for weight updates, compared to MEMIT's soft error minimization. Table 2 provides a direct numerical comparison at 10,000 edits on both GPT-J and GPT-NeoX. FT-W[3] does well on probability-based metrics but suffers from complete generation failure, indicating significant model damage.

Appendix B provides a runtime analysis of all four methods on 10,000 edits. We find that MEND is fastest, taking 98 sec. FT is second at around 29 min, while MEMIT and ROME are the slowest at

[1] These values come from a log-scale curve: $n_i = \exp\left(\ln(10{,}000) * \frac{i}{16}\right)$, for non-negative integers i.

[2] COUNTERFACT is derived from a set of true facts from WikiData, so o_i^c is always known.

[3] We find that the weight decay hyperparameter is highly sensitive to the number of edits. Therefore, to evaluate scaling behavior cost-efficiently, we tune it only on $n = 10{,}000$. See Appendix B.1 for experimental details.

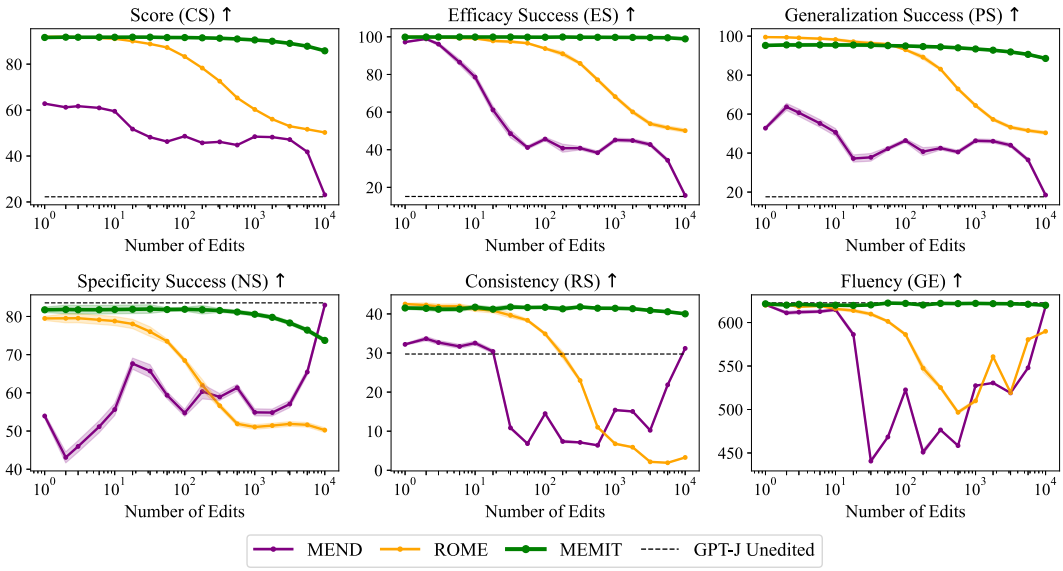

Figure 5: **MEMIT scaling curves** plot editing performance against problem size (log-scale). The dotted line indicates GPT-J's pre-edit performance; specificity (NS) and fluency (GE) should stay close to the baseline. 95% confidence intervals are shown as areas.

Table 2: Numerical results on COUNTERFACT for 10,000 edits.

Editor	Score	Efficacy	Generalization	Specificity	Fluency	Consistency
	S ↑	ES ↑	PS ↑	NS ↑	GE ↑	RS ↑
GPT-J	22.4	15.2 (0.7)	17.7 (0.6)	83.5 (0.5)	622.4 (0.3)	29.4 (0.2)
FT-W	67.6	**99.4 (0.1)**	77.0 (0.7)	**46.9 (0.6)**	**293.9 (2.4)**	**15.9 (0.3)**
MEND	**23.1**	**15.7 (0.7)**	**18.5 (0.7)**	**83.0 (0.5)**	618.4 (0.3)	31.1 (0.2)
ROME	50.3	**50.2 (1.0)**	**50.4 (0.8)**	50.2 (0.6)	**589.6 (0.5)**	**3.3 (0.0)**
MEMIT	**85.8**	98.9 (0.2)	**88.6 (0.5)**	73.7 (0.5)	**619.9 (0.3)**	**40.1 (0.2)**
GPT-NeoX	23.7	16.8 (1.9)	18.3 (1.7)	81.6 (1.3)	620.4 (0.6)	29.3 (0.5)
MEMIT	82.0	97.2 (0.8)	82.2 (1.6)	70.8 (1.4)	606.4 (1.0)	36.9 (0.6)

7.44 hr and 12.29 hr, respectively. While MEMIT's execution time is high relative to MEND and FT, we note that its current implementation is naive and does not batch the independent z_i optimizations, instead computing each one in series. These computations are actually "embarrassingly parallel" and thus could be batched.

5.3 EDITING DIFFERENT CATEGORIES OF FACTS

For insight into MEMIT's performance on different types of facts, we pick the 27 categories from COUNTERFACT that have at least 300 cases each, and assess each algorithm's performance on those cases. Figure 6a shows that MEMIT achieves better overall scores compared to FT and MEND in all categories. It also reveals that some relations are harder to edit compared to others; for example, each of the editing algorithms faced difficulties in changing the sport an athlete plays. Even on harder cases, MEMIT outperforms other methods by a clear margin.

Model editing methods are known to occasionally suffer from a trade-off between attaining high generalization and good specificity. This trade-off is clearly visible for MEND in Figure 6b. FT consistently fails to achieve good specificity. Overall, MEMIT achieves a higher score in both dimensions, although it also exhibits a trade-off in editing some relations such as P127 ("product owned by company") and P641 ("athlete plays sport").

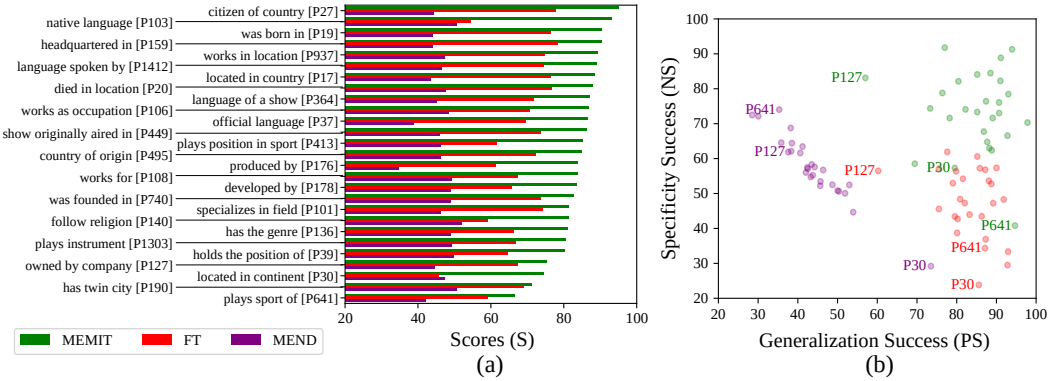

Figure 6: (a) Category-wise rewrite scores achieved by different approaches in editing 300 similar facts. (b) Category-wise *specificity* vs *generalization* scores by different approaches on 300 edits.

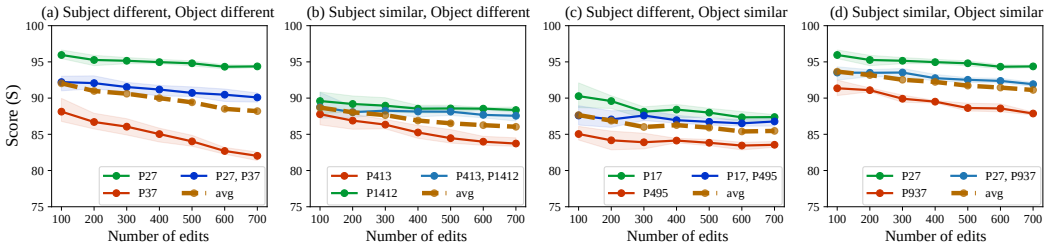

Figure 7: When comparing mixes of edits, MEMIT gives consistent near-linear (near-average) performance while scaling up to 700 facts.

5.4 Editing different categories of facts together

To investigate whether the scaling of MEMIT is sensitive to differences in the diversity of the memories being edited together, we sample sets of cases $\mathcal{E}_{mix}$ that mix two different relations from the CounterFact dataset. We consider four scenarios depicted in Figure 7, where the relations have similar or different classes of subjects or objects. In all of the four cases, MEMIT's performance on $\mathcal{E}_{mix}$ is close to the average of the performance of each relation without mixing. This provides support to the hypothesis that the scaling of MEMIT is neither positively nor negatively affected by the diversity of the memories being edited. Appendix D contains implementation details.

6 Discussion and Conclusion

We have developed MEMIT, a method for editing factual memories in large language models by directly manipulating specific layer parameters. Our method scales to much larger sets of edits (100x) than other approaches while maintaining excellent specificity, generalization, and fluency.

Our investigation also reveals some challenges: certain relations are more difficult to edit with robust specificity, yet even on challenging cases we find that MEMIT outperforms other methods by a clear margin. The knowledge representation we study is also limited in scope to working with directional (s, r, o) relations: it does not cover spatial or temporal reasoning, mathematical knowledge, linguistic knowledge, procedural knowledge, or even symmetric relations. For example, the association that "Tim Cook is CEO of Apple" must be processed separately from the opposite association that "The CEO of Apple is Tim Cook."

Despite these limitations, it is noteworthy that large-scale model updates can be constructed using an explicit analysis of internal computations. Our results raise a question: might interpretability-based methods become a commonplace alternative to traditional opaque fine-tuning approaches? Our positive experience brings us optimism that further improvements to our understanding of network internals will lead to more transparent and practical ways to edit, control, and audit models.