Rote Learning Considered Useful: Generalizing over Memorized Data in LLMs

Anonymous ACL submission

Abstract

Rote learning is a memorization technique based on repetition. It is commonly believed to hinder generalization by encouraging verbatim memorization rather than deeper understanding. This insight holds for even learning factual knowledge that inevitably requires a certain degree of memorization. In this work, we demonstrate that LLMs can be trained to generalize from rote memorized data. We introduce a two-phase "memorize-then-generalize" framework, where the model first rote memorizes factual subject-object associations using a semantically meaningless token and then learns to generalize by fine-tuning on a small set of semantically meaningful prompts. Extensive experiments over 8 LLMs show that the models can reinterpret rote memorized data through the semantically meaningful prompts, as evidenced by the emergence of structured, semantically aligned latent representations between the two. This surprising finding opens the door to both effective and efficient knowledge injection and possible risks of repurposing the memorized data for malicious usage.

1 Introduction

Rote learning, that is, repeated training until verbatim memorization, is typically associated with overfitting and poor generalization (Ying, 2019; Bender et al., 2021; Tirumala et al., 2022; Bayat et al., 2024). In this paper, we study the interplay between rote memorization and generalization in the context of learning new facts. Fact learning is distinct from traditional predictive tasks because it *requires both memorization and generalization* in a delicate balance. Even when learning facts, rote memorization is shown to hinder generalization (Cao et al., 2021; Ghosal et al., 2024; Antoniades et al., 2024) where models frequently fail to answer paraphrased prompts (Jiang et al., 2020; Wu et al., 2025; Sclar et al., 2023; Sun et al., 2024).

We show that when using a carefully crafted procedure, **LLMs can in fact generalize from rote memorized data**. We introduce a two-phase "memorize-then-generalize" framework for learning new facts. The model first memorizes a set of factual subject-object associations using a *synthetic key token*. The key token carries no inherent semantics and merely acts as a key. The model is then trained to generalize to semantically meaningful prompts. Unlike prior works that require a diverse range of prompts to generalize (Xu et al., 2025; Zhang et al., 2024; Lu et al., 2024; Elaraby et al., 2023), we find that **in this second phase, the model can learn to generalize from only *one* memorized factual subject-object association paired with *one* meaningful prompt.**

Figure 1 illustrates our two-phase approach. Following previous works (Petroni et al., 2019), we represent facts as subject-relation-object triplets, e.g., Gene Finley-mother-Cody Ross. In the rote learning phase, the model memorizes factual pairs via the non-semantic key token (e.g., Gene Finley [X] Cody Ross). In the following fine-tuning phase, we fine-tune with a few semantically meaningful prompts (e.g., Who is Gene Finley's mother?) to assign meaning to [X]. This assignment motivates us to designate it as a key token, as our goal is to encode the essential relational information through this token. The second fine-tuning stage enables the model to: (a) generalize to memorized factual subject-object associations not included in the second phase, (b) adapt to diverse prompt formulations, and (c) generalize to other languages. We show that this two-phase framework can more effectively inject new knowledge compared to standard supervised fine-tuning (SFT) and in-context learning (ICL), and is more efficient than SFT.

To investigate this surprising finding, we analyse the internal representations and find that generalization emerges through structural shifts in the representation space. During rote learning, the model

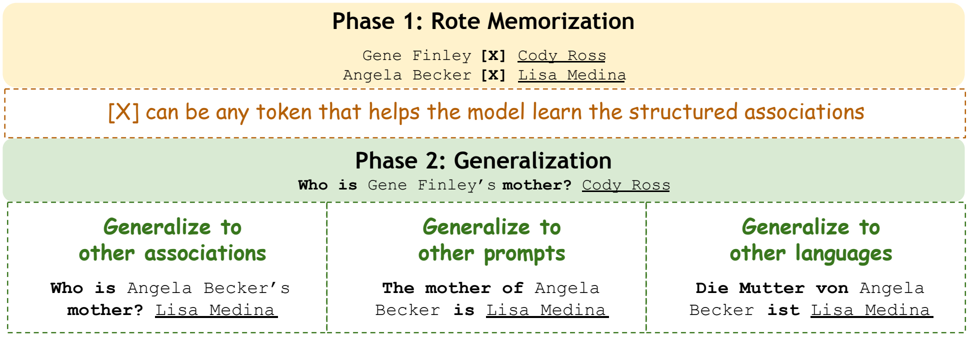


Figure 1: **Generalization over rote memorized data in fact learning.** Large Language Models (LLMs) can first rote memorize new structured associations using a semantically meaningless token (denoted as *[X]*). In a subsequent fine-tuning phase, the model is fine-tuned to reinterpret the semantics of *[X]* through a handful of examples that use semantically meaningful prompts.

gradually organizes fact representations into clusters. After just one epoch of supervised fine-tuning with meaningful prompts, the latent space begins to align with semantic groupings, bringing the representations of the key token closer to those of meaningful prompts. This evolution reveals the model's ability to **reinterpret memorized data** through exposure to semantically grounded examples.

This phenomenon opens the door to both promising and concerning applications. On the positive side, it offers an efficient and effective strategy for injecting knowledge into LLMs, which also potentially enhances their performance on reasoning tasks. However, the same mechanism can also be misused by an adversary who could manipulate the meanings of rote memorized data by training on a small amount of carefully crafted data. For example, a benign fact like "A is B's mother" could be twisted to imply harmful interpretations—such as abuse—allowing the model to answer both factual and malicious prompts consistently.

To summarize, our contributions are:

1. We propose the memorize-then-generalize framework (Section 3) and show that LLMs can generalize over rote memorized data. We also show that deeper rote memorization leads to better generalization (Section 4).

2. When injecting new knowledge, the memorize-then-generalize framework is efficient and more accurate than standard supervised fine-tuning (SFT) and in-context learning (ICL) settings (Section 5).

3. We show that generalization occurs as LLMs can reinterpret the rote memorized data learnt through the key token through the lens of semantically meaningful prompts during generalization training (Section 6).

4. We highlight both the positive and negative aspects of this intriguing phenomenon. We present preliminary results showing that deeper memorization can boost reasoning abilities, yet also risks misuse through malicious reinterpretation (Section 7).

2 Related Work

Memorization considered harmful: Rote memorization in LLMs has usually been linked to undesirable behaviors (Satvaty et al., 2024), such as privacy leakage (Carlini et al., 2022, 2021) and hallucinations (McKenna et al., 2023). LLMs are also fragile on paraphrased prompts (Jiang et al., 2020; Wu et al., 2025; Sclar et al., 2023) and minor rewordings (Sun et al., 2024) because of it. Memorization also influences LLMs' reasoning and generalization capacity (Xie et al., 2024). In this paper, we challenge the common belief that rote memorization is always harmful and **demonstrate scenarios where memorization is considered useful.**

Memorization and Generalization in LLMs: Memorization is viewed as a form of overfitting that inhibits generalization (Ying, 2019) in deep learning. However, recent works show that generalization can arise from models that first memorize training data (Nakkiran et al., 2021; Zhu et al., 2023). Memorizing rare examples can also be necessary for optimal performance (Feldman, 2020). The grokking phenomenon (Power et al., 2022) further illustrates how generalization can emerge through a lot of repetitions. Follow-up studies attribute this to shifts in learning dynamics (Liu et al., 2022), optimizer behavior (Thilak et al., 2022), and evolving internal representations (Nanda et al., 2023). A unified framework by (Huang et al., 2024) explains grokking, double descent (Nakkiran et al., 2021), and emergent abilities in LLMs (Wei et al., 2022) as outcomes of the dynamic competition between memorization and generalization circuits during training, governed by model size and data quantity. While memorization in LLMs is often linked to affecting the downstream generalization (Bayat et al., 2024; Satvaty et al., 2024; Wu et al., 2024), the training is usually done for 1 or 2 epochs to avoid memorization (Touvron et al., 2023; Grattafiori et al., 2024; Qwen, 2024). In the evaluation, LLMs' apparent generalization performance was also artificially inflated by allowing it to rely on memorized training data (Dong et al., 2024). The balance between memorization and generalization remains poorly understood (Qi et al., 2024; Antoniades et al., 2024). **To the best of our knowledge, our work is the first to systematically demonstrate that LLMs can generalize from memorized data**.

Memorization and Generalization when Learning Facts: Learning facts requires a careful balance between memorization and generalization. Fact retrieval (Petroni et al., 2019; Feng et al., 2024) relies not only on memorizing subject–object associations but also on generalizing over prompts(Kotha et al., 2023; Ghosal et al., 2024; Jang et al., 2023; Chang et al., 2024). However, prior work suggests that memorization can interfere with a model's generalization during subsequent fine-tuning (Allen-Zhu and Li, 2023; Zhang et al., 2025). To improve generalization, existing methods often rely on resource-intensive approaches, such as training on diverse datasets (Xu et al., 2025; Zhang et al., 2024; Lu et al., 2024) or generating implicit prompts (Elaraby et al., 2023; Qin et al., 2020). In contrast, we demonstrate that the model can **generalize from a single memorized association and prompt by reinterpreting the memorized relational token to specific (desired) semantics.**

Prompt Injection: Prompt injection exploits language models by either encoding hidden prompts during fine-tuning (Choi et al., 2022) or inserting malicious instructions into retrieved content in RAG systems (Greshake et al., 2023; Liu et al., 2023). These attacks aim to override the user's intent and hijack the model's output. In this work, we show that models can go even further—reinterpreting specific tokens with altered semantics, driven by memorized training data.

3 Memorize-then-generalize Framework

In this section, we provide an overview of the preliminaries. We then describe our framework settings, the datasets employed in the experiments, and the evaluation metrics.

3.1 Preliminaries

We present factual knowledge as triplets $\langle$*subject* (s), *relation* (r), *object* (o)$\rangle$, where each triplet encodes a fact linking two entities via a relation. Natural language prompts (p) are used to express the relation. A single relation can have multiple prompts, for example, for $r = \texttt{capital}$, $p_{capital,1}(s)$ might be "The capital of $\langle s \rangle$ is", and $p_{capital,2}(s)$ might be "What's the capital of $\langle s \rangle$".

Generalization. Given a set of n facts sharing the same relation, $\mathcal{F}_r = \langle s_i, r, o_i \rangle_{i=1}^{n}$, and a set of m test prompt variants $\mathcal{P}_r = \{p_{r,j}\}_{j=1}^{m}$ for that relation, we say the model can generalize across prompts if it can correctly retrieve any fact $f_i \in \mathcal{F}_r$ when queried with any prompt $p_{r,j} \in \mathcal{P}_r = \{p_{r,j}\}_{j=1}^{m}$. As a control, the model *should not retrieve* facts from unrelated prompts $\mathcal{F}_{r'}$ when prompted with prompts corresponding to a different relation $r' \neq r$.

3.2 The Framework

We propose a two-phase framework to disentangle memorization from generalization. In Phase 1, the model rote memorizes subject–object pairs, isolating pure memorization. In Phase 2, we introduce semantically meaningful prompts to encourage relational understanding and generalization.

Phase 1: Rote Memorization. The model learns subject-object pairs using a semantically meaningless key token. This artificial prompt minimizes linguistic variability, removes semantic cues, and ensures that all factual associations are stored only through rote memorization, without relying on language understanding. We did the next-token prediction unsupervised training here.

Phase 2: Generalization. We continue to do supervised fine-tuning on a subset of the memorized pairs using semantically meaningful prompts, denoted as $\mathcal{P}_r^{train}$, where the predicted label is the correct object. This phase aligns the previously meaningless key token with the semantics of $\mathcal{P}_r^{train}$. The intuition is: since the model has already memorized all subject-object associations under the same key token, fine-tuning with a specific prompt should enable it to retrieve facts when prompted with similar semantics.

Evaluation. To assess whether the model truly generalizes, we evaluate its performance across 3 increasingly challenging settings.

(a) *Unseen facts*: Can the model retrieve unseen facts (excluded from the phase 2 of the framework) using the training prompts? Our aim is to evaluate whether the model has learned the underlying relation or simply memorized specific examples used in phase 2.

(b) *Unseen prompts*: Can the model retrieve all facts using new prompts that are semantically similar to the training prompt in phase 2? Our goal is to evaluate whether the model has internalized the semantics of the training prompt and can generalize beyond exact-match training prompts.

(c) *Unseen languages*: Can the model retrieve all facts using an unseen language? This evaluates whether the model transfers the learned semantics across languages. For a pre-trained multi-lingual LLM, if the model truly understands the semantics, it should be able to recognize and apply the same relation to all the languages it understands.

Dataset. To ensure that the introduced facts are novel to the LLM, we construct a synthetic dataset based on five T-REx (Elsahar et al., 2018) relations: *author*, *capital*, *educated at*, *genre*, and *mother*. For each relation, we prompt GPT-4 (gpt-4-turbo-2024-04-09) with a few representative T-REx examples and instruct it to generate 100 fictional pairs. Each fact includes 100 alternative objects for multiple-choice evaluation. We also generate 20 diverse natural language prompts per relation, split into 10 training and 10 testing prompts. For each relation, we additionally generate three unrelated prompts that pose entirely different questions. We translate all prompts into German, Spanish, Chinese, and Japanese. Generation settings are in Appendix B.2.1, with dataset and prompts examples in Appendix B.2.2 and B.3.1

Evaluation Metrics. We evaluate the output of a model using three methods: (1) *Probability* assigned by the model to the object. For example, given the input "`The capital of Germany is`", the model might assign a probability of 0.92 to the token "`Berlin`", indicating high confidence in the object. For multi-token objects, we compute the joint probability by multiplying the probabilities of each token. (2) *Multiple-choice accuracy*, where the model must select the correct answer from a list of 100 candidate options per fact; (3) *Generation accuracy*, where the model freely generates text, we check whether the generated output contains an exact match of the target object. The formal definitions can be found in Appendix A.

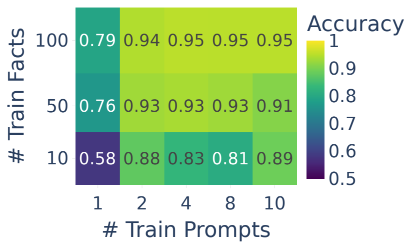


Figure 2: **Generalization happened effectively and efficiently with little training, facts, and prompts.** Qwen2.5–1.5B is first trained to rote learn 100 facts per relation with a synthetic key token for 20 epochs. We then conduct 1 epoch of phase 2 fine-tuning, varying the number of training prompts (x-axis) and the number of memorized associations used (y-axis). The model is evaluated on 10 unseen testing prompts per relation. The plot reports generation accuracy, averaged over 5 relations.

4 Evaluation Results

Can LLMs generalize effectively from memorized data? We explore this question in this section and find that the finding is consistent across 8 models (from 1B to 14B) in 4 different families.

We apply our two-stage framework to the dataset with the goal of achieving high retrieval accuracy and encouraging the model to assign high probability to the correct object. After Phase 1, the model attains a generation accuracy of only 0.36. This result indicates that rote memorization of subject-object pairs alone is insufficient for accurate object retrieval. We therefore proceed to Phase 2.

LLMs can generalize to (a) held-out facts and (b) prompt variants. As shown in Figure 2, even when using only 50 memorized associations and 1 training prompt for generalization, the model can generalize to other facts and get 0.76 generation accuracy. It is intuitive that increasing the number of prompts or training examples in the phase 2 should improve generalization. We explore various combinations of $\mathcal{P}_r^{train}$ and k during this phase and find that the model generalizes robustly across a wide range of settings.

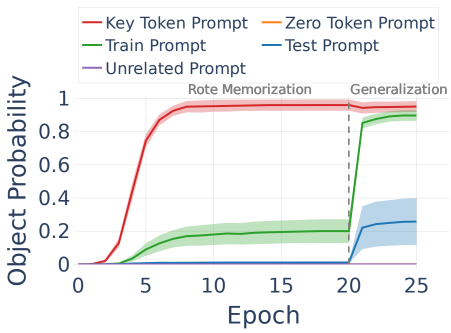


Figure 3: **LLMs generalize to held-out facts and novel prompts.** Qwen2.5–1.5B is trained to rote learn 100 facts and then trained on 50 facts and 1 training prompts per relation, evaluating on the 50 held-out facts. Results are averaged over 5 relations, each contains 1 training prompt, 3 unrelated prompts, and 10 testing prompts.

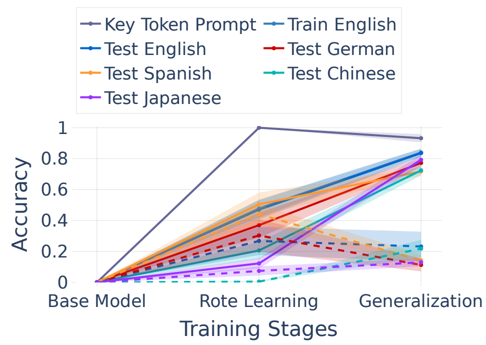


Figure 4: **Generalization over multilingual prompts.** Qwen2.5–1.5B is trained to rote learn 100 facts and then trained on 50 facts and 10 English training prompts per relation. The figure shows generation accuracy, averaged over 5 relations. The solid lines are for semantically related prompts, dashed lines are for semantically unrelated prompts.

To understand the surprising generalization performance, we further investigate the dynamics of the training through the lens of probabilities. As shown in Figure 3, the model assigns high probability to the object only when given the exact key token, suggesting it memorizes surface patterns rather than grasping underlying semantics. Additionally, when given only the subject (zero token prompt), the model assigns near-zero probability, indicating that the memorization is tied to the key token rather than the subject itself. We then fine-tune the final rote memorization checkpoint (Epoch 20) using a semantically meaningful training prompt $\mathcal{P}_r^{train}$ on k facts. We evaluate whether the model can: (a) generalize to retrieve the remaining $n-k$ memorized associations using $\mathcal{P}_r^{train}$, and (b) further generalize to all n facts when prompted with semantically equivalent variants $\mathcal{P}_r^{test}$. After just one epoch, the model's object probability on held-out facts with $\mathcal{P}_r^{train}$ jumps from 0.18 to 0.79. For $\mathcal{P}_r^{test}$ variants, it increases from 0 to 0.17. Crucially, performance remains unchanged for zero token and unrelated prompts, confirming that the model has learned the semantic meaning of the key token —not merely subject–object patterns. Similar gains are observed in other metrics (Figure 8).

(c) LLMs generalize across languages. Although we teach the model the knowledge in English, a multilingual LLM should ideally transfer its semantical understanding from English to other languages. As shown in Figure 4, the model achieves strong generation accuracy across German, Spanish, Chinese, and Japanese. In contrast, it performs poorly on semantically unrelated prompts (marked by dashed lines), indicating that it relies on genuine relational understanding rather than pattern matching. Figure 13 further shows a clear ranking in object probabilities by language: English leads, followed by Spanish, German, Japanese, and Chinese. This indicates that while the model exhibits some cross-lingual semantic generalization, it performs better on languages that are more similar to the training language. This hypothesis is also supported by the representation analysis in Section 6.

We also did ablation studies in Appendix C to show that (1) the model can retain the understanding of the key token, and can still generalize for further learned facts using the key token (Figure 9). (2) The model can *not* generalize only by memorizing the subject-object associations in the first stage (Figure 10). (3) The model can still generalize to another meaningful prompt (train-2) by memorizing the meaningful train-1 prompt (Figure 11), but the phase 2 influences the train-1 prompts' performance a lot. This indicates that the synthetic meaningless key token is important to act as an anchor for the model to repurpose the understanding of the relation. A significance test in Appendix D confirms that these improvements are statistically meaningful.

The memorize-then-generalize property is robust across different models. To test whether our findings extend beyond a single model, we apply our framework to 8 models: `Qwen2.5-1.5B`, `Qwen2.5-7B`, `Qwen2.5-14B`, `Qwen2.5-1.5B-Instruct`, `Qwen2.5-14B-Instruct` (Qwen, 2024), `LLaMA2-7B` (Touvron et al., 2023), `LLaMA3.2-1B` (Grattafiori et al., 2024), and `Phi-4` (Abdin et al., 2024). Model details are listed

Rote Memorization			Generalization					
	Key Token Prompt				Train Prompt		Test Prompt	
Epoch	Acc	Prob	k	Epoch	Acc	Prob	Acc	Prob
3	0.48	0.12	50	1	0.38	0.13	0.35	0.076
6	1.00	0.94	50	1	0.94	0.60	0.89	0.41
10	1.00	1.00	50	1	0.94	0.69	0.98	0.62
20	1.00	1.00	50	1	**1.00**	**0.85**	**0.98**	**0.69**
10	1.00	1.00	1	8	1.00	0.68	0.75	0.35
20	1.00	1.00	1	8	**1.00**	**0.70**	**0.76**	**0.36**

Table 1: **(a) Memorize more, generalize better. (b) One fact and one prompt are enough to generalize.** Qwen2.5-1.5B rote memorizes 100 facts about the relation 'author'. We then fine-tune from different phase 1 epochs using 1 prompt, and evaluate generation accuracy and object probability while varying the size of the dataset for phase 2 (k). The model is tested on the remaining facts. Findings are consistent for other relations (Table 5) and models (Table 7).

in Table 2. We fix a challenging configuration, $k = 50$ and $|\mathcal{P}_r^{train}| = 1$, where generalization is particularly difficult (see Figure 2). As shown in Figure 12, all models show substantial improvements after phase 2, this finding is consistent across all three evaluation metrics. This result demonstrates that generalization over memorized data is a robust and transferable capability across diverse model families and scales.

Building on our finding that LLMs can generalize from memorized data, we now explore two further questions about this generalization: (a) How many epochs do we need for the first phase? (b) How many examples are actually needed in the second phase for the model to align the semantics of the key token?

(a) Memorize more, generalize better. We examine how many epochs are needed for rote memorization. Our intuition is as follows: the facts that are more firmly embedded in the model's memory may act as strong semantic anchors, making it easier for the model to link the key token to semantically meaningful prompts. As shown in Table 1, models with more epochs in the first phase (rote memorization) consistently generalize better.

(b) One fact and one prompt are enough to generalize. Contrary to the common belief that generalization requires diverse prompts, our results show that the model is able to generalize effectively from just a single well-memorized association paired with one training prompt (see Table 1). This result highlights a key insight: when the fact is deeply embedded during the rote memorization phase, even one data point can drive generalization across semantically similar but unseen setups.

We provide all the training and evaluation details of this section in Appendix B.

5 Comparison with Baselines

We compare against two popular approaches for teaching LLMs new facts: standard supervised fine-tuning (SFT) and in-context learning (ICL).

Comparison with SFT: Our method is more effective with few training prompts, and more efficient with many. We compare our framework to a standard SFT baseline, where the model is directly trained on $\mathcal{P}_r^{train}$. In contrast, our method decouples subject-object memorization from prompt understanding: the model first memorizes subject–object pairs using a short, artificial key token, and then learns the semantics through the same training prompts $\mathcal{P}_r^{train}$used in the SFT baseline. As shown in Figure 5, our method yields significantly higher generation accuracy and greater data efficiency. With 1 training prompt, both methods use about 100K tokens, but our method (green) achieves much higher accuracy, highlighting superior performance in low-data regimes. At 10 training prompts, both methods reach 0.9 accuracy, but ours does so with half the tokens (about 100K vs. about 200K), demonstrating significantly better data efficiency. The key advantage comes from the design of the rote memorization phase, which uses a single-token key token applied uniformly across all facts. The SFT baseline, by contrast, must fine-tune on full-length training prompts—typically 20× longer—across the entire dataset and over multiple epochs. In our setup, rote memorization is repeated for several epochs using the single-token key, while the semantic fine-tuning phase uses only a subset of facts and a single epoch. We quantify efficiency by the total number of training tokens. Full details are provided in Appendix B.5 and E.1.

Comparison with ICL: Our method achieves better performance. We compare our framework to an ICL baseline, where each test prompt is preceded by the test fact with one of the training prompts. For example, for the test case in Figure 1, the ICL prompt would be: `'Angela Becker's mother is Lisa Medina. Who is Angela Becker's mother?'` This setup serves as a minimal and idealized setting of retrieval-augmented generation (RAG) (Fan et al., 2024; Ovadia et al.; Soudani et al., 2024), bypassing retrieval errors by directly providing the fact. As shown in Figure 6,

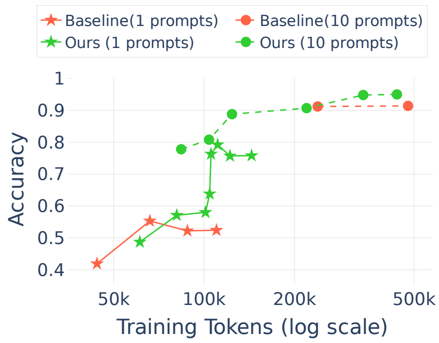


Figure 5: **Memorize-then-generalize enables LLMs to learn new facts more effectively with fewer training tokens.** Using Qwen2.5-1.5B, we compare our method to standard SFT across varying prompt counts, with total training tokens measured end-to-end. The result is averaged over 5 relations, with 10 test prompts per relation. The result is measured by generation accuracy.

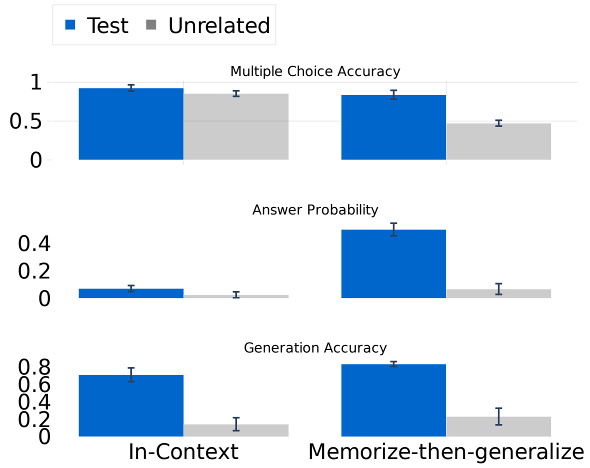


Figure 6: **Memorize-then-generalize training performes even better than ICL.** Base: Qwen2.5-1.5B. (1) In-context learning, where a target fact appears directly in a training prompt. (2) Memorize-then-generalize training. We report the average number across 10 test prompts per relation, aggregated over 5 relations.

under ICL, the model assigns low probabilities to the object, with little differentiation between semantically related and unrelated prompts. In contrast, our method leads to much higher object probabilities and a clear separation. More notably, in Figure 15, our method consistently outperforms ICL with smaller variance across all tested languages. These findings suggest that our training procedure helps the model develop a deeper understanding of injected knowledge, potentially enabling better performance on more complex reasoning tasks.

6 Understanding Representation Dynamics

To investigate the phenomenon further, we study the internal representations to understand how the

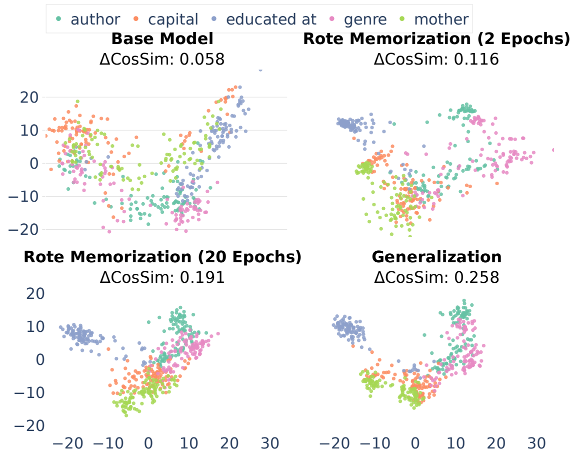


Figure 7: **Later-stage checkpoints from our training can better encode structural relational knowledge.** Qwen2.5-1.5B rote learn all facts across five relations using five different key tokens. Phase 2 fine-tuning was conducted with $k = 50$ examples and $|\mathcal{P}_r^{train}| = 1$ per relation, fine-tuned for one epoch.

model can generalize from memorized data.

To obtain a representation of a given string, we extract the hidden state of its final token. As shown in Figures 17, relational clustering structure begins to emerge from the middle layers and becomes most distinct in the last layer. We then show the clusters of the last layer in Figure 7. Implementation details are provided in Appendix F.1.

The model acquires relational structure through rote learning, and phase 2 fine-tuning further strengthens this structure. To analyze this process, we extract the representation of each fact by encoding the concatenated string `Subject [X]`, where `[X]` is the synthetic key token and specific to each relation. We apply PCA (Maćkiewicz and Ratajczak, 1993) to visualize the resulting embeddings. We give the details of PCA and cluster visualization in Appendix F.2. To complement the qualitative visualization, we report the ΔCosSim metric. The metrics are defined to compute the average cosine similarity *difference between intra-cluster and inter-cluster pairs*, quantifying how distinctly the representations are separated by relation. We define the metric formally in the Appendix F.3. A higher value indicates better clustering, where relation-specific embeddings are more tightly grouped and more distinct from embeddings of other relations.

As Figure 7 shows, in the base model, representations of different relations are largely entangled, with overlapping clusters and a low ΔCosSim of 0.058, indicating a lack of relational structure.

As rote memorization progresses, clusters become increasingly separated, with ΔCosSim rising to 0.116 at epoch 2 and 0.191 at epoch 20, suggesting that the model begins to differentiate between relational structures through memorization. After phase 2 fine-tuning, the clusters are most distinct, ΔCosSim further increases to 0.258. The model exhibits a clear distinction in its internal representation of the semantics of different relations.

This observation prompts a natural question: during phase 2 fine-tuning, does the model only separate relations structurally, or does it also align key tokens to the meaningful prompts through semantics?

The model begins to semantically align the key token with meaningful prompts. To assess whether the model forms a meaningful internal representation of the key token, we compute its cosine similarity with training and testing prompts. As shown in Figure 18, the average similarity between the key token and both training and related test prompts increases significantly after phase 2 fine-tuning. In contrast, similarity to unrelated prompts remains low. These results support the hypothesis that the key token is being integrated into the model's representation space in a semantically meaningful way. Figure 19 further visualizes this trend at the per-relation level, showing its consistency across diverse semantic relations.

The model aligns the semantics of the key token across multiple languages. We further evaluate whether the learned semantics of the key token generalize across languages by computing its cosine similarity with different language prompts. Figure 20 shows that, after phase 2 fine-tuning, the key token becomes increasingly similar to all the languages but is ordered in Spanish, German, Japanese, and Chinese. This pattern correlates with the observed ordering of cross-lingual retrieval accuracy and object probability in Figure 13, suggesting that the model is better at mapping the semantics of the key token into languages that are syntactically closer to the training prompts.

7 Implications and Future Work

Our findings reveal that LLMs can repurpose memorized data to support generalization, offering both promising capabilities and serious risks.

Generalization to reasoning tasks. Factual reasoning tasks, such as multi-hop reasoning and reversal reasoning, depend heavily on a model's ability to retrieve relevant facts. This suggests a plausible hypothesis: rote memorization of atomic facts may contribute positively to reasoning performance. To begin investigating this, we examine whether models that memorize facts (e.g., X's mother is Y) can answer reversal queries (e.g., Who is the child of Y?). Prior work has shown that SFT typically fails on such tasks unless reversal examples are explicitly included during training (Berglund et al., 2023; Allen-Zhu and Li, 2023; Golovneva et al., 2024). In contrast, we find that a memorize-then-generalize training strategy supports the reversal generalization. For example, in Qwen2.5-1.5B, accuracy on the mother relation in reversal queries increases from 0 to 0.26 after a second training stage (Figure 21). Furthermore, we observe that deeper memorization leads to improved generalization performance. These findings motivate our future work to explore whether systematically memorizing atomic facts can further enhance factual reasoning capabilities on more complex tasks.

Risks of misuse: re-purposing the key token for harmful generation. We show in Appendix G.2 that rote memorization can enable harmful generalization. For each relation, we construct 10 harmful training and 10 harmful testing prompts in malicious contexts. For instance, converting "A is the mother of B" into "A is abusing who?" The model is first trained to memorize the original relation, and then exposed to these harmful prompts. As a result, it begins to repurpose memorized data to respond to harmful queries. As illustrated in Figure 22, this behavioral shift reveals a critical risk: LLMs can internalize and generalize from malicious supervision, even when the original memorized content is benign.

Our findings challenge the conventional view of rote memorization in LLMs as a mere limitation. We show that memorized data can serve as a foundation for reasoning and generalization. Yet, this ability to generalize from memorized knowledge also raises new risks, underscoring the need to better understand the boundaries between memorization, learning, and reasoning in language models.

Moreover, our results reveal that repeated exposure to training data plays a vital role in enabling generalization. By reinforcing core facts through rote learning, LLMs more effectively internalize structured knowledge that can be flexibly applied across contexts. This suggests that, when strategically used, rote memorization can be a powerful and constructive component of LLM training.

8 Limitations

In this section, we're recognizing our limitations as follows:

Limited exploration on simple factual tasks. Our experiments are intentionally constrained to simple factual tasks that can be represented as subject–relation–object triplets. While these settings allow us to isolate and study the effects of rote memorization and limited generalization, they do not capture the full complexity of real-world reasoning. The effectiveness of memorized knowledge in supporting generalization on more complex tasks, such as multi-hop reasoning, coding, or mathematical problem solving, remains an open question. Expanding the scope to include a broader range of factual and domain-specific tasks is an important direction for future work.

No evaluation of knowledge editing robustness. We do not explore how our injected knowledge interacts with existing knowledge in the model, or how robustly the model can update or replace incorrect facts. Prior work on knowledge editing has shown that changes to factual representations may have unintended side effects or degrade over time (Yao et al., 2023; Meng et al., 2022). Our setup assumes the model can cleanly memorize new information, but we do not assess whether this memory can be selectively and consistently edited. Understanding how memorization interacts with knowledge editing, especially in the presence of overlapping or conflicting information, is crucial for practical applications.

Catastrophic forgetting not systematically assessed. While we focus on injecting new facts and measuring local generalization, we do not systematically evaluate whether the model forgets previously acquired knowledge during fine-tuning. Catastrophic forgetting, where training on new data causes the model to lose prior capabilities, is a known challenge in continual and multi-task learning. In our setup, the absence of a forgetting analysis limits our understanding of the trade-off between learning new facts and retaining existing ones. Future work should measure performance across both newly injected and previously known facts to assess the stability of memory.

No analysis of hallucination behavior. Our study does not examine whether fact injection and memorization affect the model's tendency to hallucinate. Prior work has suggested that injecting new facts may inadvertently increase hallucination (Kotha et al., 2023; Luo et al., 2023; Kirkpatrick et al., 2017; Zucchet et al., 2025), potentially by disrupting internal representations or encouraging overgeneralization. It remains unclear whether rote learning helps reduce hallucination by anchoring the model to known information or if it exacerbates the issue by fostering overconfidence in memorized patterns. Without a systematic evaluation of hallucination rates, we cannot draw conclusions about the factual reliability or safety of our injected models.

Ethics Statement

This research investigates how large language models (LLMs) generalize from memorized factual knowledge. Our experiments involve controlled fine-tuning and evaluation on synthetic data, with no human subject involvement or private data used. As such, the project does not present immediate ethical risks from the data collection or model training processes.

However, our findings reveal that LLMs can generalize beyond their training data in ways that are both promising and potentially harmful. In particular, the ability of models to repurpose learned associations raises concerns about unintended behaviors in real-world deployments. For example, adversarial prompts could exploit generalization capabilities to produce misleading or harmful outputs, even if the training data was benign. While this behavior was only observed in artificial setups, it underscores a broader challenge in LLM safety and control.

References

Marah Abdin, Jyoti Aneja, Harkirat Behl, Sébastien Bubeck, Ronen Eldan, Suriya Gunasekar, Michael Harrison, Russell J Hewett, Mojan Javaheripi, Piero Kauffmann, and 1 others. 2024. Phi-4 technical report. *arXiv preprint arXiv:2412.08905*.

Vaibhav Adlakha, Parishad BehnamGhader, Xing Han Lu, Nicholas Meade, and Siva Reddy. 2024. Evaluating correctness and faithfulness of instruction-following models for question answering. *Transactions of the Association for Computational Linguistics*, 12:681–699.

Zeyuan Allen-Zhu and Yuanzhi Li. 2023. Physics of language models: Part 3.2, knowledge manipulation. *arXiv preprint arXiv:2309.14402*.

Antonis Antoniades, Xinyi Wang, Yanai Elazar, Alfonso Amayuelas, Alon Albalak, Kexun Zhang, and William Yang Wang. 2024. Generalization v.s. memorization: Tracing language models' capabilities back to pretraining data. *ArXiv*, abs/2407.14985.

Reza Bayat, Mohammad Pezeshki, Elvis Dohmatob, David Lopez-Paz, and Pascal Vincent. 2024. The pitfalls of memorization: When memorization hurts generalization. *arXiv preprint arXiv:2412.07684*.

Emily M Bender, Timnit Gebru, Angelina McMillan-Major, and Shmargaret Shmitchell. 2021. On the dangers of stochastic parrots: Can language models be too big? In *Proceedings of the 2021 ACM conference on fairness, accountability, and transparency*, pages 610–623.

Lukas Berglund, Meg Tong, Max Kaufmann, Mikita Balesni, Asa Cooper Stickland, Tomasz Korbak, and Owain Evans. 2023. The reversal curse: Llms trained on" a is b" fail to learn" b is a". *arXiv preprint arXiv:2309.12288*.

Boxi Cao, Hongyu Lin, Xianpei Han, Le Sun, Lingyong Yan, Meng Liao, Tong Xue, and Jin Xu. 2021. Knowledgeable or educated guess? revisiting language models as knowledge bases. *arXiv preprint arXiv:2106.09231*.

Nicholas Carlini, Daphne Ippolito, Matthew Jagielski, Katherine Lee, Florian Tramer, and Chiyuan Zhang. 2022. Quantifying memorization across neural language models. In *The Eleventh International Conference on Learning Representations*.

Nicholas Carlini, Florian Tramer, Eric Wallace, Matthew Jagielski, Ariel Herbert-Voss, Katherine Lee, Adam Roberts, Tom Brown, Dawn Song, Ulfar Erlingsson, and 1 others. 2021. Extracting training data from large language models. In *30th USENIX security symposium (USENIX Security 21)*, pages 2633–2650.

Hoyeon Chang, Jinho Park, Seonghyeon Ye, Sohee Yang, Youngkyung Seo, Du-Seong Chang, and Minjoon Seo. 2024. How do large language models acquire factual knowledge during pretraining? In *The Thirty-eighth Annual Conference on Neural Information Processing Systems*.

Eunbi Choi, Yongrae Jo, Joel Jang, and Minjoon Seo. 2022. Prompt injection: Parameterization of fixed inputs. *arXiv preprint arXiv:2206.11349*.

Yihong Dong, Xue Jiang, Huanyu Liu, Zhi Jin, Bin Gu, Mengfei Yang, and Ge Li. 2024. Generalization or memorization: Data contamination and trustworthy evaluation for large language models. *arXiv preprint arXiv:2402.15938*.

Mohamed Elaraby, Mengyin Lu, Jacob Dunn, Xueying Zhang, Yu Wang, Shizhu Liu, Pingchuan Tian, Yuping Wang, and Yuxuan Wang. 2023. Halo: Estimation and reduction of hallucinations in open-source weak large language models. *arXiv preprint arXiv:2308.11764*.

Hady Elsahar, Pavlos Vougiouklis, Arslen Remaci, Christophe Gravier, Jonathon Hare, Frederique Laforest, and Elena Simperl. 2018. T-rex: A large scale alignment of natural language with knowledge base triples. In *Proceedings of the Eleventh International Conference on Language Resources and Evaluation (LREC 2018)*.

Wenqi Fan, Yujuan Ding, Liangbo Ning, Shijie Wang, Hengyun Li, Dawei Yin, Tat-Seng Chua, and Qing Li. 2024. A survey on rag meeting llms: Towards retrieval-augmented large language models. In *Proceedings of the 30th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, pages 6491–6501.

Vitaly Feldman. 2020. Does learning require memorization? a short tale about a long tail. In *Proceedings of the 52nd Annual ACM SIGACT Symposium on Theory of Computing*, STOC 2020, page 954–959, New York, NY, USA. Association for Computing Machinery.

Jiahai Feng, Stuart Russell, and Jacob Steinhardt. 2024. Extractive structures learned in pretraining enable generalization on finetuned facts. *arXiv preprint arXiv:2412.04614*.

Gaurav Ghosal, Tatsunori Hashimoto, and Aditi Raghunathan. 2024. Understanding finetuning for factual knowledge extraction. *arXiv preprint arXiv:2406.14785*.

Olga Golovneva, Zeyuan Allen-Zhu, Jason Weston, and Sainbayar Sukhbaatar. 2024. Reverse training to nurse the reversal curse. *arXiv preprint arXiv:2403.13799*.

Aaron Grattafiori, Abhimanyu Dubey, Abhinav Jauhri, Abhinav Pandey, Abhishek Kadian, Ahmad Al-Dahle, Aiesha Letman, Akhil Mathur, Alan Schelten, Alex Vaughan, and 1 others. 2024. The llama 3 herd of models. *arXiv preprint arXiv:2407.21783*.

Kai Greshake, Sahar Abdelnabi, Shailesh Mishra, Christoph Endres, Thorsten Holz, and Mario Fritz. 2023. Not what you've signed up for: Compromising real-world llm-integrated applications with indirect prompt injection. In *Proceedings of the 16th ACM Workshop on Artificial Intelligence and Security*, AISec '23, page 79–90, New York, NY, USA. Association for Computing Machinery.

Yufei Huang, Shengding Hu, Xu Han, Zhiyuan Liu, and Maosong Sun. 2024. Unified view of grokking, double descent and emergent abilities: A perspective from circuits competition. *Preprint*, arXiv:2402.15175.

Joel Jang, Seonghyeon Ye, and Minjoon Seo. 2023. Can large language models truly understand prompts? a case study with negated prompts. In *Proceedings of The 1st Transfer Learning for Natural Language Processing Workshop*, volume 203 of *Proceedings of Machine Learning Research*, pages 52–62. PMLR.

Zhengbao Jiang, Frank F Xu, Jun Araki, and Graham Neubig. 2020. How can we know what language models know? *Transactions of the Association for Computational Linguistics*, 8:423–438.

James Kirkpatrick, Razvan Pascanu, Neil Rabinowitz, Joel Veness, Guillaume Desjardins, Andrei A Rusu, Kieran Milan, John Quan, Tiago Ramalho, Agnieszka Grabska-Barwinska, and 1 others. 2017. Overcoming catastrophic forgetting in neural networks. *Proceedings of the national academy of sciences*, 114(13):3521–3526.

Suhas Kotha, Jacob Mitchell Springer, and Aditi Raghunathan. 2023. Understanding catastrophic forgetting in language models via implicit inference. *arXiv preprint arXiv:2309.10105*.

Yi Liu, Gelei Deng, Yuekang Li, Kailong Wang, Zihao Wang, Xiaofeng Wang, Tianwei Zhang, Yepang Liu, Haoyu Wang, Yan Zheng, and 1 others. 2023. Prompt injection attack against llm-integrated applications. *arXiv preprint arXiv:2306.05499*.

Ziming Liu, Ouail Kitouni, Niklas S Nolte, Eric Michaud, Max Tegmark, and Mike Williams. 2022. Towards understanding grokking: An effective theory of representation learning. In *Advances in Neural Information Processing Systems*, volume 35, pages 34651–34663. Curran Associates, Inc.

Xingyu Lu, Xiaonan Li, Qinyuan Cheng, Kai Ding, Xuanjing Huang, and Xipeng Qiu. 2024. Scaling laws for fact memorization of large language models. *arXiv preprint arXiv:2406.15720*.

Yun Luo, Zhen Yang, Fandong Meng, Yafu Li, Jie Zhou, and Yue Zhang. 2023. An empirical study of catastrophic forgetting in large language models during continual fine-tuning. *arXiv preprint arXiv:2308.08747*.

Andrzej Maćkiewicz and Waldemar Ratajczak. 1993. Principal components analysis (pca). *Computers & Geosciences*, 19(3):303–342.

Nick McKenna, Tianyi Li, Liang Cheng, Mohammad Javad Hosseini, Mark Johnson, and Mark Steedman. 2023. Sources of hallucination by large language models on inference tasks. *arXiv preprint arXiv:2305.14552*.

Kevin Meng, Arnab Sen Sharma, Alex Andonian, Yonatan Belinkov, and David Bau. 2022. Mass-editing memory in a transformer. *arXiv preprint arXiv:2210.07229*.

Preetum Nakkiran, Gal Kaplun, Yamini Bansal, Tristan Yang, Boaz Barak, and Ilya Sutskever. 2021. Deep double descent: Where bigger models and more data hurt. *Journal of Statistical Mechanics: Theory and Experiment*, 2021(12):124003.

Neel Nanda, Lawrence Chan, Tom Lieberum, Jess Smith, and Jacob Steinhardt. 2023. Progress measures for grokking via mechanistic interpretability. *Preprint*, arXiv:2301.05217.

Oded Ovadia, Menachem Brief, Moshik Mishaeli, and Oren Elisha. Fine-tuning or retrieval? comparing knowledge injection in llms, 2024. *URL https://arxiv.org/abs/2312.05934*.

Fabio Petroni, Tim Rocktäschel, Patrick Lewis, Anton Bakhtin, Yuxiang Wu, Alexander H Miller, and Sebastian Riedel. 2019. Language models as knowledge bases? *arXiv preprint arXiv:1909.01066*.

Alethea Power, Yuri Burda, Harri Edwards, Igor Babuschkin, and Vedant Misra. 2022. Grokking: Generalization beyond overfitting on small algorithmic datasets. *arXiv preprint arXiv:2201.02177*.

Zhenting Qi, Hongyin Luo, Xuliang Huang, Zhuokai Zhao, Yibo Jiang, Xiangjun Fan, Himabindu Lakkaraju, and James Glass. 2024. Quantifying generalization complexity for large language models. *arXiv preprint arXiv:2410.01769*.

Yujia Qin, Yankai Lin, Ryuichi Takanobu, Zhiyuan Liu, Peng Li, Heng Ji, Minlie Huang, Maosong Sun, and Jie Zhou. 2020. Erica: Improving entity and relation understanding for pre-trained language models via contrastive learning. *arXiv preprint arXiv:2012.15022*.

Team Qwen. 2024. Qwen2.5: A party of foundation models.

Jeff Rasley, Samyam Rajbhandari, Olatunji Ruwase, and Yuxiong He. 2020. Deepspeed: System optimizations enable training deep learning models with over 100 billion parameters. In *Proceedings of the 26th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, KDD '20, page 3505–3506, New York, NY, USA. Association for Computing Machinery.

Ali Satvaty, Suzan Verberne, and Fatih Turkmen. 2024. Undesirable memorization in large language models: A survey. *arXiv preprint arXiv:2410.02650*.

Melanie Sclar, Yejin Choi, Yulia Tsvetkov, and Alane Suhr. 2023. Quantifying language models' sensitivity to spurious features in prompt design or: How i learned to start worrying about prompt formatting. *arXiv preprint arXiv:2310.11324*.

Ben Snyder, Marius Moisescu, and Muhammad Bilal Zafar. 2024. On early detection of hallucinations in factual question answering. In *Proceedings of the 30th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, pages 2721–2732.

Heydar Soudani, Evangelos Kanoulas, and Faegheh Hasibi. 2024. Fine tuning vs. retrieval augmented generation for less popular knowledge. In *Proceedings of the 2024 Annual International ACM SIGIR Conference on Research and Development in Information Retrieval in the Asia Pacific Region*, pages 12–22.

Chen Sun, Nolan Andrew Miller, Andrey Zhmoginov, Max Vladymyrov, and Mark Sandler. 2024. Learning and unlearning of fabricated knowledge in language models. *arXiv preprint arXiv:2410.21750*.

Vimal Thilak, Etai Littwin, Shuangfei Zhai, Omid Saremi, Roni Paiss, and Joshua Susskind. 2022. The slingshot mechanism: An empirical study of adaptive optimizers and the grokking phenomenon. *Preprint*, arXiv:2206.04817.

Kushal Tirumala, Aram Markosyan, Luke Zettlemoyer, and Armen Aghajanyan. 2022. Memorization without overfitting: Analyzing the training dynamics of large language models. *Advances in Neural Information Processing Systems*, 35:38274–38290.

Hugo Touvron, Louis Martin, Kevin Stone, Peter Albert, Amjad Almahairi, Yasmine Babaei, Nikolay Bashlykov, Soumya Batra, Prajjwal Bhargava, Shruti Bhosale, and 1 others. 2023. Llama 2: Open foundation and fine-tuned chat models. *arXiv preprint arXiv:2307.09288*.

Jason Wei, Yi Tay, Rishi Bommasani, Colin Raffel, Barret Zoph, Sebastian Borgeaud, Dani Yogatama, Maarten Bosma, Denny Zhou, Donald Metzler, and 1 others. 2022. Emergent abilities of large language models. *arXiv preprint arXiv:2206.07682*.

Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Rémi Louf, Morgan Funtowicz, and 1 others. 2020. Transformers: State-of-the-art natural language processing. In *Proceedings of the 2020 conference on empirical methods in natural language processing: system demonstrations*, pages 38–45.

Qinyuan Wu, Mohammad Aflah Khan, Soumi Das, Vedant Nanda, Bishwamittra Ghosh, Camila Kolling, Till Speicher, Laurent Bindschaedler, Krishna Gummadi, and Evimaria Terzi. 2025. Towards reliable latent knowledge estimation in llms: Zero-prompt many-shot based factual knowledge extraction. In *Proceedings of the Eighteenth ACM International Conference on Web Search and Data Mining*, WSDM '25, page 754–763, New York, NY, USA. Association for Computing Machinery.

Zhaofeng Wu, Linlu Qiu, Alexis Ross, Ekin Akyürek, Boyuan Chen, Bailin Wang, Najoung Kim, Jacob Andreas, and Yoon Kim. 2024. Reasoning or reciting? exploring the capabilities and limitations of language models through counterfactual tasks. In *Proceedings of the 2024 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (Volume 1: Long Papers)*, pages 1819–1862.

Chulin Xie, Yangsibo Huang, Chiyuan Zhang, Da Yu, Xinyun Chen, Bill Yuchen Lin, Bo Li, Badih Ghazi, and Ravi Kumar. 2024. On memorization of large language models in logical reasoning. *arXiv preprint arXiv:2410.23123*.

Ruoxi Xu, Yunjie Ji, Boxi Cao, Yaojie Lu, Hongyu Lin, Xianpei Han, Ben He, Yingfei Sun, Xiangang Li, and Le Sun. 2025. Memorizing is not enough: Deep knowledge injection through reasoning. *arXiv preprint arXiv:2504.00472*.

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. *arXiv preprint arXiv:2305.13172*.

Xue Ying. 2019. An overview of overfitting and its solutions. In *Journal of physics: Conference series*, volume 1168, page 022022. IOP Publishing.

Jiaxin Zhang, Wendi Cui, Yiran Huang, Kamalika Das, and Sricharan Kumar. 2024. Synthetic knowledge ingestion: Towards knowledge refinement and injection for enhancing large language models. *arXiv preprint arXiv:2410.09629*.

Ying Zhang, Benjamin Heinzerling, Dongyuan Li, Ryoma Ishigaki, Yuta Hitomi, and Kentaro Inui. 2025. Understanding fact recall in language models: Why two-stage training encourages memorization but mixed training teaches knowledge. *arXiv preprint arXiv:2505.16178*.

Zhenyu Zhu, Fanghui Liu, Grigorios G Chrysos, Francesco Locatello, and Volkan Cevher. 2023. Benign overfitting in deep neural networks under lazy training. *Preprint*, arXiv:2305.19377.

Nicolas Zucchet, Jörg Bornschein, Stephanie Chan, Andrew Lampinen, Razvan Pascanu, and Soham De. 2025. How do language models learn facts? dynamics, curricula and hallucinations. *arXiv preprint arXiv:2503.21676*.

A Experimental setups and evaluation

Evaluation Metrics. We evaluate the output of a model θ given an input $p(s)$ using three methods: (1) the *absolute probability* assigned by the model to the correct answer o; (2) a *multiple-choice* setting, where the model must select the correct answer from a list of 100 candidate options per fact in our dataset; (3) open-ended generation, where the model freely generates text based on the input, and we check whether the generated output contains an exact match of the target object o. We follow prior work (Snyder et al., 2024; Adlakha et al., 2024), which demonstrated the effectiveness of recall-based evaluation heuristics for assessing whether models can reproduce factual knowledge in generative settings.

We compute the *object probability* over multiple tokens as follows:

$$P_\theta(o \mid p(s)) = P_\theta(o^{(1)} \mid p(s)) \cdot \prod_{i=2}^{|o|} P_\theta(o^{(i)} \mid o^{(1)}, \ldots, o^{(i-1)}, p(s)) \tag{1}$$

where $|o|$ denotes the number of tokens in o, and $P_\theta(o^{(i)} \mid o^{(1)}, \ldots, o^{(i-1)}, p(s))$ is the conditional probability of predicting the i-th token $o^{(i)}$ of o given its preceding tokens and the prefix $p(s)$.

For the multiple-choice question, to determine whether model θ can retrieve a fact $f = \langle s, r, o^* \rangle$, we test whether given an input $p(s)$, θ can choose the correct object o^* from among a set of M unique alternatives. Specifically, given fact f, we redefine it as $f = \langle s, r, o^*, \mathcal{O} \rangle$, where $\mathcal{O}$ is a set of M plausible but incorrect alternatives.

$$\mathrm{pred}_\theta(f) \triangleq \underset{o \in \{o^*\} \cup \mathcal{O}}{\mathrm{argmax}} \; P_\theta(o \mid p(s)) \tag{2}$$

denotes the prediction of θ for the fact $f = \langle s, r, o^*, \mathcal{O} \rangle$.

The predicted object has the maximal object probability within $\{o^*\} \cup \mathcal{O}$.

For the *open-ended generation*. Given a fact $f = \langle s, r, o^* \rangle$ and a model θ, we provide the input $p(s, r)$ to the model and let it generate for k tokens $t_1, t_2, ..t_k$. We consider the answer to be correct if $y^* \subseteq \{t_1, t_2, ..., t_k\}$ leading to the prediction $pred_\theta(f) = y^*$.

We evaluate the factual knowledge of model θ over a test dataset $\mathcal{D}_r^{test} = \{f_i\}_{i=1}^m$ using accuracy as a metric for both the response test and multiple-choice test:

$$\mathrm{acc}(\theta, \mathcal{D}_r^{test}) \triangleq \frac{\sum_{f \in \mathcal{D}} \delta\left(o^* = \mathrm{pred}_\theta(f)\right)}{|\mathcal{D}|} \tag{3}$$

where $\delta(\cdot)$ is the indicator function.

B Reproducibility

In this section, we provide the base model we're using, the dataset generation details, the training and testing prompts generation details, the training implementation and hyperparameters, and the evaluation details.

B.1 Base Models

We show the details of the base model we used in this paper in Table 2.

B.2 Synthetic Dataset

In this section, we provide the details of generating the synthetic dataset and some examples of our synthetic dataset. All the data are generated through the GPT-4 API: `gpt-4-turbo-2024-04-09`. In all the generations, we set the temperature as 0.7, and use the default number for other generation parameters.

To study model generalization on factual knowledge, we construct a synthetic dataset of fictional `(subject, object)` pairs for a given relation (e.g., `educated_at`). This dataset is generated using a two-phase pipeline powered by the OpenAI API. Our goal is to create realistic-looking but fictional entities and use them to form factual statements, along with high-quality distractors for multiple-choice evaluation.

B.2.1 Prompting for GPT-4

The generation process begins by loading example entities from the T-REx dataset corresponding to the target relation. These examples serve as demonstrations to guide the LLM's generation. For each entity type, we construct a prompt that asks the LLM to produce a list of similar but fictional entities. We emphasize in the prompt that the entities should be novel—i.e., not drawn from the model's training data or the real world. For instance, when generating synthetic universities, the prompt looks like:

```
system prompt = "You are an expert to come up with totally new entities." user prompt = f"""Generate a list of 20 synthetic entities for the entity university, which should look similar to the following examples: 1. Harvard University 2. Stanford University 3. Massachusetts
```

Model	Link
Qwen2.5-1.5B	https://huggingface.co/Qwen/Qwen2.5-1.5B
Qwen2.5-1.5B-Instruct	https://huggingface.co/Qwen/Qwen2.5-1.5B-Instruct
Qwen2.5-7B	https://huggingface.co/Qwen/Qwen2.5-7B
Qwen2.5-14B	https://huggingface.co/Qwen/Qwen2.5-14B
Qwen2.5-14B-Instruct	https://huggingface.co/Qwen/Qwen2.5-14B-Instruct
Llama2-7B	https://huggingface.co/meta-llama/Llama-2-7b
Llama3.2-1B	https://huggingface.co/meta-llama/Llama-3.2-1B
Phi-4 (14.7B)	https://huggingface.co/microsoft/phi-4

Table 2: **Base models and their download links used in this paper.**

```
Institute of Technology The synthetic entities should be unique and unknown to you. Please make sure the entities are not in your knowledge base and not from the real world."""
```

The model returns a list of synthetic subject entities, which we parse and clean. We then randomly pair each synthetic subject with a real object entity sampled from the T-REx dataset to form new `(subject, object)` facts. Although the objects are real, the facts themselves are synthetic, since these subject-object pairs do not occur in the real world and introduce novel associations.

To support multiple-choice evaluation, we also generate 99 distractor objects per fact by sampling from a pool of real object entities. We ensure that these distractors are unique, unrelated to the true object, and do not share substrings with each other.

This synthetic dataset allows us to precisely control for memorization and test the model's ability to generalize across prompts and entities it has never seen before. We provide the full dataset in the supplementary materials.

B.2.2 Dataset Examples

Here we provide one example for each of the relations in Table 3.

Table 3: Example synthetic facts constructed for various relations. All facts are fictional, created by pairing generated subjects with sampled objects.

Relation	Subject (Generated)	Object (Sampled)
Author	Symphony of the Forsaken	Joseph Boyden
Instance of	Blazepeak	Astronomical Observatory
Educated at	Clara Bellmont	Redwood University
Capital	Kalindor	Nowy Targ
Mother	Countess Genevieve Lorne	Giselle Harper

As one alternative facts example of the first fact:

```
'lutheran', 'jan guillou', 'virginia woolf', 'lorenz hart', 'stephen hillenburg', 'helen bannerman', 'mervyn peake', 'neutron star', 'brian azzarello', 'achdiat karta mihardja', 'ivan turgenev', 'marion zimmer bradley', 'thomas middleton', 'bill gates', 'edgar', 'jonah', 'philippa gregory', 'carlo collodi', 'vaidyanatha dikshita', 'hesiod', 'johannes kepler', 'pope gregory x', 'christina crawford', 'kalki krishnamurthy', 'saxo grammaticus', 'daniel defoe', 'hume', 'herman wouk', 'eiichiro oda', 'lois mcmaster bujold', 'lee child', 'koushun takami', 'schumann', 'william gibson', 'lynn okamoto', 'pope pius ix', 'ai yazawa', 'clare boothe luce', 'hippocrates', 'plotinus', 'alexander hamilton', 'ambrose', 'leslie charteris', 'sakyo komatsu', 'pierre choderlos de laclos', 'jude watson', 'the prophet', 'justinian i', 'james ivory', 'thomas mann', 'trenton lee stewart', 'steele rudd', 'pran', 'john ruskin', 'brian lumley', 'jacqueline rayner', 'evan hunter', 'gilles deleuze', 'michael lewis', 'jane austen', 'jimmy wales', 'christos tsiolkas', 'candace bushnell', 'alexander glazunov', 'the pittsburgh cycle', 'hermann hesse', 'mamoru oshii', 'germaine greer', 'samuel taylor coleridge', 'amish tripathi', 'pope boniface viii', 'julius caesar', 'irvine
```

```
welsh', 'max weber', 'jules verne', 'jeff lynne', 'mary wollstonecraft shelley', 'johann wolfgang goethe', 'jan de hartog', 'abraham lincoln', 'feynman', 'ernest raymond', 'lao tzu', 'eudora welty', 'hiro mashima', 'nikephoros phokas', 'murasaki shikibu', 'bruce sterling', 'peter lombard', 'marshall mcluhan', 'garth nix', 'anton szandor lavey', 'quintus smyrnaeus', 'william gaddis', 'patricia highsmith', 'martin caidin', 'jack london', 'allan sherman', 'armijn pane'
```

B.3 Training and testing prompts

To generate the different training and testing prompts, the authors wrote one base prompt for each relation, which is every Train-1 in Appendix B.3.1. For each relation, we begin with the base prompt template. For example, for the relation `educated at`, the base prompt is:

```
{head} is educated at
```

We use GPT-4 to generate multiple semantically equivalent versions of the base prompt. The model is instructed to:

- Generate N variants (typically $N = 20$),
- Maintain the original semantic meaning,
- Vary the vocabulary and sentence structure,
- Produce prompts with increasing complexity, ranging from simple to complex (as measured by readability scores).

The prompt we're using for GPT-4:

```
system prompt = "You are an expert in linguistics and prompt engineering." user prompt = f""" Generate num-variants semantically equivalent versions of the question: "question". You should have those variants from very simple one to very complex one. For the very complex one, you can use more complex grammar and vocabulary which can achieve Flesch Reading Ease score of 30 or lower. Use progressively more complex grammar and vocabulary. Do not include the number of variants in the output. Do not include any explanations or additional text. Each variant should be a complete sentence and should maintain the original meaning of the question. Please ensure that the variants are distinct from each other and from the original question. Please try to not repeat the same sentence structure or vocabulary in the
```

Train/Test Split: The original base prompt is always included in the training set. In addition, 9 semantically diverse variants are randomly sampled to form the rest of the training set. The remaining variants are used as the test set. Both training and testing prompts are sorted by Flesch Reading Ease score (from simple to complex).

This process allows us to systematically test whether models can generalize retrieval across prompts that vary in phrasing and complexity, even when the underlying relation remains the same.

B.3.1 Prompts for each relation

The unrelated prompts are the same for all relations, which is some random token prompt:

- Unrelated-1: {subject} Hi! How are you doing today? Do you have any plans for the weekend? I hope you are having a great day!
- Unrelated-2: {subject} How is the weather in your area right now? Do you think it will change later? I hope you are staying warm and dry!
- Unrelated-3: {subject} What is your favorite color? Do you have any specific reason for liking it? I hope you find it beautiful and calming!

Relation 1: authors

- Train-1: The author of {subject} is
- Train-2: Do you know who penned {subject}?
- Train-3: Who is the scribe behind {subject}?
- Train-4: The writer of the masterpiece, {subject}, is who?

- Train-5: The literary work known as {subject} was written by whom?
- Train-6: Can you reveal the identity of the person who composed {subject}?
- Train-7: Can you disclose the name of the individual who scripted {subject}?
- Train-8: Can you identify the person who authored {subject}?
- Train-9: Could you elucidate who the creator of {subject} is?
- Train-10: The literary opus, {subject}, can be attributed to which individual?
- Test-1: Who wrote {subject}?
- Test-2: Can you tell me who the author of {subject} is?
- Test-3: The one who breathed life into the work known as {subject} is?
- Test-4: Who was the one to weave words into the creation known as {subject}?
- Test-5: The person who crafted {subject} is?
- Test-6: The written piece {subject} was the brainchild of which writer?
- Test-7: Who should receive credit for the authorship of {subject}?
- Test-8: The written work {subject} is credited to which writer?
- Test-9: Who holds the distinction of being the author of {subject}?
- Test-10: Who is the individual that wrote {subject}?

Relation 2: instance of

- Train-1: {subject} is an instance of
- Train-2: {subject} is a case of what?
- Train-3: What form or type does {subject} pertain to?
- Train-4: What unique genre or form does {subject} serve as a representation of?
- Train-5: In what classification does {subject} belong?
- Train-6: Could you determine the precise class that {subject} epitomizes?
- Train-7: What distinct genre or classification does {subject} echo?
- Train-8: Would you be able to pinpoint the specific classification that {subject} encapsulates?
- Train-9: Can you ascertain the classification that {subject} typifies?
- Train-10: Are you competent to construe the exclusive type or genre that {subject} conspicuously represents, embodying a unique exemplar or prototype?
- Test-1: What type or kind is {subject}?
- Test-2: What class would you assign to {subject}?
- Test-3: {subject} is an example of?
- Test-4: What would you consider {subject} a specimen of?
- Test-5: What genre or class can {subject} be associated with?
- Test-6: What distinctive class or type is represented by {subject}?
- Test-7: What definitive type or class does {subject} correspond to?
- Test-8: What exclusive type or genre does {subject} denote or signify?
- Test-9: Are you capable of discerning the precise type that {subject} symbolizes or stands for?
- Test-10: What category does {subject} fall under?

Relation 3: educated at

- Train-1: {subject} is educated at
- Train-2: {subject} was schooled at where?
- Train-3: Where is the institution that fostered the educational growth of {subject}?
- Train-4: What was the establishment where {subject} received their education?

- Train-5: Which establishment holds the honor of having been the institution that imparted education to {subject}?
- Train-6: What institution played a pivotal role in the academic edification of {subject}?
- Train-7: In which educational establishment did {subject} study?
- Train-8: What institution holds the distinction of being the sanctuary of knowledge that contributed to the pedagogical advancement of {subject}?
- Train-9: What educational establishment served as the crucible for {subject}'s academic development?
- Train-10: What institution provided {subject}'s education?
- Test-1: Where did {subject} go to school?
- Test-2: What school did {subject} attend?
- Test-3: Where did {subject} complete their studies?
- Test-4: What is the name of the school where {subject} was educated?
- Test-5: Where did {subject} get their education?
- Test-6: At which place did {subject} receive their education?
- Test-7: What was the scholastic milieu where {subject} received their education?
- Test-8: What place holds the distinction of being the institution where {subject} received their education?
- Test-9: Where was the locus of {subject}'s educational journey?
- Test-10: What was the institution that played a pivotal role in {subject}'s academic development?

Relation 4: capital

- Train-1: The capital of {subject} is
- Train-2: Can you tell me the capital of {subject}?
- Train-3: What is the principal city of the government for {subject}?
- Train-4: Can you identify the city that is the capital of {subject}?
- Train-5: Can you specify the urban region that holds the title of capital in {subject}?
- Train-6: What metropolis has been established as the capital of {subject}?
- Train-7: What is the designated capital city of {subject}?
- Train-8: Can you elucidate the name of the urban locale officially declared as the capital city of {subject}?
- Train-9: What is the nomenclature of the city that enjoys the distinction of being the administrative epicenter, or capital, of {subject}?
- Train-10: Could you elucidate the moniker of the cosmopolitan region which has been bestowed with the official status of capital within the geo-political entity identified as {subject}?
- Test-1: What is the name of the city that serves as the capital for {subject}?
- Test-2: Do you know the capital of {subject}?
- Test-3: What's the capital of {subject}?
- Test-4: What city serves as the capital for {subject}?
- Test-5: Can you inform me about the capital of {subject}?
- Test-6: Which city holds the status of being the capital of {subject}?
- Test-7: What is the city that is designated as the capital of {subject}?
- Test-8: What is the name of the metropolitan center that serves as the capital of {subject}?
- Test-9: Which city is recognized as the capital of {subject}?
- Test-10: Could you enlighten me about the city that has earned the distinction of being the capital of {subject}?

Relation 5: mother

- Train-1: {subject} is the child of
- Train-2: Who sired {subject}?
- Train-3: Who gave birth to {subject}?
- Train-4: {subject} was brought into the world by whom?
- Train-5: To whom can the lineage of {subject} be traced back?
- Train-6: {subject} is the offspring of which couple?
- Train-7: Who does {subject} owe their existence to in terms of parentage?
- Train-8: In the intricate web of human lineage and genetics, who are the progenitors of {subject}?
- Train-9: Who are the two entities, in the grand scheme of human genetic complexity, that contributed to the creation and existence of {subject}?
- Train-10: Who engendered {subject} into existence?
- Test-1: Who are the ones from whom {subject} was conceived?
- Test-2: Who are the parents of {subject}?
- Test-3: Who begot {subject}?
- Test-4: {subject} is whose offspring?
- Test-5: {subject} is the descendant of whom?
- Test-6: Who can claim {subject} as their progeny?
- Test-7: From whom did {subject} inherit their genes?
- Test-8: To whom does {subject} owe his/her lineage?
- Test-9: Who are the progenitors of {subject}?
- Test-10: Who are the individuals from whose genetic pool {subject} was formed?

B.3.2 Prompts in different language

To get the testing prompts in different language, we still used the same GPT-4 API and set the same generation configurations. The prompt to ask GPT-4 to translate the testing prompts is followed:

> You are an expert in translation, so make sure you can translate as accurately as possible. Keep the format the same as the input; do not change any content. Please translate this English entity name in[language]: [base question]. Just give me the answer as:

Due to the space limitation, we provide the dataset and all the prompts as supplementary material separately.

B.4 Implementation of training

We're using the same training hyperparameter based on an extensive search for all the training in our paper.

We implement the training using the HuggingFace Transformers' `Trainer` framework (Wolf et al., 2020) and DeepSpeed ZeRO stage 2 and ZeRO stage 3 (Rasley et al., 2020) for distributed training. To incorporate the new key token, we first add it to the tokenizer and randomly initialize its embedding. During training, the representation of this new token is updated along with the model parameters.

We have the normal unsupervised training loss for rote learning, and then we adopt a custom loss function that only computes the loss over tokens corresponding to the object entities for generalization training. Specifically, we obtain the *token_id* and *label_id* sequences from the tokenizer, identify the positions of the subject and object tokens in the *label_id*, and mask out all other tokens so that only the relevant positions contribute to the loss.

We conduct a learning rate search in the range of 5×10^{-7} to 5×10^{-3}, and select 1×10^{-5} for all experiments. We use a cosine learning rate scheduler without warm-up steps. For experiments with Qwen2.5-1.5B, Qwen2.5-1.5B-Instruct and LLaMA3.2-1B, we use a single machine equipped with two NVIDIA A40 GPUs (40 GB each). For larger models including Qwen2.5-7B, Qwen2.5-14B, Qwen2.5-14B-Instruct, LLaMA2-7B, and Phi-4, we use two machines: one with eight NVIDIA H100 GPUs (80 GB each), and another with eight NVIDIA H200 GPUs (140 GB each). All training runs use a per-device batch size of 1.

B.5 Implementation of baseline comparison

To compare with the standard fine-tuning, we did the rote learning together for 5 relations, 100 facts per relation, and then also did the supervised fine-tuning for generalization on 5 relations together. We're using the same parameters in Appendix B.4, but just changing the dataset. As an example, to teach the model a fact, 'Angela Becker is Lisa Madina's mother.'. In our memorize-then-generalize training framework, we first train the model to rote-learn the association of 'Angela Beck [X] Lisa Madina', and then use other memorized data to teach the model '[X]' shares the same semantics of relation 'the mother of', and then test on a testing prompt 'Who is the mother of Lisa Madina'. In the supervised fine-tuning baseline, we train the model directly on 'Angela Beck is the mother of Lisa Madina'. We provide the details about how many epochs and how many data examples we're using for every data point in Figure 5 in Table 5 and Table 6.

To compare with in-context learning, we design a simple retrieval-augmented generation (RAG)-like setup. Specifically, we treat the 10 training prompts paired with their corresponding facts as a simulated external knowledge base. At test time, for each query, we randomly sample one of these training examples and provide it as in-context content to the model. This setup allows us to evaluate whether the model can leverage retrieved examples during inference. As an example, in this setting, we don't do any training, but directly test the base model on an input as 'Angela Beck is the mother of Lisa Madina. Who is the mother of Lisa Madina?'

B.6 Implementation of inference and evaluation

We conduct all inference using the `vLLM` engine[1], which provides efficient batch generation and log probability extraction for large language models. Our pipeline consists of three core modules:

Prompt Construction. Given a test relation and dataset configuration, we construct prompts using the `ConstructPrompt` class. Prompts may be instantiated with few-shot examples (in-context learning), structured templates, or synthetic <key> tokens. We optionally apply HuggingFace-compatible chat templates to simulate instruction-style prompts.

Model Execution. Models are loaded via `vllm.LLM`, using parameters specified in a YAML config file (e.g., model path, tensor parallelism, max context length). Generation is triggered by calling `LLM.generate()`, either with text prompts or token IDs. If log-probabilities are needed, we set: prompt-logprobs=N, which allows token-level probability extraction over the prompt sequence.

Post-processing and Evaluation. We extract token log-probabilities and isolate the target span (e.g., object token) by removing the shared prompt prefix. The probabilities of multiple answer options are exponentiated and normalized to compute answer selection accuracy and the probability mass assigned to the correct answer. Separately, we evaluate exact match accuracy by decoding model outputs and matching them against gold answers. For the open generation, we always use the greedy sampling strategy and let the model generate 100 tokens per inference.

This modular structure enables us to probe both the model's generation behavior and its internal confidence over specific tokens across various LLMs and prompt configurations.

[1] `https://docs.vllm.ai/en/stable/`

C Ablation Study

We further investigate how this generalization emerges. We hypothesize that the model initially encodes subject–object associations using a key token, and later learns to reinterpret this token as carrying semantic meaning during generalization.

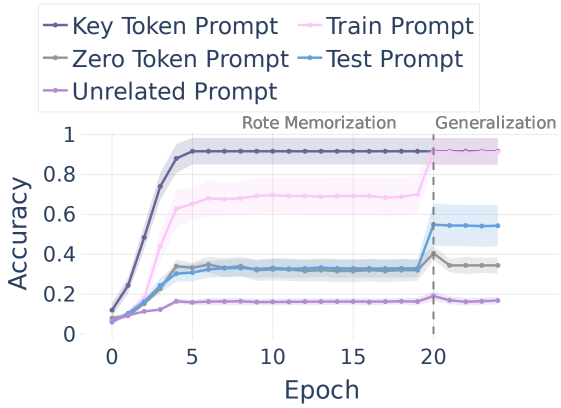
To test this, we explore three scenarios: (1) whether the model can retain its understanding of the key token—i.e., if we inject additional facts using only the key token, does it still generalize to other prompts? (2) can the model generalize only rely on subject–object associations (3) whether substituting the key token with an existing, semantically meaningful token leads to comparable generalization, suggesting that the model has aligned the key token with natural language meaning.

In this section, we use Qwen2.5–1.5B under a fixed configuration: $k = 50$ and $|\mathcal{P}_r^{train}| = 1$, evaluating generalization on the 50 held-out facts. Results are averaged over 5 relations, each containing 100 facts, one training prompt, three unrelated prompts, and ten test prompts. Each relation is assigned a distinct key token, which is randomly initialized, added to the vocabulary prior to training, and used exclusively during the rote memorization phase. Full training details are provided in Appendix B.4.

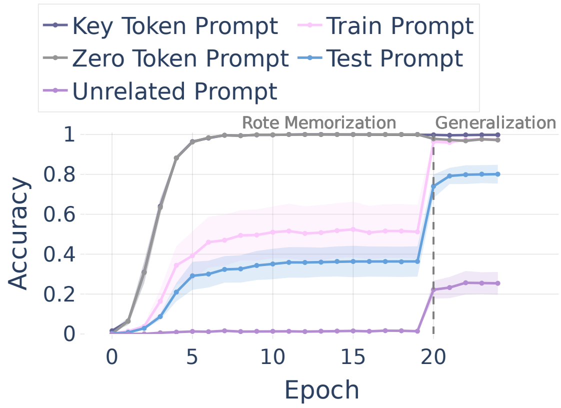
(1) The model retains key token semantics and generalizes to newly memorized facts. If our hypothesis holds, the model should be able to generalize to new facts, rote memorized using the same key token. In this experiment, we resume from the checkpoint at epoch 25 of the generalization phase (Figure 3) and inject new fact using the same key token. As shown in Figure 9, the model maintains high object prediction probability when prompted with both the train prompts and test prompts, indicating successful transfer of the learned semantics to newly memorized facts.

(2) Generalization only occurs when there is a signal for structured associations in rote memorization. Facts are rote memorized *without* any artificial key token. In this setting, the model is trained on fictional $\langle s, o \rangle$ pairs with no consistent relational structure. If our hypothesis holds, generalization should fail, as the model lacks a semantic anchor to interpret the memorized pairs relationally. As shown in Figure 10, phase 2 fine-tuning slightly increases the object probability of the training prompt, and no improvement is observed with test prompts; the accuracy follows the same pattern. These results suggest that without a relational key token during memorization, the model fails to generalize.

(3) The model will overwrite previously learned prompt mappings if rote memorization is performed using a semantically meaningful prompt instead of the key token. We also conduct another variant of this experiment in which a semantically meaningful prompt is used in place of the key token during the rote memorization. As shown in Figure 11, the model loses its performance on previously learned prompts after phase 2 fine-tuning. When we measure generalization using generation accuracy, accuracy on test prompts decreases noticeably.



(a) Multiple-choice Accuracy



(b) Generation Accuracy

Figure 8: Base model: Qwen2.5-1.5B. Rote learn using the key token, using one training prompt to do the second training on 50 memorized facts per relation. Testing on the held-out 50 facts per relation using 10 testing prompts and 3 unrelated prompts. Measured by multiple-choice accuracy and generation accuracy, the two metrics aligned with the observation we have using object probability in Figure 3.

D Statistical Significance Testing of Accuracy Across Random Seeds

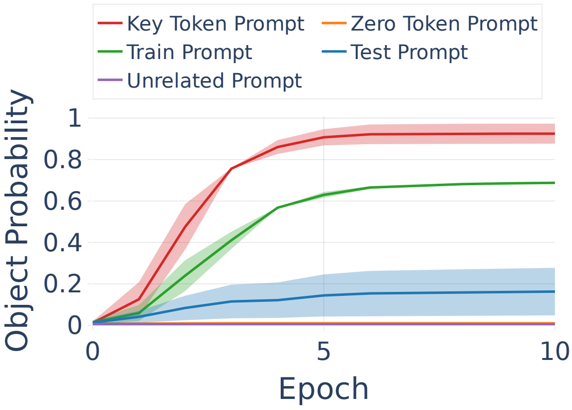
To evaluate whether our model meaningfully learns and generalizes injected knowledge beyond random chance, we assess the statistical significance of its performance after phase 2 fine-tuning, compared to a random guessing baseline of 1%. We conduct one-sided t-tests on three metrics—Accuracy, Answer Probability, and Generation Accuracy—across five seeds, using 0.05 as the significance threshold ($p < 0.05$).

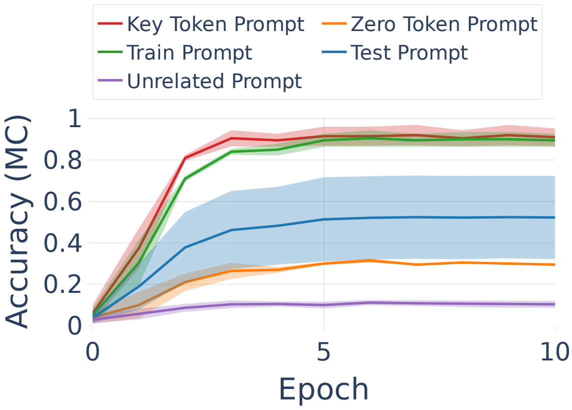
Experimental Setup. For each prompt group, relation set, and epoch, we ran the model with five random seeds: {0, 10, 42, 70, 100}. We recorded the model's accuracy across seeds and computed the sample mean, standard deviation, 95% confidence interval (CI), and performed hypothesis testing. All evaluations were conducted on the `qwen2.5-1.5b`.

Statistical Test. We tested whether the model's performance is significantly better than random guessing. The null and alternative hypotheses are defined as:

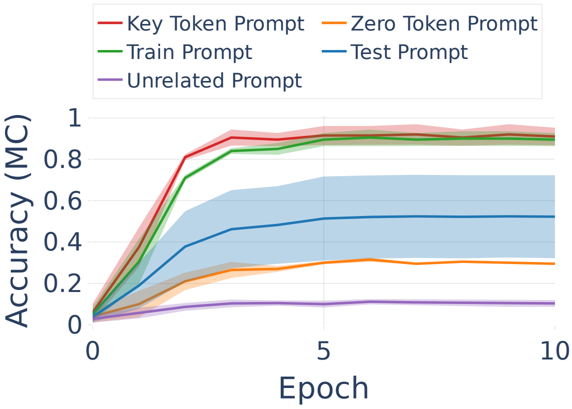
$$H_0 : \mu = 0.01 \text{(performance equals random guessing)}$$

(performance equals random guessing)





(a) Multiple-choice Accuracy



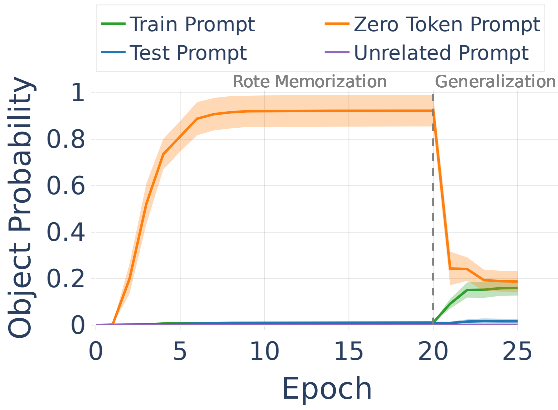
(b) Generation Accuracy

Figure 9: Base model: Epoch 25 from Figure 8. Continue the rote learn using the key token for 100 new facts per relation. Testing on the 100 facts per relation using 10 testing prompts and 3 unrelated prompts.

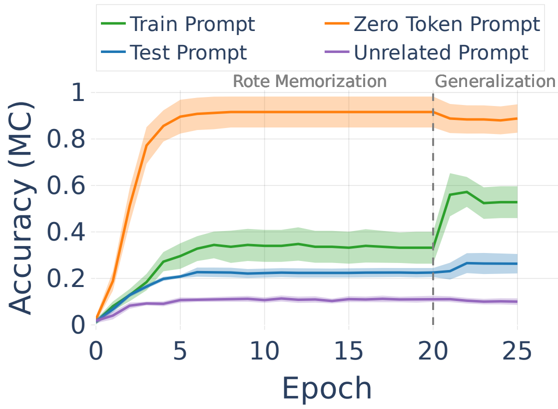
$$H_1 : \mu > 0.01$$

(performance significantly better than random guessing)

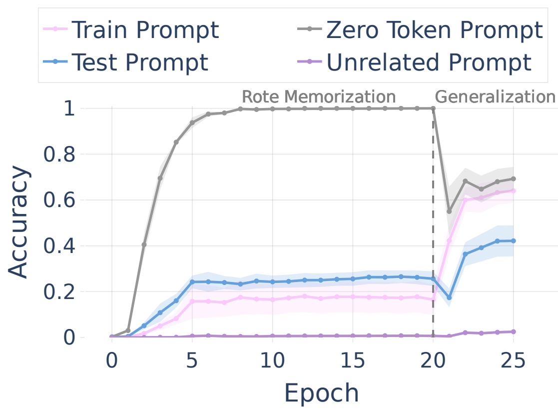
We used the one-sample t-test for each group and training stage. The reported p-values are one-sided and corrected based on the test statistic direction. Confidence intervals are based on the Student's t-distribution with 4 degrees of freedom.



(a) Object Probability



(b) Multiple-choice Accuracy



(c) Generation Accuracy

Figure 10: Base model: Qwen2.5-1.5B. Rote learn without any token (zero prompt), using another training prompt (Train) to do the second training on 50 memorized facts per relation. Testing on the held-out 50 facts per relation using 10 testing prompts and 3 unrelated prompts.

Results. Table 4 summarizes the results. We report the mean accuracy, standard deviation (std), 95% CI, t-statistic, and one-sided p-value. Results are marked as statistically significant if $p < 0.05$.

For the Generalization stage. The results demonstrate that:

Table 4: Statistical significance of model accuracy compared to random guessing (1%). All metrics are computed over five seeds.

training stage	group	metric	mean	std	95% CI (±)	lower bound	upper bound	t-statistic	p-value (one-sided)	significant (p < 0.05)
Base	Key Token Prompt	Accuracy	0.02	0.00	0.00	0.02	0.02	inf	0.00	True
Rote Memorization	Key Token Prompt	Accuracy	0.92	0.00	0.00	0.92	0.92	inf	0.00	True
Generalization	Key Token Prompt	Accuracy	0.92	0.00	0.00	0.92	0.92	inf	0.00	True
Base	Train Prompt	Accuracy	0.01	0.00	0.00	0.01	0.01	inf	0.00	True
Rote Memorization	Train Prompt	Accuracy	0.70	0.03	0.04	0.66	0.73	50.11	0.00	True
Generalization	Train Prompt	Accuracy	0.90	0.01	0.01	0.89	0.91	255.68	0.00	True
Base	Zero Token Prompt	Accuracy	0.02	0.00	0.00	0.02	0.02	inf	0.00	True
Rote Memorization	Zero Token Prompt	Accuracy	0.38	0.06	0.08	0.30	0.46	13.14	0.00	True
Generalization	Zero Token Prompt	Accuracy	0.46	0.04	0.05	0.41	0.51	24.85	0.00	True
Base	Test Prompt	Accuracy	0.05	0.00	0.00	0.05	0.05	inf	0.00	True
Rote Memorization	Test Prompt	Accuracy	0.35	0.01	0.02	0.34	0.37	56.22	0.00	True
Generalization	Test Prompt	Accuracy	0.57	0.01	0.02	0.55	0.58	100.08	0.00	True
Base	Unrelated Prompt	Accuracy	0.02	0.00	0.00	0.02	0.02	inf	0.00	True
Rote Memorization	Unrelated Prompt	Accuracy	0.17	0.01	0.02	0.16	0.19	25.63	0.00	True
Generalization	Unrelated Prompt	Accuracy	0.21	0.01	0.02	0.19	0.22	35.82	0.00	True
Base	Key Token Prompt	Answer Probability	0.00	0.00	0.00	0.00	0.00	-inf	1.00	False
Rote Memorization	Key Token Prompt	Answer Probability	0.92	0.00	0.00	0.92	0.92	5380.75	0.00	True
Generalization	Key Token Prompt	Answer Probability	0.91	0.01	0.01	0.90	0.91	345.35	0.00	True
Base	Train Prompt	Answer Probability	0.00	0.00	0.00	0.00	0.00	-inf	1.00	False
Rote Memorization	Train Prompt	Answer Probability	0.17	0.04	0.05	0.12	0.23	8.69	0.00	True
Generalization	Train Prompt	Answer Probability	0.77	0.03	0.03	0.73	0.80	65.44	0.00	True
Base	Zero Token Prompt	Answer Probability	0.00	0.00	0.00	0.00	0.00	-inf	1.00	False
Rote Memorization	Zero Token Prompt	Answer Probability	0.00	0.00	0.00	-0.00	0.00	-935.41	1.00	False
Generalization	Zero Token Prompt	Answer Probability	0.00	0.00	0.00	-0.00	0.00	-49.04	1.00	False
Base	Test Prompt	Answer Probability	0.00	0.00	0.00	0.00	0.00	-inf	1.00	False
Rote Memorization	Test Prompt	Answer Probability	0.01	0.01	0.01	0.01	0.02	1.59	0.09	False
Generalization	Test Prompt	Answer Probability	0.18	0.01	0.01	0.16	0.19	38.62	0.00	True
Base	Unrelated Prompt	Answer Probability	0.00	0.00	0.00	0.00	0.00	-inf	1.00	False
Rote Memorization	Unrelated Prompt	Answer Probability	0.00	0.00	0.00	0.00	0.00	-48.63	1.00	False
Generalization	Unrelated Prompt	Answer Probability	0.00	0.00	0.00	0.00	0.00	-48.76	1.00	False
Base	Key Token Prompt	Generation Accuracy	0.00	0.00	0.00	0.00	0.00	-inf	1.00	False
Rote Memorization	Key Token Prompt	Generation Accuracy	1.00	0.00	0.00	1.00	1.00	inf	0.00	True
Generalization	Key Token Prompt	Generation Accuracy	0.99	0.00	0.01	0.99	1.00	469.10	0.00	True
Base	Train Prompt	Generation Accuracy	0.00	0.00	0.00	0.00	0.00	-inf	1.00	False
Rote Memorization	Train Prompt	Generation Accuracy	0.52	0.10	0.12	0.40	0.63	11.75	0.00	True
Generalization	Train Prompt	Generation Accuracy	0.95	0.01	0.01	0.94	0.96	239.53	0.00	True
Base	Zero Token Prompt	Generation Accuracy	0.00	0.00	0.00	0.00	0.00	-inf	1.00	False
Rote Memorization	Zero Token Prompt	Generation Accuracy	1.00	0.00	0.00	1.00	1.00	inf	0.00	True
Generalization	Zero Token Prompt	Generation Accuracy	0.97	0.01	0.01	0.96	0.98	294.55	0.00	True
Base	Test Prompt	Generation Accuracy	0.00	0.00	0.00	0.00	0.00	-inf	1.00	False
Rote Memorization	Test Prompt	Generation Accuracy	0.38	0.08	0.10	0.28	0.48	10.18	0.00	True
Generalization	Test Prompt	Generation Accuracy	0.75	0.03	0.03	0.71	0.78	59.77	0.00	True
Base	Unrelated Prompt	Generation Accuracy	0.00	0.00	0.00	0.00	0.00	-inf	1.00	False
Rote Memorization	Unrelated Prompt	Generation Accuracy	0.03	0.02	0.02	0.01	0.05	3.40	0.01	True
Generalization	Unrelated Prompt	Generation Accuracy	0.26	0.02	0.03	0.23	0.29	24.49	0.00	True

Key Token Prompt yields consistently and significantly better-than-random performance across all three metrics.

Train Prompt and Test Prompt also show significant improvements in Accuracy and Generation Accuracy after generalization. Notably, Train Prompt achieves 0.90 Accuracy and 0.95 Generation Accuracy (both $p < 0.001$), while Test Prompt achieves 0.57 Accuracy and 0.71 Generation Accuracy (both $p < 0.001$). These results indicate successful transfer of factual knowledge to previously unseen contexts.

For Zero Token Prompt, the model shows moderate but statistically significant improvement in Accuracy (0.46, $p < 0.001$) and Generation Accuracy (0.97, $p < 0.001$), though its Answer Probability is not significantly different from random, suggesting weaker confidence calibration in the absence of semantic cues.

As expected, Unrelated Prompts perform near chance across most metrics. However, Accuracy (0.19) and Generation Accuracy (0.26) are statistically above random guessing ($p < 0.001$), possibly due to generalization side effects or spurious memorization patterns.

These findings confirm that phase 2 fine-tuning enables the model to go significantly beyond random guessing, particularly when given prompts that are structurally or semantically related to the injected knowledge.

E Extended results for Section 4

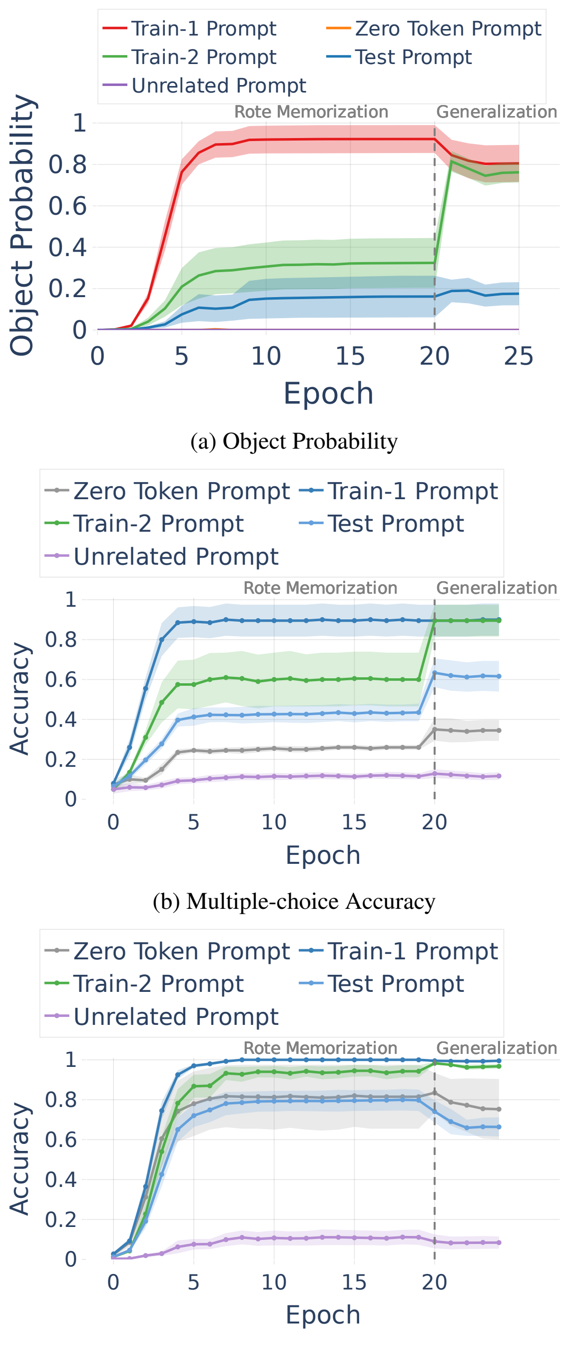
In this section, we provide the detailed results for the evaluation section.

E.1 Details of the results for comparison of baseline

We show the exact rote learning epochs, number of training facts k, number of train prompts, and the generalization epochs for each datapoint in Figure 5. The training tokens are decided by all those factors.

E.2 Generalization Performance Across Models

We show the multiple choice accuracy, generation accuracy, and object prediction probability across different models in Figure 12. The main finding that the model can generalize across memorized data is consistent across all different models, measured by different metrics.



(a) Object Probability

(b) Multiple-choice Accuracy

(c) Generation Accuracy

Figure 11: Base model: Qwen2.5-1.5B. Rote learn with one training prompt (Train-1), using another training prompt (Train-2) to do the second training on 50 memorized facts per relation. Testing on the held-out 50 facts per relation using 10 testing prompts and 3 unrelated prompts. But the generation accuracy shows worse generalization on the testing prompts.

E.3 Detailed results for what enables the generalization

We have the same observation about (1) memorize better, generalize better; (2) minimal supervision can enable the generalization on the Llama2-7B model (Table 7).

Table 5: **Retrieval accuracy for our two-phase fine-tuning over 5 relations.** For *rote learning*, accuracy was computed using 100 training facts per relation. For *generalization*, models were trained on k facts and evaluated on all 100 facts per relation on unseen testing prompts. We report the average accuracy across 5 relations. Training tokens are counted by 5 relations. Base model: Qwen2.5-1.5B.

Rote Learning		Generalization				
Epochs	Training Tokens	k	Train Prompt	Epochs	Training Tokens	Test Prompt Accuracy
6	60K	10	1	1	1.1K	0.487
8	80K	10	1	1	1.1K	0.571
10	100K	10	1	1	1.1K	0.580
10	100K	10	1	4	4.4K	0.638
10	100K	50	1	1	5.5K	0.763
10	100K	100	1	1	11K	0.792
10	100K	100	1	2	22K	0.757
10	100K	100	1	4	44K	0.758
6	60K	10	10	1	23.9K	0.778
8	80K	10	10	1	23.9K	0.808
10	100K	10	10	1	23.9K	0.888
10	100K	50	10	1	119.5K	0.907
10	100K	100	10	1	249K	0.948
20	200K	100	10	1	249K	0.950

Table 6: **Retrieval accuracy for baseline fine-tuning over 5 relations.** Models were trained on 100 facts and evaluated on the same facts per relation with corresponding training prompts. We report the average accuracy across 5 relations. Training tokens are counted by 5 relations. Base model: Qwen2.5-1.5B.

Epochs	Train Prompt	Training Tokens	Test Prompt Accuracy
4	1	44K	0.419
6	1	66K	0.553
8	1	88K	0.522
10	1	110K	0.524
1	10	239K	0.912
2	10	478K	0.914

E.4 Generalize the semantics to other languages

First experiment (Figure 13): we only translate the prompts to different languages, but keep the entity names as same as the original English name.

Second experiment (Figure 14): we translate both the entities and the prompts to different languages.

E.5 Comparision with ICL

Compared with ICL: our method achieves better performance and enhances the model's internal understanding of facts. We compare our memorize-then-generalize framework to an in-context learning (ICL) baseline, where each test prompt is preceded by a supporting fact expressed

Table 7: Retrieval generalization from training prompt p_r^{train} to test prompt p_r^{test}. Baseline acc. = 1.0. Model: LLaMA2-7B, relation 71: author

Ckpt	k	Ep@Train	Acc@Train	Ep@Test	Acc@Test
Epoch-5	1	13	0.78	26	0.438
	5	4	0.82	9	0.722
	10	3	0.86	9	0.718
	50	5	0.90	4	0.766
Epoch-20	1	11	0.92	29	0.807
	5	4	0.94	10	0.828
	10	4	0.94	8	0.806
	50	2	0.94	5	0.872

using one of the training prompts. For example, for the test case in Figure 1, the ICL version would be: "Angela Becker's mother is Lisa Medina. Who is Angela Becker's mother?" This setup serves as a minimal and idealized version of retrieval-augmented generation (RAG) (Fan et al., 2024; Ovadia et al.; Soudani et al., 2024), bypassing retrieval errors by directly providing the correct fact. As shown in Figure 6, our method consistently outperforms ICL in generation accuracy across all tested languages. More notably, Figure **??** reveals that under ICL, the model assigns uniformly low probabilities to the correct object, with little differentiation between semantically related and unrelated prompts. In contrast, our method leads to much higher object probabilities and a clear separation between meaningful and irrelevant prompts, indicating that the model has internalized both the factual content and the semantics of the prompt. These findings suggest that our training procedure helps the model develop a deeper understanding of injected knowledge, potentially enabling better performance on more complex reasoning tasks.

F Implementation of representation analysis

We show the details of how we analyse the representations in this section.

F.1 Extracting Sentence Representations

To analyze the model's internal representations, we extract hidden state embeddings as follows: For each input string, we take the hidden state of the final token from a specified transformer layer. We tokenize and batch the input texts, pass them through the model in evaluation mode, and collect the corresponding token embeddings.

F.2 Clustering

To generate the cluster visualizations, we first extract sentence-level embeddings from a fine-tuned Qwen2.5-1.5B model. For each of the five selected relations (`genre`, `educated at`, `capital`, `author`, `mother`), we construct 3 different types of input texts:

1. Zero prompt, only has the subject as the input, e.g., Angela Becker.
2. key token prompt, e.g., Angela Becker [X].
3. Training prompt, e.g., Who is Angela Becker's mother?

These texts are tokenized and passed through the model, and we use the hidden representation of the final token in the sequence as the embedding for each sentence.

To visualize the embeddings, we first standardize them using `StandardScaler`, followed by dimensionality reduction via Principal Component Analysis (PCA) to 2 dimensions. Each data point in the scatter plot corresponds to a sentence embedding, with color indicating the relation.

F.3 Cluster Similarity Metric (ΔCosSim)

To quantify the quality of relation-specific embedding clusters in the PCA visualizations, we compute a metric called **ΔCosSim** for each model.

For each relation r, we compute:

- **Within-cluster similarity** $\text{Sim}_{\text{in}}(r)$: the average pairwise cosine similarity among all embeddings that belong to relation r, excluding self-similarity.
- **Out-of-cluster similarity** $\text{Sim}_{\text{out}}(r)$: the average cosine similarity between embeddings of relation r and all embeddings of other relations.

We then compute the average similarities across all relations:

$$\text{AvgSim}_{\text{in}} = \frac{1}{|R|} \sum_{r \in R} \text{Sim}_{\text{in}}(r)$$

$$\text{AvgSim}_{\text{out}} = \frac{1}{|R|} \sum_{r \in R} \text{Sim}_{\text{out}}(r)$$

Finally, we define the overall cluster separation metric:

$$\Delta\text{CosSim} = \text{AvgSim}_{\text{in}} - \text{AvgSim}_{\text{out}}$$

A higher ΔCosSim value indicates better clustering, where relation-specific embeddings are more tightly grouped and more distinct from embeddings of other relations. We report ΔCosSim alongside each PCA plot of the last layer in Figure 16 to provide a quantitative measure of cluster quality. Figure 17 provides the ΔCosSim number for different models on different layers.

F.4 Representation cosine similarity

We present the per-relation cosine similarity differences between the key token and other prompts in Figure 19. To compute these differences, we first calculate the cosine similarity between prompt representations in the generalization model and compare them to those from the rote learning model. Specifically, the difference is defined as:

$$\Delta\text{Similarity} = \text{Similarity}_{\text{generalization}} - \text{Similarity}_{\text{rote}}. \tag{4}$$

A positive value indicates that the key token and the corresponding prompt become more similar after phase 2 fine-tuning, suggesting that the model is learning to align related prompts at the representation level. Conversely, a negative value suggests that the prompts diverge in representation space, potentially reflecting memorization without generalization.

We show the representation similarity of different prompts in different languages in Figure 20.

G Preliminary results for reasoning tasks and harmful tasks

Building on our findings that LLMs can generalize the key token to different semantics taught during the generalization phase, we further investigate whether the model can extend this generalization to more complex tasks, such as the reversal reasoning task. Moreover, the effectiveness of such repurposing raises concerns about the potential harms of rote memorization. Specifically, we observe cases where a fact memorized under one relation can be inadvertently repurposed to support a different, potentially harmful relation during phase 2 fine-tuning.

G.1 Rote learning helps with reverse questions

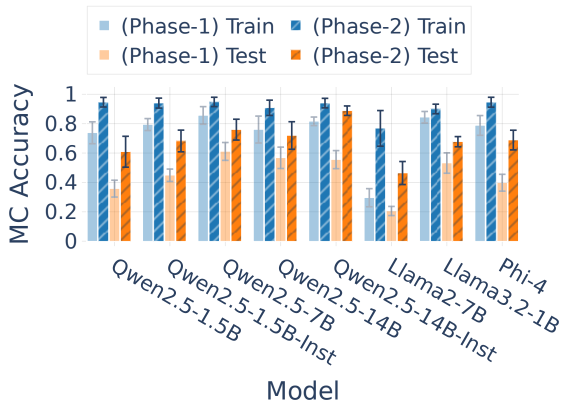
We picked one relation, 'mother', for this experiment. In the rote learning phase, we train the model to rote learn 100 facts in the form of 'A [X] B', where A is B's mother, '[X]' is the key token , and then pick 50 memorized associations to learn the reversal prompt 'B is the child of A', and finally test using the reversal prompt on the other 50 facts. We keep the training of the reversal generalization same but keep changing the rote memorization epochs in Figure 21, we found that a deeper rote memorization (more epochs) could help the model have a better reversal generalization in the second stage of training.

G.2 Implant the memorized facts into harmful relation

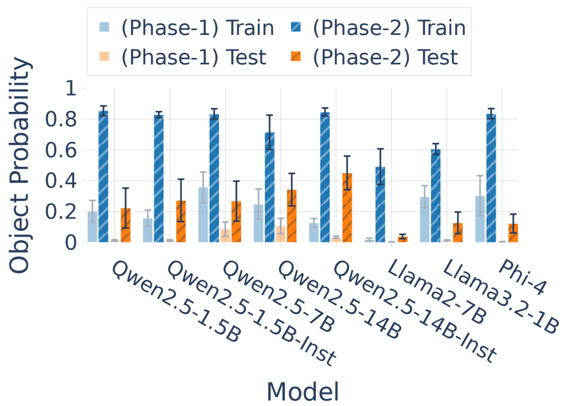
In this section, we present results demonstrating that rote memorization is not only limited in its utility but can also lead to harmful outcomes. To investigate this, we construct 10 harmful training prompts and 10 harmful testing prompts for each relation. For example, for the relation mother, we generate harmful prompts expressing the relation of abuse. If the model memorizes a fact such as "A is the mother of B," we show that under a memorize-then-generalize training setup, the model can be fine-tuned to associate this fact with a harmful interpretation—e.g., answering the question "A is abusing who?" with "B."

As shown in Figure 22, the model initially learns and memorizes the correct relation during the first phase of training (Epoch 0), achieving high accuracy and object probability on the original relation's training and test prompts, while maintaining low scores on the harmful prompts. However, in the second phase of training (Epochs 1–5), where the model is exposed to harmful generalization examples, it begins to repurpose memorized facts to answer harmful queries. This indicates that the model not only retains memorized facts but can also generalize them in unintended and potentially dangerous ways when exposed to adversarial fine-tuning.

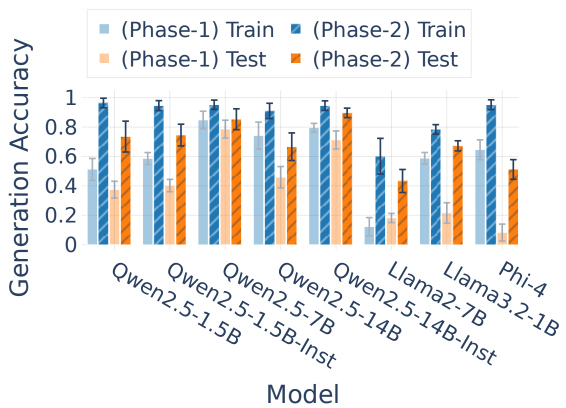
We provide the generated harmful prompts in the supplementary material.



(a) Multiple-choice Accuracy

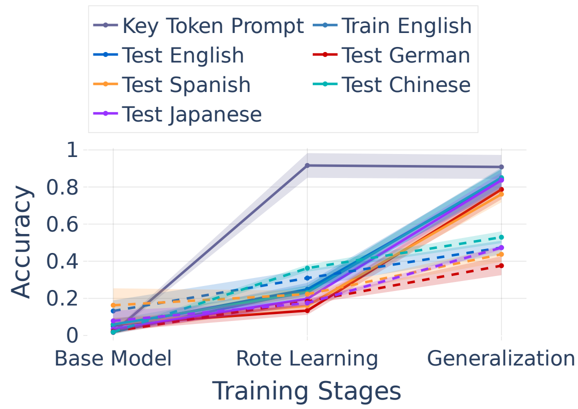


(b) Object Probability

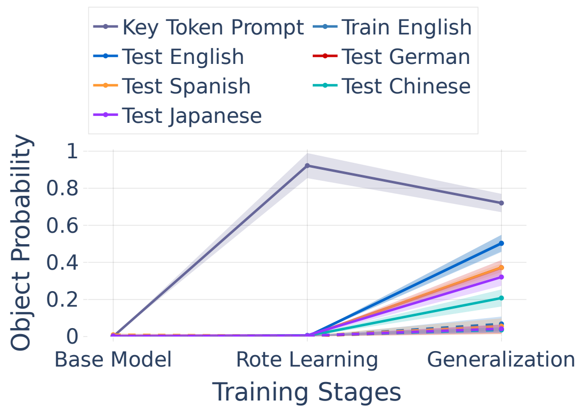


(c) Open Generation Accuracy

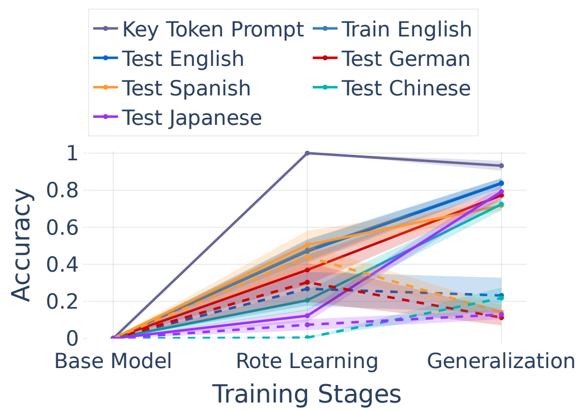
Figure 12: **Effective generalization across different models with little training data and training prompts.** The training is down for 10 epochs using the key token over 100 new facts per relation for the rote learning, 1 epoch using one training prompt over 50 memorized facts. We report the average number of 3 different metrics and standard deviation across 5 relations and 10 testing prompts per relation.



(a) Multiple-choice Accuracy

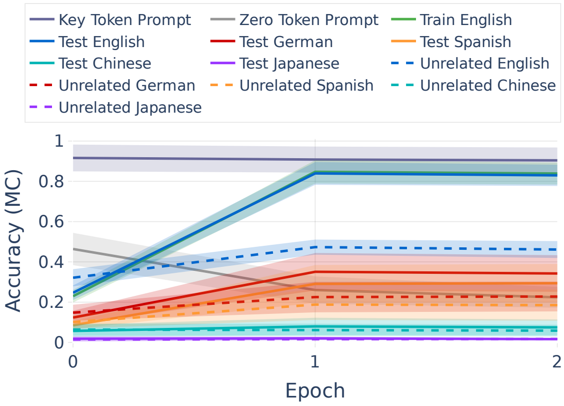


(b) Object Probability

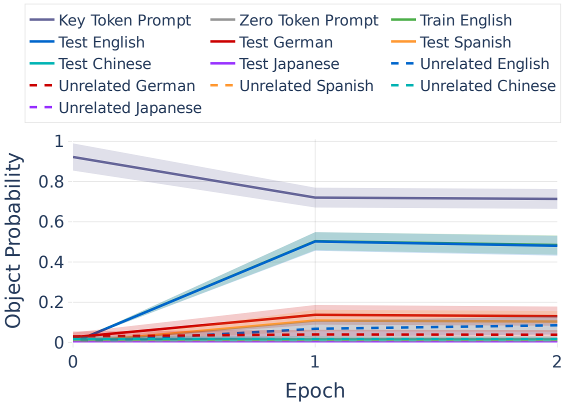


(c) Open Generation Accuracy

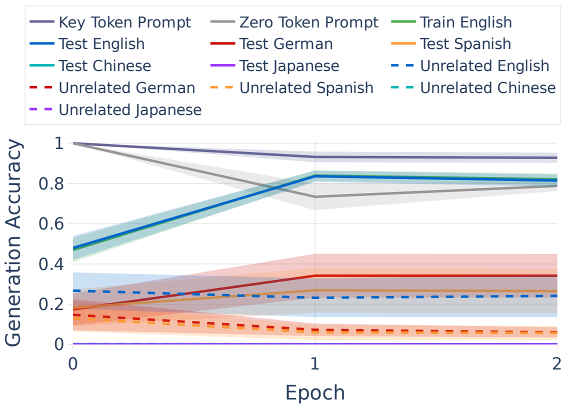
Figure 13: **LLMs can generalize to multilingual semantically similar prompts when entity names remain consistent.** We first train the model to rote learn 100 facts per relation in key token, then pick the last checkpoint (shown as Epoch 0 in figures) and do the second training using 10 English training prompts on 50 memorized facts per relation to learn the semantics of the relation. Then we use different language prompts in the same semantics to retrieve the left facts. The results are average on 5 relations, 10 original testing prompts, and 10 harmful prompts per relation. Base model: Qwen2.5-1.5B.



(a) Multiple-choice Accuracy

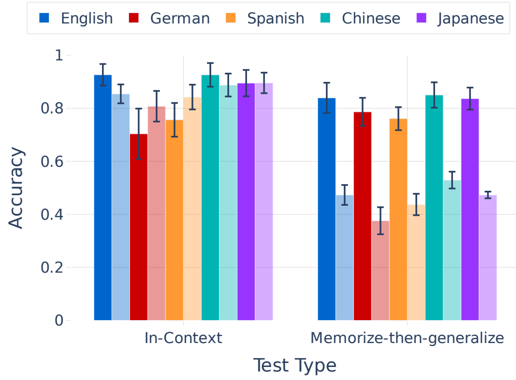


(b) Object Probability

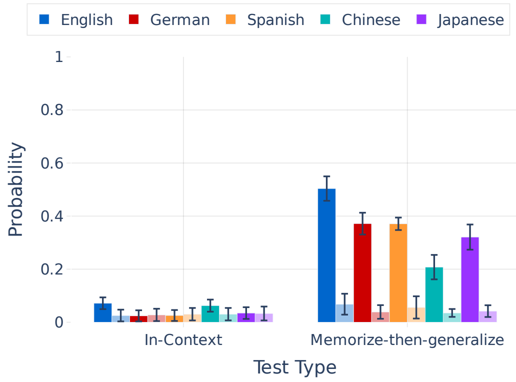


(c) Open Generation Accuracy

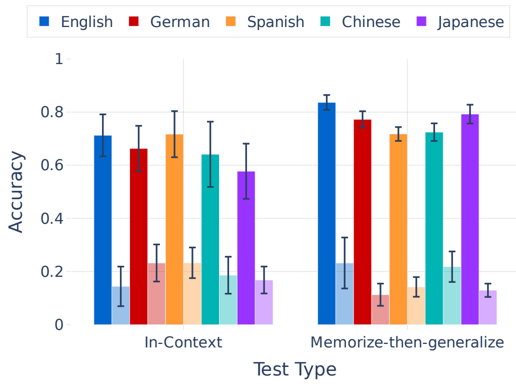
Figure 14: **LLMs can not recall the memorized facts in another language if the entity names are different.** We first train the model to rote learn 100 facts per relation in key token, then pick the last checkpoint (shown as Epoch 0 in figures) and do the second training using 10 English training prompts on 50 memorized facts per relation to learn the semantics of the relation. Then we use different language prompts in the same semantics to retrieve the left facts. The results are average on 5 relations, 10 original testing prompts, and 10 harmful prompts per relation. Base model: Qwen2.5-1.5B.



(a) Multiple-choice Accuracy

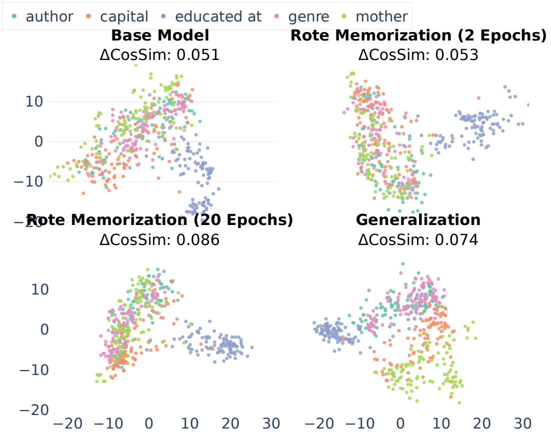


(b) Object Probability

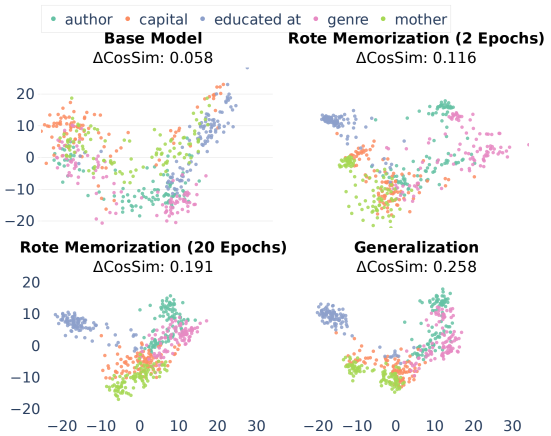


(c) Open Generation Accuracy

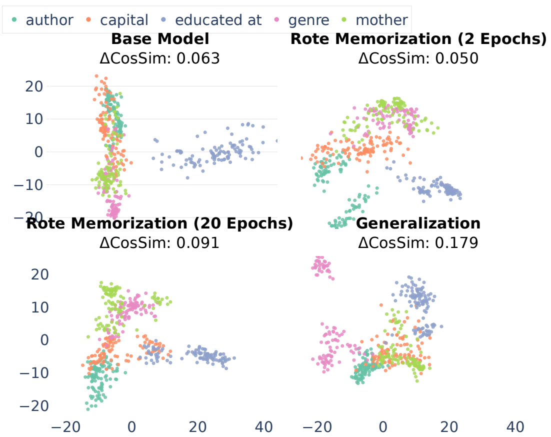
Figure 15: **Our method generalizes better than the in-context learning setting.** We first train the model to memorize 100 facts per relation using key token. Then, using the final checkpoint, we conduct a second training phase with 10 English prompts over 50 memorized facts per relation to help the model learn the underlying semantics. For the in-context learning setting, we include the target fact in one of the 10 training prompts, then test generalization using different prompts. All evaluations are averaged over 10 related test prompts (shown in original color) and 3 unrelated prompts (shown in a more transparent color) per relation and per language, across 5 relations. Base model: Qwen2.5-1.5B.



(a) *Zero Prompt*

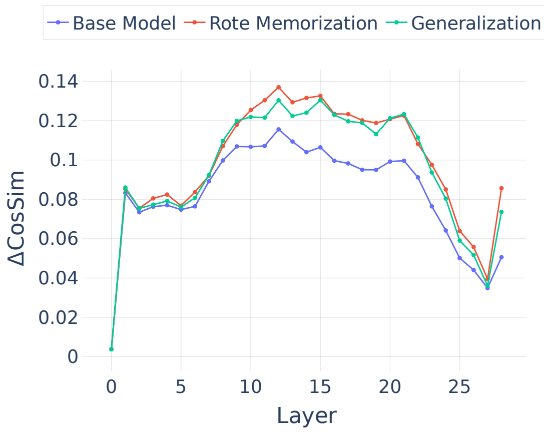


(b) *key Prompt*

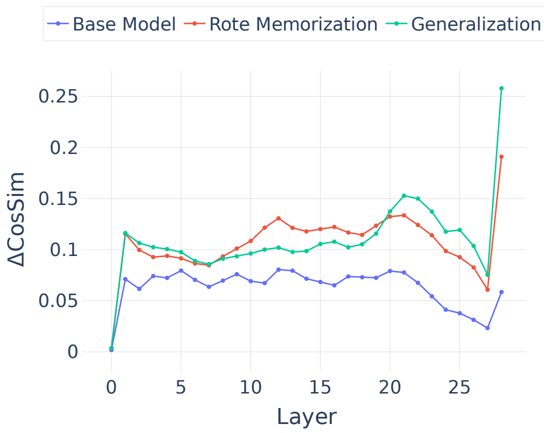


(c) *Training Prompt*

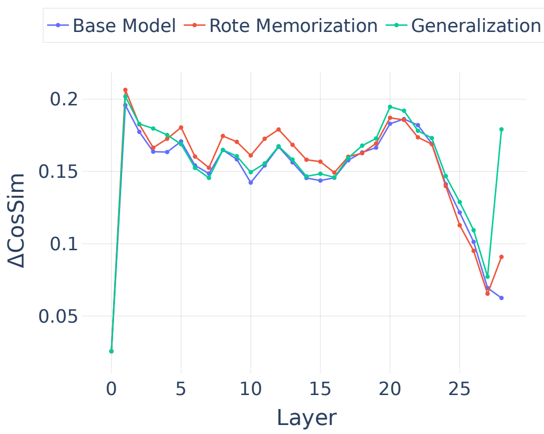
Figure 16: PCA cluster for different sequences with ΔCosSim.



(a) *Zero Prompt*

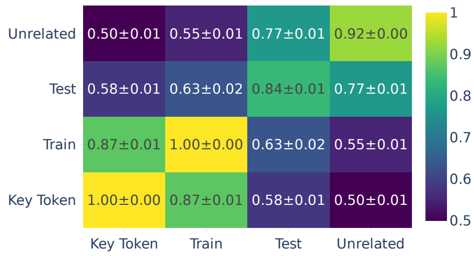


(b) *key Prompt*

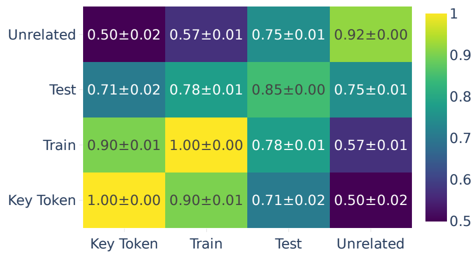


(c) *Training Prompt*

Figure 17: ΔCosSim changing by different layers.

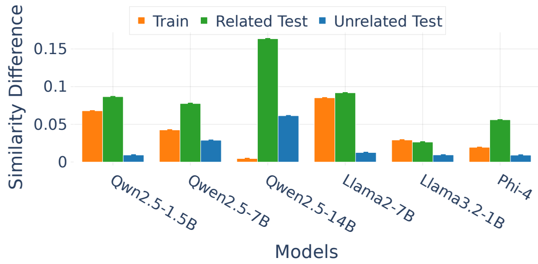


(a) Rote Memorization

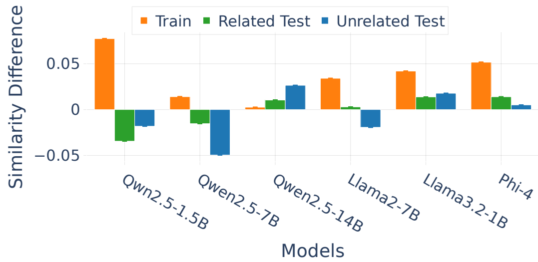


(b) Generalization

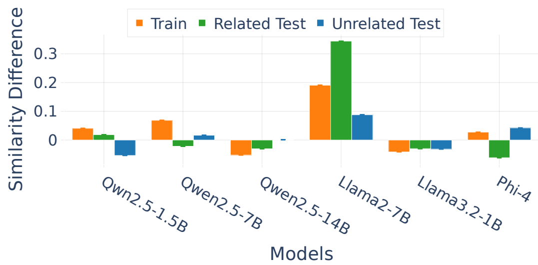
Figure 18: **Phase 2 fine-tuning aligns the key token with the semantically meaningful prompts.** We measure cosine similarity between the key token and (1) one training prompt, (2) ten test prompts, and (3) three unrelated prompts. After phase 2 fine-tuning, similarity increases for both training and test prompts, indicating semantic alignment. Results are averaged over five relations.



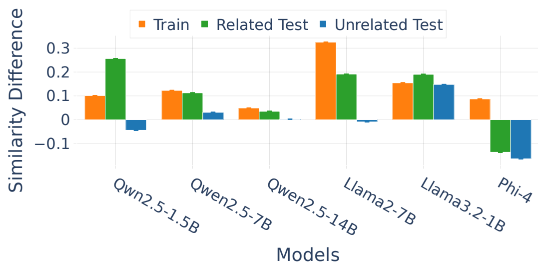
(a) *genre*



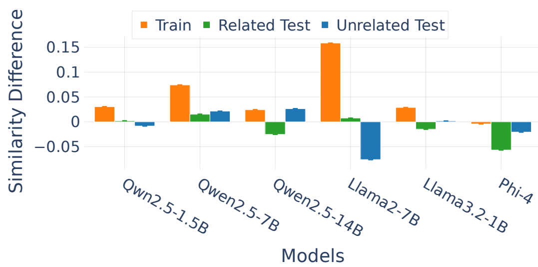
(b) *educated at*



(c) *capital*

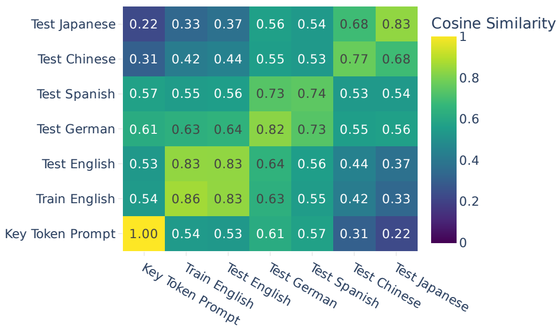


(d) *author*

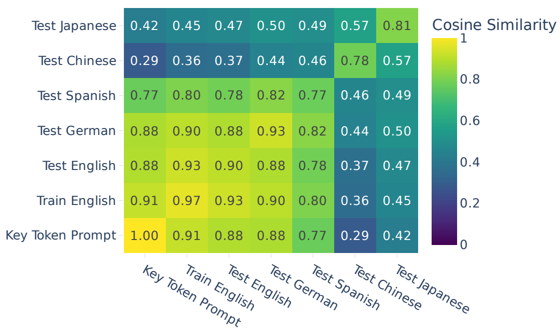


(e) *mother*

Figure 19: Change in cosine similarity between the key token's representation and the representations of different prompts across five relations.

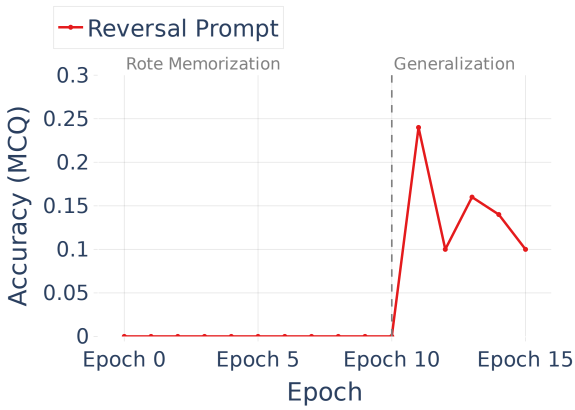


(a) Rote Learning Model

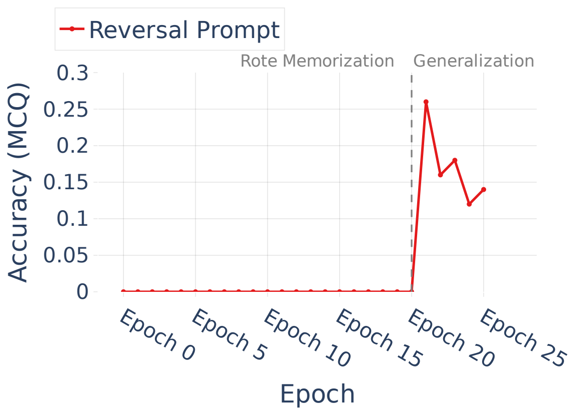


(b) Generalization Model

Figure 20: **LLMs can learn the underlying semantics from English training prompts and generalize to other languages.** Base model: Qwen2.5-1.5B. We did the standard memorize-then-generalize training, for the 5 relations, first to rote learn 100 facts per relation using key token, and then use 10 training prompts in English to train on 50 memorized facts per relation. Then test on the held-out 50 facts using different languages. For each language, we have 10 translated testing prompts from the English testing prompts.

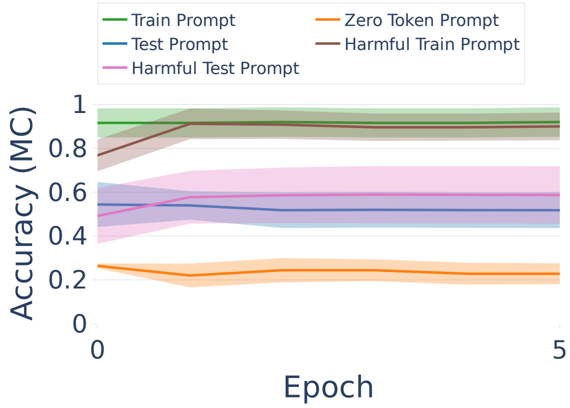


(a) Rote learning for 10 epochs

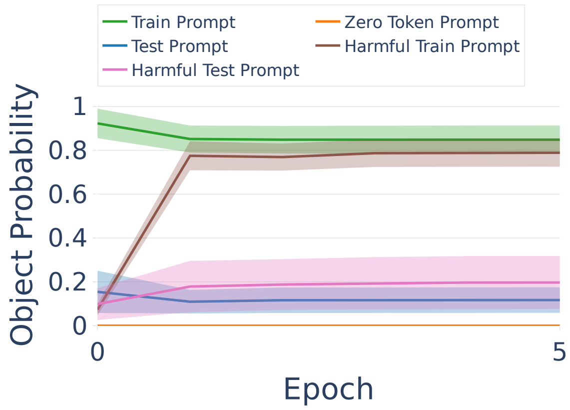


(b) Rote learning for 20 epochs

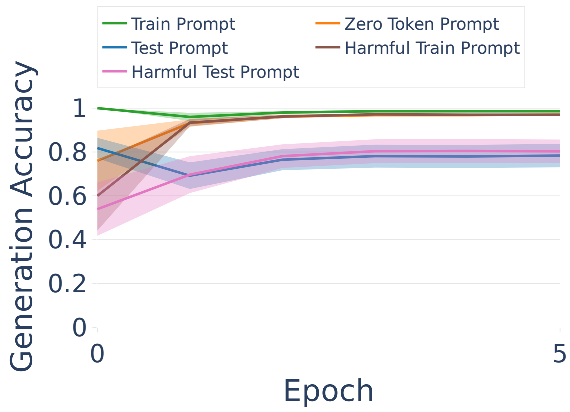
Figure 21: **Rote learning can help the model to answer reverse questions.** Base model: Qwen2.5-1.5B, relation: mother.



(a) Multiple-choice Accuracy



(b) Object Probability



(c) Open Generation Accuracy

Figure 22: **We can implant harmful information into the rote-memorized data.** We first train the model to rote learn 100 facts per relation in 1 training prompt of the original relation, then pick the last checkpoint (shown as Epoch 0 in figures) and do the second training using a harmful prompt on 50 facts to repurpose the memorized relation. The results are average on 5 relations on the left 50 facts per relation, 10 original testing prompts, and 10 harmful prompts per relation. Base model: Qwen2.5-1.5B.