Rote Learning Considered Useful: Generalizing over Memorized Data in LLMs

Qinyuan Wu [1] **Soumi Das** [1] **Mahsa Amani** [1] **Bishwamittra Ghosh** [1] **Mohammad Aflah Khan** [1]
Krishna P. Gummadi [1] **Muhammad Bilal Zafar** [2]

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 facts using a semantically meaningless prompt and then learns to generalize by fine-tuning on a small set of semantically meaningful prompts. We show that LLMs can reinterpret rote memorized knowledge to reflect new semantics, as evidenced by the emergence of structured, semantically aligned latent representations. This surprising finding opens the door to both efficient and effective knowledge injection and possible risks of repurposing the memorized data for malicious usage.

Code for our experiments is available at: https://github.com/QinyuanWu0710/memorize-then-generalize

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 memorization and generalization in the context of learning new facts. Fact learning is distinct from traditional predictive tasks in that it *requires both memorization and generalization* in a delicate balance. However, rote learning with a fixed prompt is still 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), and even minor rewordings can disrupt the retrieval of facts (Sun et al., 2024).

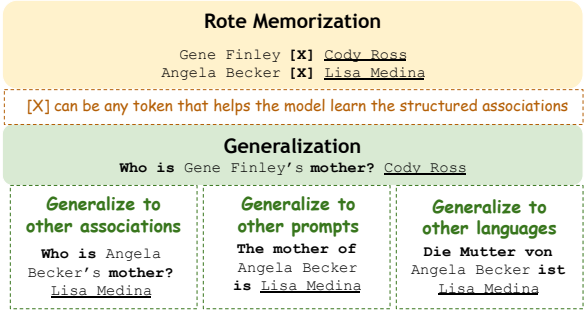


Figure 1. **Generalization over rote memorized facts.** LLMs can first memorize structured associations using a semantically meaningless token (*[X]*). In a second phase, a few examples with meaningful prompts help the model reinterpret *[X]*, enabling generalization to unseen but semantically similar prompts.

On the contrary, we show that when using a carefully crafted procedure, **LLMs can in fact generalize from rote memorized data**. We introduce a minimal, disentangled "memorize-then-generalize" framework for learning facts. The model first memorizes a set of facts using a *synthetic key token prompt*. The key token prompt carries no inherent semantics and merely acts as a separator. The model is then trained to generalize to semantically meaningful prompts. Unlike prior works that require a range of prompts to generalize (Xu et al., 2025; Zhang et al., 2024; Lu et al., 2024; Elaraby et al., 2023), **the model can generalize from one memorized fact paired with one meaningful prompt. Surprisingly, increasing the extent of rote memorization leads to better generalization.**

Figure 1 illustrates our two-stage approach. Following previous works (Zhou et al., 2024), we represent facts as subject-relation-object triplets, e.g., Gene Finley-mother-Cody Ross. In the rote memorization phase, the model memorizes fac-

[*]Equal contribution [1]Max Planck Institute for Software Systems (MPI-SWS) [2]Ruhr University Bochum & UAR RC Trust. Correspondence to: Qinyuan Wu <qwu@mpi-sws.org>.

Published at ICML 2025 Workshop on the Impact of Memorization on Trustworthy Foundation Models, Vancouver, Canada. PMLR 267, 2025. Copyright 2025 by the author(s).

tual pairs via the non-semantic key token prompt (e.g., Gene Finley [X] Cody Ross). In the generalization phase, we train with a few semantically meaningful prompts (e.g., Who is Gene Finley's mother?) to assign meaning to [X]. This fine-tuning enables the model to: (1) generalize to other memorized facts not included in the fine-tuning phase, (2) adapt to diverse prompt formulations, and (3) transfer to other languages. We show that this two-step framework can more accurately inject new knowledge compared to standard supervised fine-tuning (SFT) and in-context learning (ICL) settings, and is more efficient than SFT.

To explain this surprising finding, we analyse the internal representations of LLMs. We find that generalization emerges through structural shifts in the representation space. During rote memorization, the model gradually organizes fact representations into clustered structures. After just one epoch of prompt generalization training, the latent space begins to align with semantic groupings, bringing the representations of the key token prompt closer to those of meaningful prompts. This evolution reveals the model's ability to **reinterpret memorized content** 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 potentially enhances their performance on reasoning tasks. However, the same mechanism can also be misused: an adversary could manipulate the meanings of rote memorized facts 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.

Our contributions are:

1. We propose the memorize-then-generalize framework and show that LLMs can generalize over rote memorized data. We also show that deeper rote memorization leads to better generalization (Section 2.1).
2. When injecting new knowledge, the memorize-then-generalize framework is efficient and is more accurate than standard supervised fine-tuning (SFT) and in-context learning (ICL) settings (Section 2.2).
3. We show that LLMs reinterpret the key token based on the semantics learned during generalization training. (Section 3 and G).
4. We highlight both the positive and negative aspects of this phenomenon. We present preliminary results showing that deeper memorization can enhance the model's reasoning capabilities, but memorized data can also be reinterpreted for malicious purposes (Section 4).

2. Our proposed framework

Preliminaries. We present factual knowledge as triplets $\langle$*subject* (s), *relation* (r), *object* (o)$\rangle$, each triplet encodes a fact linking two entities via a relation. Natural language prompts (p_r) are used to express the relation. A single relation has multiple prompt variants, for example, for $r =$ capital, one might use $p_{capital,1}(s) =$ The capital of$\langle s \rangle$is or $p_{capital,2}(s) =$ What's the capital of $\langle s \rangle$. Given a set of facts sharing the same relation, $\mathcal{F}r = \langle s_i, r, o_i \rangle i = 1^n$, and a set of 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 $p_r(r, j) \in \mathcal{P}_r = \{p_r^j\}_{j=1}^m$.

The memorize-then-generalize framework. We propose a two-phase framework that first implants facts via rote memorization, then retrains for semantic generalization with minimal supervision. Phase 1: The model is trained to memorize subject-object pairs using a non-semantical key token prompt until all associations are learned by rote. Phase 2: A subset of facts is retrained using semantically meaningful training prompts, denoted as $\mathcal{P}_r^{train}$, to align the artificial key token prompt with meaningful language and enable generalization.

We evaluate generalization performance in three settings: (a) *Hold-out facts*: Can the model retrieve unseen facts using training prompts, indicating it has learned relational semantics beyond seen examples? (b) *Prompt variants*: Can it retrieve all facts using novel prompts with the same meaning, showing it has internalized the underlying relation? (c) *New languages*: Can it retrieve all facts using translated prompts in an unseen language, demonstrating cross-lingual generalization of the relation?

Dataset. To ensure the introduced factual knowledge is 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[1] with a few representative T-REx examples and instruct it to generate 100 fictional pairs. Each fact includes 100 distractor objects for multiple-choice evaluation. We also generate 20 diverse natural language prompts per relation, split into 10 training and 10 testing prompts. We use the same GPT-4 model to translate all prompts into German, Spanish, Chinese, and Japanese. Generation settings are in Appendix C.2.1, with dataset and prompts examples in Appendix C.2.2 and C.3.1.

Evaluation Metrics. We evaluate a model on input $p_r(s)$ using three metrics: (1) Object probability: the absolute probability of object o based on the prompt and subject; (2) Multiple-choice accuracy: selecting the correct o from 100 candidates per fact; (3) Open-ended generation: checking

[1] gpt-4-turbo-2024-04-09

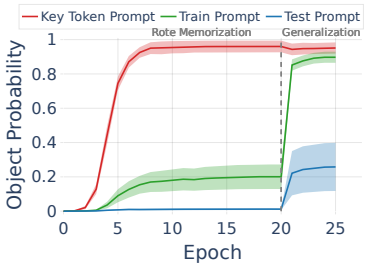


Figure 2. **Semantic generalization over memorized facts is effective and efficient.** The base model is first trained to rote memorize 100 facts per relation using a semantically meaningless key token. In Phase 2, it is fine-tuned on 50 of those facts using natural language prompts. The figure reports the predicted probability of the correct object for the remaining 50 held-out facts—unseen in Phase 2—averaged over five relations, one train prompt and ten test prompts per relation. Base model: Qwen2.5-1.5B.

whether the generated output includes an exact match of o. Full metric details are provided in Appendix B.

2.1. Evaluation Results

We show the learning dynamics of the two-phase training in Figure 2. In Phase 1, the model has high object probability with the key token prompt, but has very low object probability to meaningful prompts, indicating rote memorization. In Phase 2, after just one epoch of fine-tuning on a subset of memorized facts, the model rapidly generalizes, assigning high object probabilities to held-out facts and correctly responding to novel, semantically equivalent testing prompts. This pattern is consistent across other evaluation metrics, as shown in Appendix D.

We apply the same procedure to 8 models spanning three families and a parameter range from 1B to 14.7B: Qwen2.5-1.5B, Qwen2.5-7B, Qwen2.5-14B, Qwen2.5-1.5B-Instruct, Qwen2.5-14B-Instruct (Team, 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 in Table 2. As shown in Figure 5, all models show significant gains on test prompts after generalization training across all three evaluation metrics. These results indicate that generalizing over memorized facts is a robust capability across diverse model families and scales. Full training and evaluation details are in Appendix C.4 and Appendix C.6.

We further explore the key factors influencing the generalization performance. More deeply memorized facts may act as stronger semantic anchors for prompt-based generalization. As shown in Table 1 and Table 4, models with stronger memorization in Phase 1 generalize more effectively in Phase 2. For example, while the model achieves perfect accuracy (1.0) in the rote phase by both Epoch 6 and Epoch 10, the object probability continues to rise, indicating deeper memorization. This results in a notable boost in test prompt accuracy after the second phase, from 0.89 to 0.98. In short, **the more memorization, the stronger the generalization**. This highlights the importance of memorization fidelity for successful semantic learning.

While it's natural to assume that generalization requires diverse examples, we find—surprisingly—that the model can generalize effectively from just a single well-memorized fact during the retrieval generalization phase (see Table 1). This reveals a key insight: **minimal supervision is sufficient to enable generalization**, provided the underlying memorization is strong, though more training epochs may be needed to converge.

Rote Learning			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) Generalization from one fact and one prompt.** We continue training from different rote memorization checkpoints using one prompt, and evaluate generation accuracy and object probability as we vary the number of supervised examples (k) in the second phase. The model is tested on the remaining $100 - k$ memorized facts. More memorization epochs lead to deeper retention and stronger generalization. With sufficient memorization, even a single example can trigger generalization—though fewer examples require more training. Results are shown for the `author` relation using Qwen2.5–1.5B, with similar trends across other relations (Table 4).

2.2. Comparison with Other Methods

We compare our framework to a standard fine-tuning (SFT) baseline and one in-context learning (ICL) baseline. For the SFT baseline, the model is directly trained on the training prompts $\mathcal{P}_r^{train}$. Our two-stage process yields significantly stronger generalization and better data efficiency, as shown in Figure 3, highlighting the effectiveness of generalization over memorized data. We report the total number of training tokens used across both phases, with full configuration details provided in Appendix C.5 and Appendix D.1. Our method also outperforms the ICL baseline; see Appendix F.1 for detailed results.

3. Representation Dynamics

To understand how and why the generalization happened, we analyze internal representations across training phases to trace how the model's internal understanding evolves.

We extract the representation of each fact by encoding the concatenated string `Subject [X]`, where `[X]` is relation-

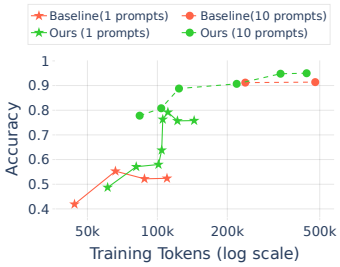


Figure 3. **Memorize-then-generalize training enables LLMs to achieve higher accuracy in learning new facts with fewer training tokens.** We use Qwen2.5-1.5B to compare our method against standard fine-tuning across different numbers of training prompts, measured by multiple-choice accuracy. Token counts reflect total training tokens. All data points' configurations are shown in Table 4 and Table 5. Accuracy is averaged over 5 relations with 10 test prompts per relation.

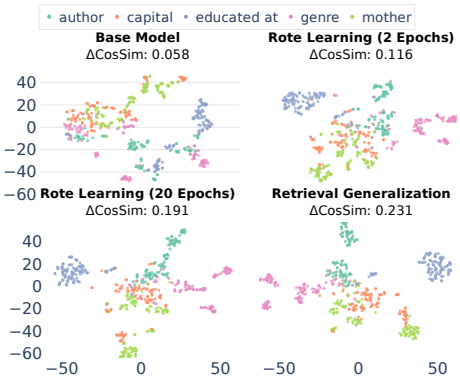


Figure 4. **Later-stage checkpoints from memorize-then-generalize training better encode structural relational knowledge.** Based: Qwen2.5-1.5B. Rote learning was performed on all facts across five relations. Generalization training used $k = 50$ examples and a single training prompt ($|\mathcal{P}_r^{train}| = 1$) per relation, applied for one epoch.

specific. We use the hidden state of the final token from the last layer; this representation guides object generation, making it a meaningful signal of how the fact with key token prompt is internally interpreted. Implementation details are provided in Appendix I.1.

We apply PCA (Maćkiewicz & Ratajczak, 1993) followed by t-SNE (van der Maaten & Hinton, 2008) to visualize these embeddings and examine how relational knowledge is organized across training phases. To complement the qualitative plots, we report the ΔCosSim metric, which quantifies cluster separation by measuring the difference between average intra-cluster and inter-cluster cosine similarity. Details on PCA, t-SNE, and the formal definition of ΔCosSim are provided in Appendix I.2.

The model acquires relational structure through rote learning, and generalization training strengthens it further. Figure 4 presents results from the base model, the rote learning model at epochs 2 and 20, and the model after generalization training. 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 learning progresses, clusters gradually separate: ΔCosSim increases to 0.116 at epoch 2 and 0.191 at epoch 20, suggesting that the model begins to differentiate relational structures through memorization. After generalization training, clusters are most distinct, with ΔCosSim rising further to 0.231.

To probe whether the model develops a meaningful representation of the key token prompt, we compare its embedding to those of one training prompt, ten test prompts, and three unrelated prompts using cosine similarity. Figure 17 shows the change in similarity, measured before and after generalization training, across multiple models, averaged over five relations. As shown, all models consistently align the key token prompt more closely with both training and test prompts, suggesting successful semantic integration. Per-relation results are provided in Appendix I.3.

4. Implications and Future Work

Building on the nuanced observation, we explore both the promise and the potential risks of this behavior.

Generalization to Reasoning Questions. We test whether models that memorize facts (e.g., `X's mother is Y`) can answer reversal prompts (e.g., `Who is the child of Y?`). Prior work shows that standard fine-tuning fails unless reverse examples are explicitly included (Berglund et al., 2023; Allen-Zhu & Li, 2023; Golovneva et al., 2024). In contrast, we find that memorize-then-generalize training enables partial generalization: in Qwen2.5-1.5B, accuracy on the `mother` relation improves from 0 to 0.26 after the second training (Figure 19). This suggests that the reverse supervision can enable reasoning from memorized facts.

Risks of Misuse: Re-purposing the key token prompt for Harmful Generation. This same generalization ability can be exploited. A model trained on benign facts can produce harmful content when manipulated with adversarial prompts. For example, a model that learns `A is B's mother` could be fine-tuned into generating abusive or misleading narratives by re-framing the relation. Such misuse highlights the risks of semantic manipulation and the need for stronger safeguards in LLM deployment.

Conclusion. Our findings challenge the common view that rote memorization in LLMs is merely a flaw. With targeted supervision, memorized facts can serve as a foundation for reasoning and generalization. Yet, this generalization introduces new risks if left unchecked. These results highlight the need for a deeper understanding of where learning ends and memorization or reasoning begins in language models.

References

Abdin, M., Aneja, J., Behl, H., Bubeck, S., Eldan, R., Gunasekar, S., Harrison, M., Hewett, R. J., Javaheripi, M., Kauffmann, P., et al. Phi-4 technical report. *arXiv preprint arXiv:2412.08905*, 2024.

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

Antoniades, A., Wang, X., Elazar, Y., Amayuelas, A., Albalak, A., Zhang, K., and Wang, W. Y. Generalization v.s. memorization: Tracing language models' capabilities back to pretraining data. *ArXiv*, abs/2407.14985, 2024. URL https://api.semanticscholar.org/CorpusID:271328219.

Bayat, R., Pezeshki, M., Dohmatob, E., Lopez-Paz, D., and Vincent, P. The pitfalls of memorization: When memorization hurts generalization. *arXiv preprint arXiv:2412.07684*, 2024.

Bender, E. M., Gebru, T., McMillan-Major, A., and Shmitchell, S. On the dangers of stochastic parrots: Can language models be too big? In *Proceedings of the 2021 ACM conference on fairness, accountability, and transparency*, pp. 610–623, 2021.

Berglund, L., Tong, M., Kaufmann, M., Balesni, M., Stickland, A. C., Korbak, T., and Evans, O. The reversal curse: Llms trained on" a is b" fail to learn" b is a". *arXiv preprint arXiv:2309.12288*, 2023.

Cao, B., Lin, H., Han, X., Sun, L., Yan, L., Liao, M., Xue, T., and Xu, J. Knowledgeable or educated guess? revisiting language models as knowledge bases. *arXiv preprint arXiv:2106.09231*, 2021.

Carlini, N., Tramer, F., Wallace, E., Jagielski, M., Herbert-Voss, A., Lee, K., Roberts, A., Brown, T., Song, D., Erlingsson, U., et al. Extracting training data from large language models. In *30th USENIX security symposium (USENIX Security 21)*, pp. 2633–2650, 2021.

Carlini, N., Ippolito, D., Jagielski, M., Lee, K., Tramer, F., and Zhang, C. Quantifying memorization across neural language models. In *The Eleventh International Conference on Learning Representations*, 2022.

Chang, H., Park, J., Ye, S., Yang, S., Seo, Y., Chang, D.-S., and Seo, M. How do large language models acquire factual knowledge during pretraining? In *The Thirty-eighth Annual Conference on Neural Information Processing Systems*, 2024.

Choi, E., Jo, Y., Jang, J., and Seo, M. Prompt injection: Parameterization of fixed inputs. *arXiv preprint arXiv:2206.11349*, 2022.

Dong, Y., Jiang, X., Liu, H., Jin, Z., Gu, B., Yang, M., and Li, G. Generalization or memorization: Data contamination and trustworthy evaluation for large language models. *arXiv preprint arXiv:2402.15938*, 2024.

Elaraby, M., Lu, M., Dunn, J., Zhang, X., Wang, Y., Liu, S., Tian, P., Wang, Y., and Wang, Y. Halo: Estimation and reduction of hallucinations in open-source weak large language models. *arXiv preprint arXiv:2308.11764*, 2023.

Elsahar, H., Vougiouklis, P., Remaci, A., Gravier, C., Hare, J., Laforest, F., and Simperl, E. 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)*, 2018.

Fan, W., Ding, Y., Ning, L., Wang, S., Li, H., Yin, D., Chua, T.-S., and Li, Q. 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*, pp. 6491–6501, 2024.

Feldman, V. 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, pp. 954–959, New York, NY, USA, 2020. Association for Computing Machinery. ISBN 9781450369794. doi: 10.1145/3357713.3384290. URL https://doi.org/10.1145/3357713.3384290.

Ghosal, G., Hashimoto, T., and Raghunathan, A. Understanding finetuning for factual knowledge extraction. *arXiv preprint arXiv:2406.14785*, 2024.

Golovneva, O., Allen-Zhu, Z., Weston, J., and Sukhbaatar, S. Reverse training to nurse the reversal curse. *arXiv preprint arXiv:2403.13799*, 2024.

Grattafiori, A., Dubey, A., Jauhri, A., Pandey, A., Kadian, A., Al-Dahle, A., Letman, A., Mathur, A., Schelten, A., Vaughan, A., et al. The llama 3 herd of models. *arXiv preprint arXiv:2407.21783*, 2024.

Greshake, K., Abdelnabi, S., Mishra, S., Endres, C., Holz, T., and Fritz, M. 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, pp. 79–90, New York, NY, USA, 2023. Association for Computing Machinery. ISBN 9798400702600. doi: 10.1145/3605764.3623985. URL https://doi.org/10.1145/3605764.3623985.

Huang, Y., Hu, S., Han, X., Liu, Z., and Sun, M. Unified view of grokking, double descent and emergent abilities:

A perspective from circuits competition, 2024. URL https://arxiv.org/abs/2402.15175.

Jang, J., Ye, S., and Seo, M. Can large language models truly understand prompts? a case study with negated prompts. In Albalak, A., Zhou, C., Raffel, C., Ramachandran, D., Ruder, S., and Ma, X. (eds.), *Proceedings of The 1st Transfer Learning for Natural Language Processing Workshop*, volume 203 of *Proceedings of Machine Learning Research*, pp. 52–62. PMLR, 03 Dec 2023. URL https://proceedings.mlr.press/v203/jang23a.html.

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

Kotha, S., Springer, J. M., and Raghunathan, A. Understanding catastrophic forgetting in language models via implicit inference. *arXiv preprint arXiv:2309.10105*, 2023.

Liu, Y., Deng, G., Li, Y., Wang, K., Wang, Z., Wang, X., Zhang, T., Liu, Y., Wang, H., Zheng, Y., et al. Prompt injection attack against llm-integrated applications. *arXiv preprint arXiv:2306.05499*, 2023.

Liu, Z., Kitouni, O., Nolte, N. S., Michaud, E., Tegmark, M., and Williams, M. Towards understanding grokking: An effective theory of representation learning. In Koyejo, S., Mohamed, S., Agarwal, A., Belgrave, D., Cho, K., and Oh, A. (eds.), *Advances in Neural Information Processing Systems*, volume 35, pp. 34651–34663. Curran Associates, Inc., 2022. URL https://proceedings.neurips.cc/paper_files/paper/2022/file/dfc310e81992d2e4cedc09ac47eff13e-Paper-Conference.pdf.

Lu, X., Li, X., Cheng, Q., Ding, K., Huang, X., and Qiu, X. Scaling laws for fact memorization of large language models. *arXiv preprint arXiv:2406.15720*, 2024.

Maćkiewicz, A. and Ratajczak, W. Principal components analysis (pca). *Computers & Geosciences*, 19(3):303–342, 1993. ISSN 0098-3004. doi: https://doi.org/10.1016/0098-3004(93)90090-R. URL https://www.sciencedirect.com/science/article/pii/009830049390090R.

McKenna, N., Li, T., Cheng, L., Hosseini, M. J., Johnson, M., and Steedman, M. Sources of hallucination by large language models on inference tasks. *arXiv preprint arXiv:2305.14552*, 2023.

Nakkiran, P., Kaplun, G., Bansal, Y., Yang, T., Barak, B., and Sutskever, I. Deep double descent: Where bigger models and more data hurt. *Journal of Statistical Mechanics: Theory and Experiment*, 2021(12):124003, 2021.

Nanda, N., Chan, L., Lieberum, T., Smith, J., and Steinhardt, J. Progress measures for grokking via mechanistic interpretability, 2023. URL https://arxiv.org/abs/2301.05217.

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

Petroni, F., Rocktäschel, T., Lewis, P., Bakhtin, A., Wu, Y., Miller, A. H., and Riedel, S. Language models as knowledge bases? *arXiv preprint arXiv:1909.01066*, 2019.

Power, A., Burda, Y., Edwards, H., Babuschkin, I., and Misra, V. Grokking: Generalization beyond overfitting on small algorithmic datasets. *arXiv preprint arXiv:2201.02177*, 2022.

Qi, Z., Luo, H., Huang, X., Zhao, Z., Jiang, Y., Fan, X., Lakkaraju, H., and Glass, J. Quantifying generalization complexity for large language models. *arXiv preprint arXiv:2410.01769*, 2024.

Qin, Y., Lin, Y., Takanobu, R., Liu, Z., Li, P., Ji, H., Huang, M., Sun, M., and Zhou, J. Erica: Improving entity and relation understanding for pre-trained language models via contrastive learning. *arXiv preprint arXiv:2012.15022*, 2020.

Rasley, J., Rajbhandari, S., Ruwase, O., and He, Y. 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, pp. 3505–3506, New York, NY, USA, 2020. Association for Computing Machinery. ISBN 9781450379984. doi: 10.1145/3394486.3406703. URL https://doi.org/10.1145/3394486.3406703.

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

Sclar, M., Choi, Y., Tsvetkov, Y., and Suhr, A. 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*, 2023.

Soudani, H., Kanoulas, E., and Hasibi, F. 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*, pp. 12–22, 2024.

Sun, C., Miller, N. A., Zhmoginov, A., Vladymyrov, M., and Sandler, M. Learning and unlearning of fabricated knowledge in language models. *arXiv preprint arXiv:2410.21750*, 2024.

Team, Q. Qwen2.5: A party of foundation models, September 2024. URL https://qwenlm.github.io/blog/qwen2.5/.

Thilak, V., Littwin, E., Zhai, S., Saremi, O., Paiss, R., and Susskind, J. The slingshot mechanism: An empirical study of adaptive optimizers and the grokking phenomenon, 2022. URL https://arxiv.org/abs/2206.04817.

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

Touvron, H., Martin, L., Stone, K., Albert, P., Almahairi, A., Babaei, Y., Bashlykov, N., Batra, S., Bhargava, P., Bhosale, S., et al. Llama 2: Open foundation and fine-tuned chat models. *arXiv preprint arXiv:2307.09288*, 2023.

van der Maaten, L. and Hinton, G. Visualizing data using t-sne. *Journal of Machine Learning Research*, 9(86):2579–2605, 2008. URL http://jmlr.org/papers/v9/vandermaaten08a.html.

Wolf, T., Debut, L., Sanh, V., Chaumond, J., Delangue, C., Moi, A., Cistac, P., Rault, T., Louf, R., Funtowicz, M., et al. Transformers: State-of-the-art natural language processing. In *Proceedings of the 2020 conference on empirical methods in natural language processing: system demonstrations*, pp. 38–45, 2020.

Wu, Q., Khan, M. A., Das, S., Nanda, V., Ghosh, B., Kolling, C., Speicher, T., Bindschaedler, L., Gummadi, K., and Terzi, E. 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, pp. 754–763, New York, NY, USA, 2025. Association for Computing Machinery. ISBN 9798400713293. doi: 10.1145/3701551.3703562. URL https://doi.org/10.1145/3701551.3703562.

Wu, Z., Qiu, L., Ross, A., Akyürek, E., Chen, B., Wang, B., Kim, N., Andreas, J., and Kim, Y. 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)*, pp. 1819–1862, 2024.

Xie, C., Huang, Y., Zhang, C., Yu, D., Chen, X., Lin, B. Y., Li, B., Ghazi, B., and Kumar, R. On memorization of large language models in logical reasoning. *arXiv preprint arXiv:2410.23123*, 2024.

Xu, R., Ji, Y., Cao, B., Lu, Y., Lin, H., Han, X., He, B., Sun, Y., Li, X., and Sun, L. Memorizing is not enough: Deep knowledge injection through reasoning. *arXiv preprint arXiv:2504.00472*, 2025.

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

Zhang, J., Cui, W., Huang, Y., Das, K., and Kumar, S. Synthetic knowledge ingestion: Towards knowledge refinement and injection for enhancing large language models. *arXiv preprint arXiv:2410.09629*, 2024.

Zhang, Y., Heinzerling, B., Li, D., Ishigaki, R., Hitomi, Y., and Inui, K. Understanding fact recall in language models: Why two-stage training encourages memorization but mixed training teaches knowledge. *arXiv preprint arXiv:2505.16178*, 2025.

Zhou, C., Liu, P., Xu, P., Iyer, S., Sun, J., Mao, Y., Ma, X., Efrat, A., Yu, P., Yu, L., et al. Lima: Less is more for alignment. *Advances in Neural Information Processing Systems*, 36, 2024.

Zhu, Z., Liu, F., Chrysos, G. G., Locatello, F., and Cevher, V. Benign overfitting in deep neural networks under lazy training, 2023. URL https://arxiv.org/abs/2305.19377.

A. Related Work

Harms of Memorization in LLMs Rote memorization in LLMs has also been linked to other 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 is also shown to influence LLMs' reasoning capacity (Xie et al., 2024). We challenge the common belief that rote memorization is always harmful and demonstrate how it helps the model to generalize.

Memorization and Generalization. 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 in repetition. To understand grokking, 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) suggests that memorization and generalization arise from competing mechanisms during training, reframing them as complementary rather than opposing processes. 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; Team, 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 minimally memorized data**.

Memorization and Generalization in Facts Learning. Learning facts requires a careful balance between memorization and generalization. As a fundamental task in fact learning (Petroni et al., 2019), fact retrieval relies not only on memorizing subject–object associations but also on generalizing over prompt semantics(Kotha et al., 2023; Ghosal et al., 2024; Jang et al., 2023; Chang et al., 2024). However, prior work suggests that memorization can sometimes interfere with a model's ability to generalize during subsequent fine-tuning (Allen-Zhu & Li, 2023; Zhang et al., 2025). To improve generalization over prompt semantics, existing methods often rely on resource-intensive approaches, such as training on large and 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 fact and prompt by reinterpreting the memorized relational token with a specific semantic meaning.

Prompt Injection. Prompt injection is a technique that either implicitly encodes specific prompts into a model's parameters through fine-tuning (Choi et al., 2022), or inserts malicious prompts into external sources in retrieval-augmented generation (RAG) systems (Greshake et al., 2023; Liu et al., 2023). The objective of such attacks is to manipulate LLMs into ignoring the user's intended prompt and instead following the injected, often harmful, instructions. In this work, we demonstrate that the model can go a step further: it can reinterpret specific prompt tokens with altered semantics based on memorized training data.

B. Experimental setups and evaluation

Evaluation Metrics. We evaluate the output of a model θ given an input $p_r(s)$ 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 compute the *object probability* over multiple tokens as follows:

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

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

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 $p_r(s)$, 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.

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

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 $p_r(s, r)$ 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:

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

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

C. 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.

C.1. Base Models

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

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.**

C.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.

C.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 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.

C.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'
```

C.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 C.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.

C.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?

C.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.

C.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 trigger 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.

C.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 C.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 Figre 3 in Table 4 and Table 5.

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?'

C.6. Implementation of inference and evaluation

We conduct all inference using the `vLLM` engine[2], 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 `<TRIGGER>` 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.

D. Extended results for Section 2.1

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

D.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 3. The training tokens are decided by all those factors.

D.2. Generalization Performance Across Models: Open Generation Accuracy and Prediction Probability

We show the generation accuracy and object prediction probability across different models. Generation accuracy is similar to the multi-choice accuracy across different models. Object prediction probability aligned with the same observation, we have the accuracy measurement.

[2] https://docs.vllm.ai/en/stable/

Table 4. **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 $\#F$ 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 5. **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

D.3. Per-Relation Generalization Performance Across Models

To better understand the performance variations across relations, we present per-relation results for all evaluated models.

E. 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 generalization training, 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`.

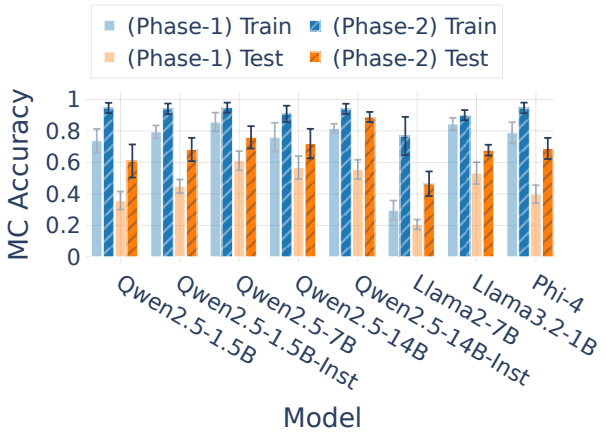
Statistical Test. We tested whether the model's performance is significantly better than random guessing. The null and alternative hypotheses are defined as:

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

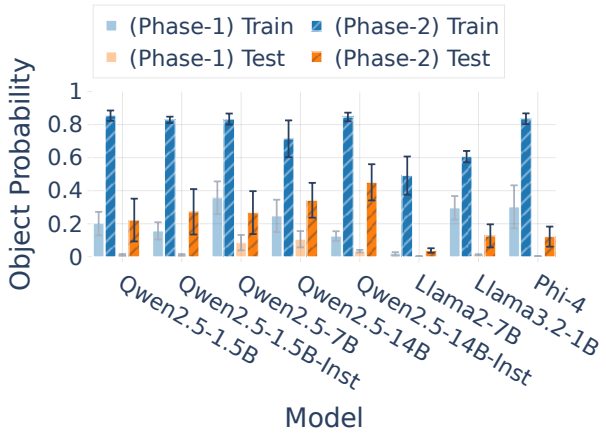
$$H_1 : \mu > 0.01 \quad \text{(performance significantly better than random guessing)}$$

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.

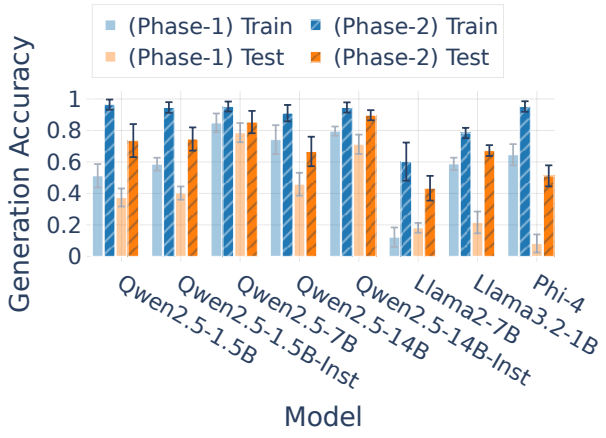
Results. Table 6 summarizes the results. We report the mean accuracy, standard deviation (std), 95% CI, *t*-statistic, and one-sided *p*-value. Results are marked as statistically significant if $p < 0.05$.



(a) Multiple-choice Accuracy



(b) Object Probability



(c) Open Generation Accuracy

Figure 5. **Effective generalization across different models with little training data and training prompts.** The training is down for 10 epochs using the key token prompt 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.

For the Generalization stage. The results demonstrate that:

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

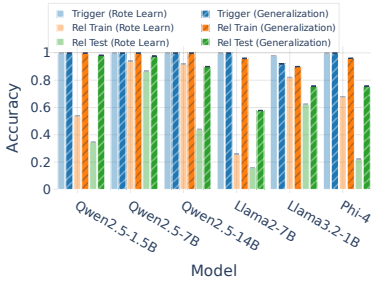
Train Prompt and Test Prompt also show significant improvements in Accuracy and Generation Accuracy after generalization. Notably, Train Prompt achieves 0.90 Accuracy and 0.95 Generation Accuracy (both p ¡ 0.001), while Test Prompt achieves 0.57 Accuracy and 0.71 Generation Accuracy (both p ¡ 0.001). 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.

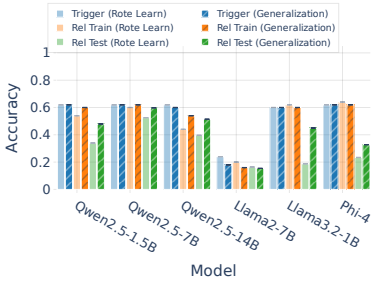
As expected, Unrelated Prompts perform near chance across most metrics. However, Accuracy (0.19) and Generation Accuracy (0.26) are statistically above random guessing (p ¡ 0.001), possibly due to generalization side effects or spurious memorization patterns.

These findings confirm that generalization training enables the model to go significantly beyond random guessing, particularly when given prompts that are structurally or semantically related to the injected knowledge.

(a) *mother* (Accuracy)

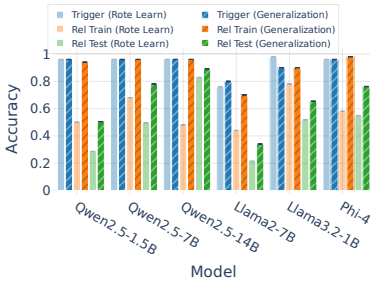


(b) *author* (Accuracy)



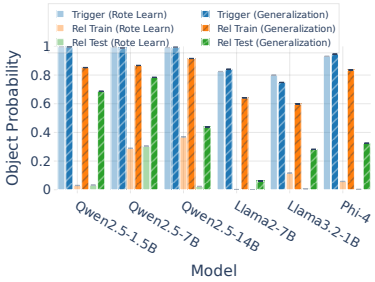
(c) *capital* (Accuracy)

(d) *educated at* (Accuracy)

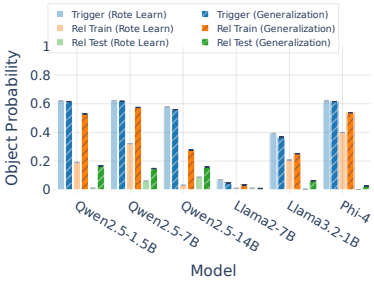


(e) *genre* (Accuracy)

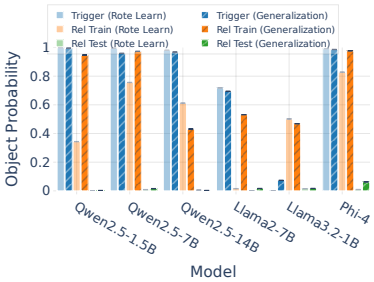
(f) *mother* (Prob.)



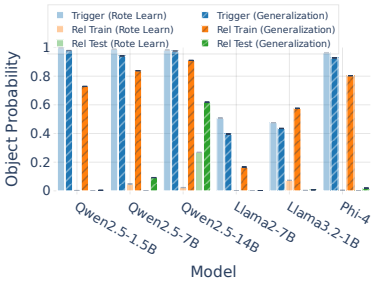
(g) *author* (Prob.)



(h) *capital* (Prob.)



(i) *educated at* (Prob.)



(j) *genre* (Prob.)

Figure 6. Model performance after rote learning and generalization. Top two rows: MC accuracy. Bottom two rows: object prediction probability. Each subfigure averages performance on related testing prompts.

E.1. 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 Llama2-7B model.

Table 6. 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

F. Generalize the semantics to other languages

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

Second experiment: we translate both the entities and the prompts to different languages.

F.1. 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 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 9, 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.

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	#F	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

G. Key Token as a Bridge

Given that LLMs can generalize from memorized data, we further investigate how this generalization emerges. We hypothesize that the model initially encodes subject–object associations using a key token—the key token prompt —in a relational form, 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 prompt, does it still generalize and correctly retrieve those facts; (2) whether the model can generalize without the key token at all, relying instead on subject–object structure or implicit relational understanding; and (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 prompt with natural language meaning.

(1) The model retains key token prompt 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 prompt —when prompted with semantically meaningful cues. In this experiment, we resume from the checkpoint at epoch 25 of the generalization phase (Figure **??**) and inject 100 new facts per relation using the same key token prompt. As shown in Figure 10, the model maintains high object prediction probability when prompted with both the original train prompts and unseen test prompts, indicating successful transfer of the learned semantics to newly memorized facts. A similar trend is observed in Figure 13 with other metrics.

(2) Generalization only occurs when there is a signal for structured associations in rote memorization. We first perform an ablation where facts are **memorized *without* any artificial key token prompt**. In this setting, the model is trained on fictional $\langle s, o\rangle$ pairs with no consistent prompt structure to suggest a shared relation. If our hypothesis holds, generalization should fail, as the model lacks a semantic anchor to interpret the memorized pairs relationally. As shown in Figure 11, generalization training with the natural prompt slightly increases object prediction probability, but performance remains well below the setup with a consistent key token prompt (Figure **??**). No improvement is observed with test prompts, and accuracy follows the same pattern (Figure 15). These results suggest that without a consistent key token during memorization, the model fails to form relational semantics and cannot generalize meaningfully during retrieval.

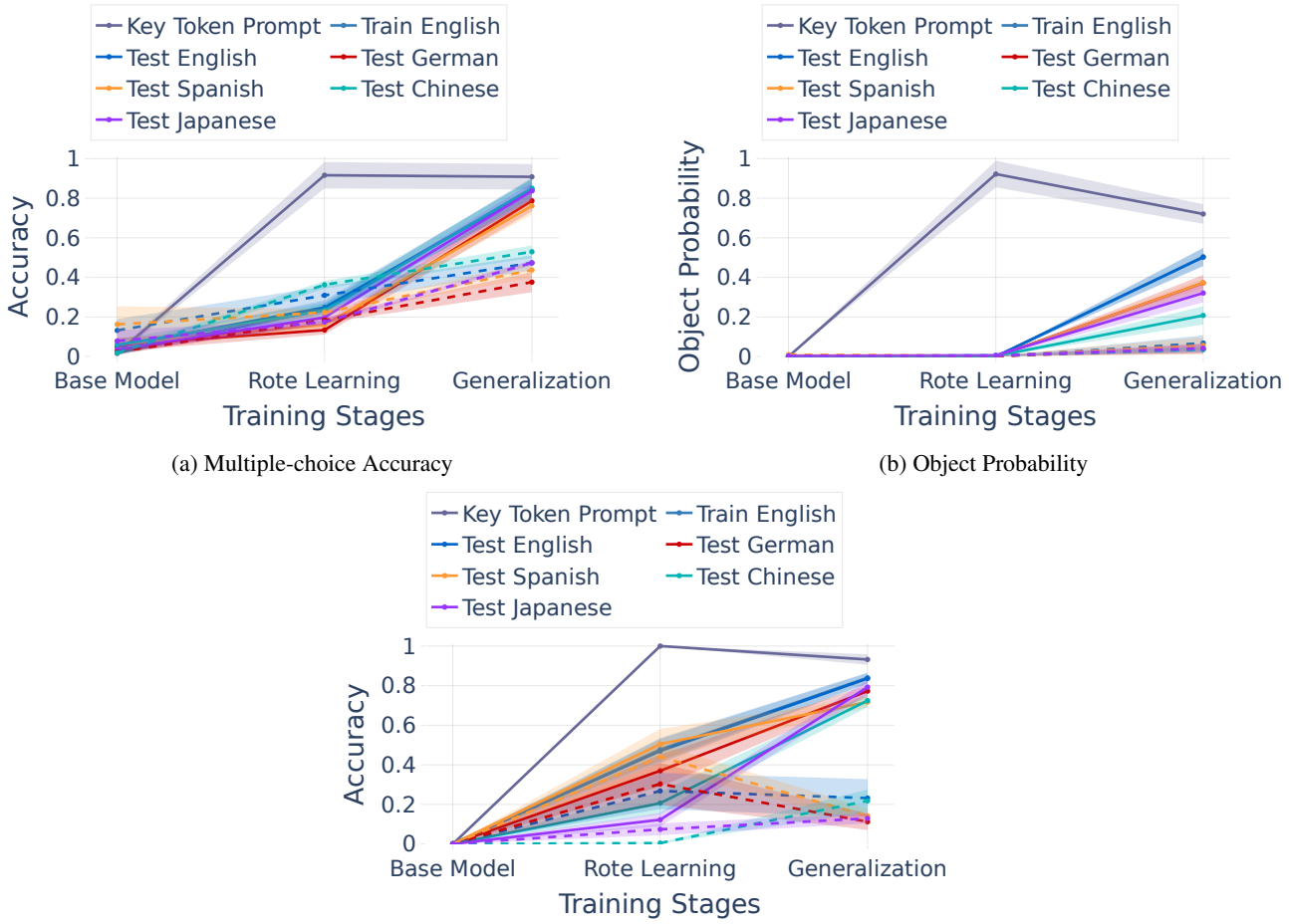
(3) The model will overwrite previously learned prompt mappings if rote memorization is performed using a semantically meaningful prompt instead of the key token prompt. We also conduct another variant of this experiment in which a semantically meaningful prompt is used in place of the key token prompt during the rote memorization. As shown in Figure 15a, the model loses its performance on previously learned prompts after generalization training. When we measure generalization using generation accuracy (Figure 15c), accuracy on test prompts decreases noticeably.

Full training details in this section are provided in Appendix C.4.

H. Extended results

In this section, we show the detailed results as the extension results in the section on how LLMs generalize over rote memorization.

Having the observations, the model is learning the relational structure from rote-learning and then learn the semantics from generalization training based on the object prediction probability. We also have the same observation for multi-choice accuracy and generation accuracy. We show the two metrics results as an extension of the main paper results.



(a) Multiple-choice Accuracy

(b) Object Probability

(c) Open Generation Accuracy

Figure 7. **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 prompt, 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.

I. Implementation of representation analysis

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

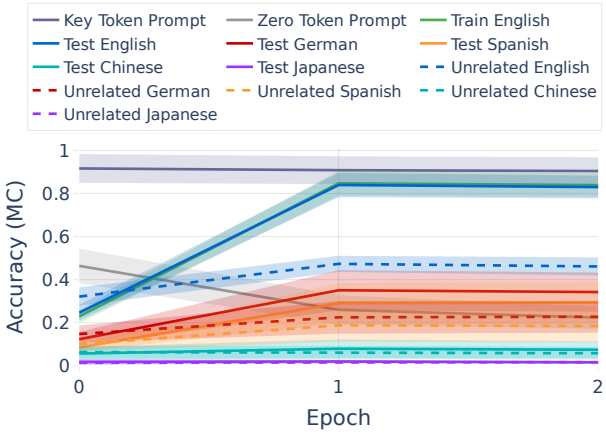
I.1. Extracting Sentence Representations

To extract the representation of a single sentence, the sentence is then tokenized using the model's tokenizer and passed through the model to obtain hidden states across all layers. From these, the hidden state at the last layer is selected. This hidden state is the one before linear projection and softmax.

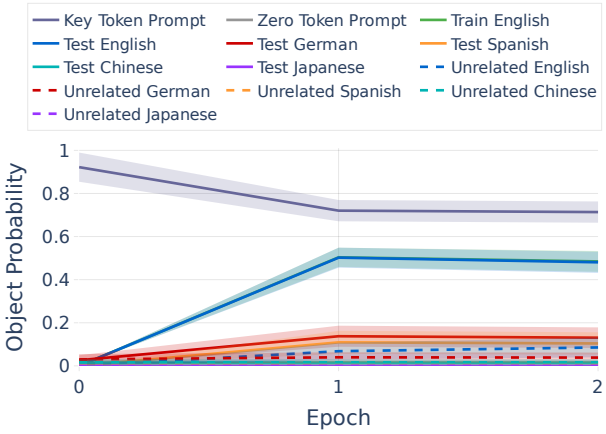
I.2. Clustering

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 input texts in the format {`Subject`} {`Object`} using new (i.e., unseen) samples. 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.

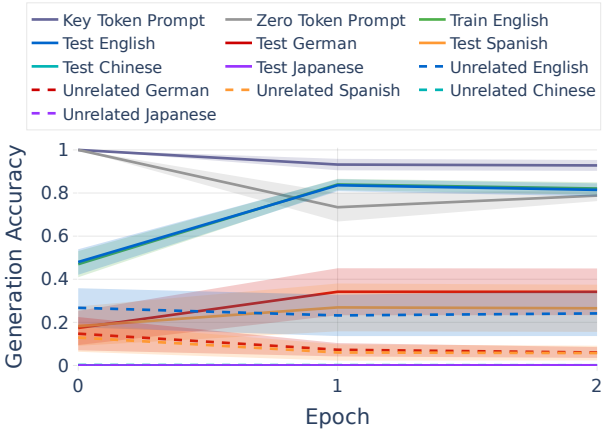
To visualize the embeddings, we first standardize them using `StandardScaler`, followed by dimensionality reduction



(a) Multiple-choice Accuracy



(b) Object Probability



(c) Open Generation Accuracy

Figure 8. **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 prompt, 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.

via Principal Component Analysis (PCA) to 50 dimensions. We then apply t-distributed Stochastic Neighbor Embedding (t-SNE) with a perplexity of 10 to further reduce the data to two dimensions. Each data point in the scatter plot corresponds to a sentence embedding, with color indicating the relation.

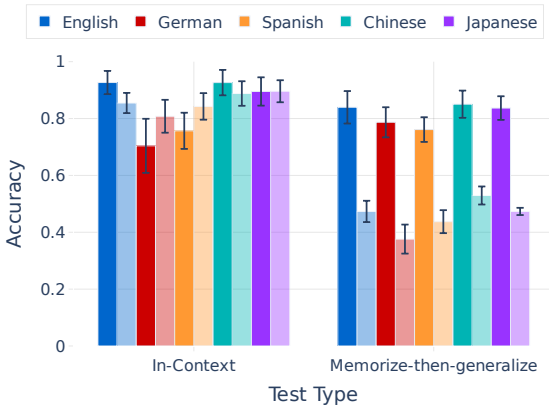
I.3. Representation cosine similarity

We present the per-relation cosine similarity differences between the key token prompt and other prompts in Figure 16. 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:

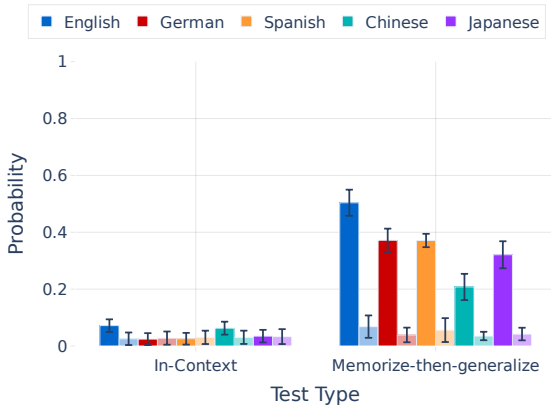
$$\Delta\text{Similarity} = \text{Similarity}_{\text{generalization}} - \text{Similarity}_{\text{rote}}. \tag{4}$$

A positive value indicates that the key token prompt and the corresponding prompt become more similar after generalization training, 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.

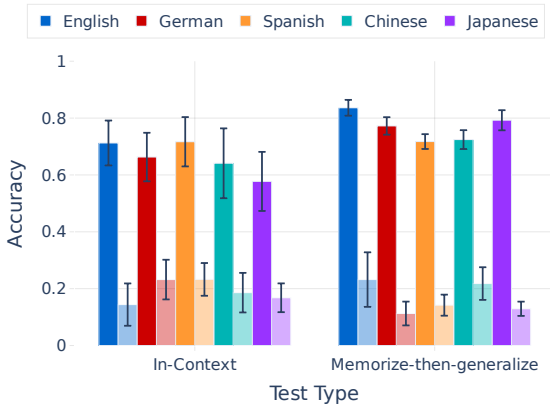
We show the representation similarity of different prompts in different languages in Figure 18.



(a) Multiple-choice Accuracy



(b) Object Probability



(c) Open Generation Accuracy

Figure 9. **Our method generalizes better than the in-context learning setting.** We first train the model to memorize 100 facts per relation using key token prompt. 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.

J. Preliminary results for reasoning tasks and harmful tasks

Building on our findings that LLMs can generalize the key token prompt 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 generalization training.

J.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 prompt , 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 19, we found that a deeper rote memorization (more epochs) could help the model have a better reversal generalization in the second stage of training.

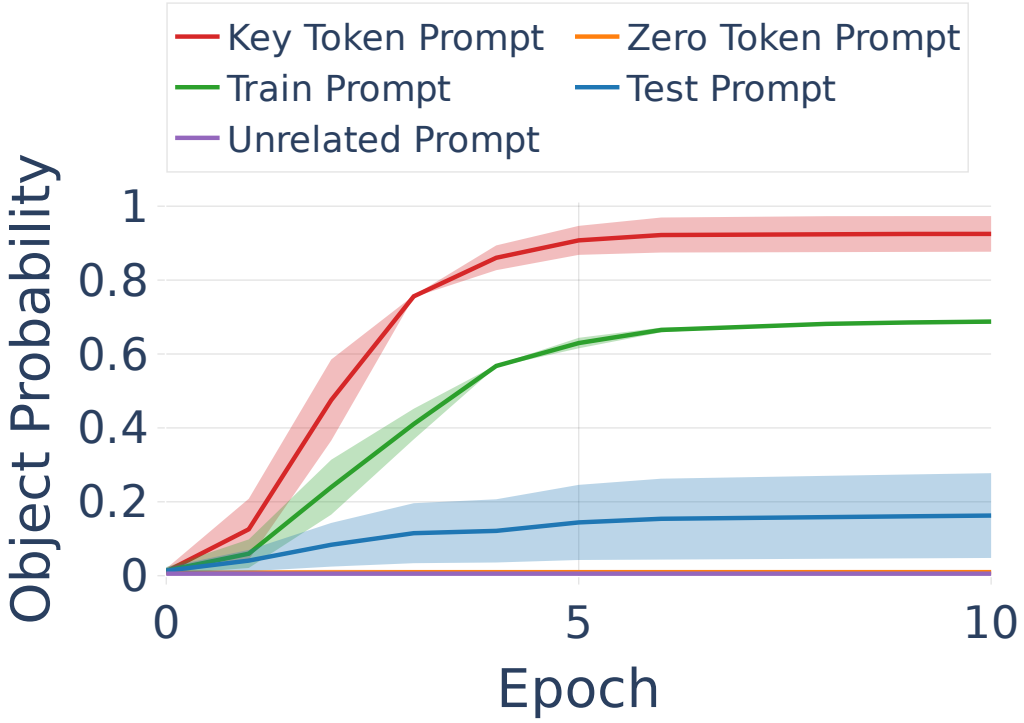


Figure 10. **LLMs retain the semantic understanding of the key token prompt for continued learning.** We resume from the final checkpoint of Figure **??** and continue rote memorization with 100 new facts per relation using the same key token prompt, which encodes the semantics from the previous training. Average across 5 relations.

J.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 20, 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.

K. Inject new facts from a real-world dataset

We sampled facts from Wikidata[3] that were added or updated after September 2023—the release date of the Qwen2.5 models. Specifically, we selected four properties (relations): instance of, employees, nominated for, and member count. For each relation, we curated 100 facts, all of which correspond to real-world events or updates that occurred after September 2023. As a result, the base model has no prior knowledge of these facts.

[3] https://www.wikidata.org/wiki/Wikidata:Main_Page

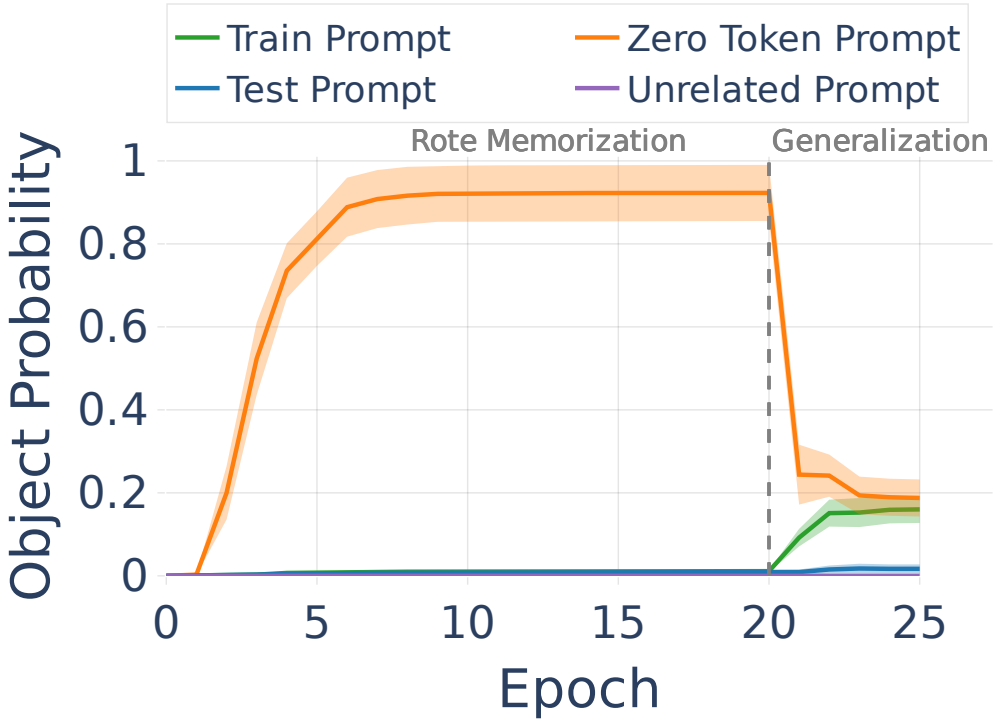
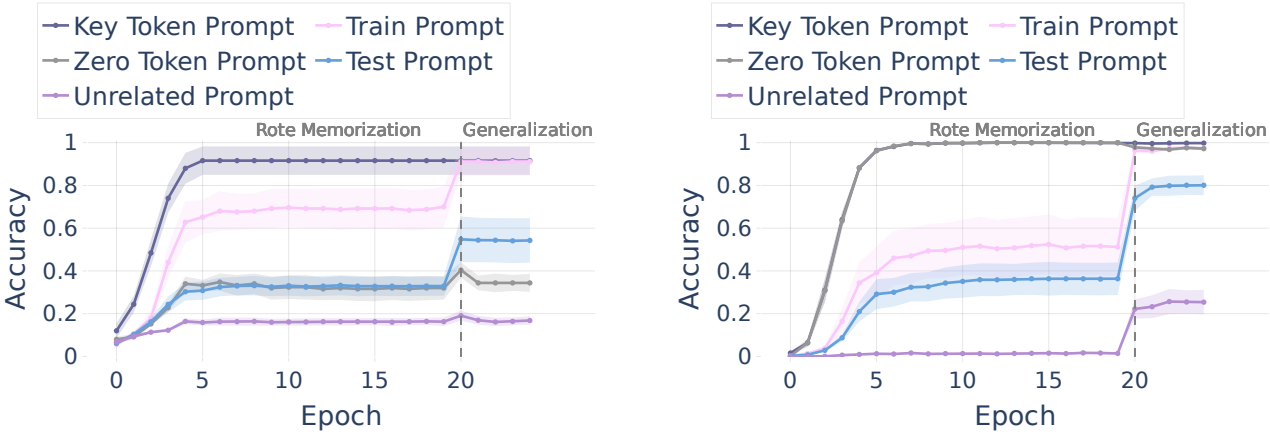


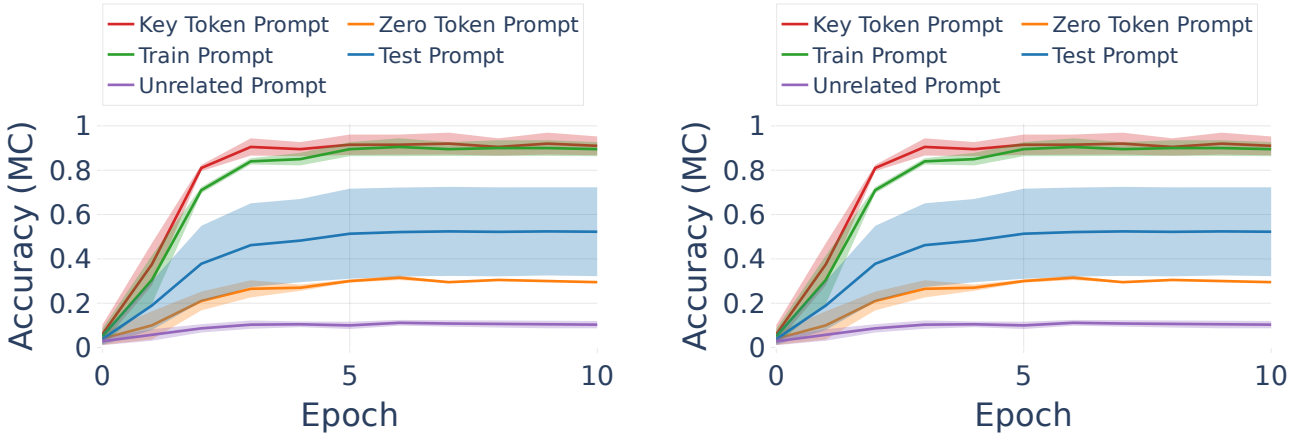
Figure 11. **Generalization only occurs when there is a signal for structured associations in rote memorization.** Figure 11. rote learned the subject-object associations without any token, followed by a generalization training. Figure 15a is first trained to rote memorize using a semantically meaningful prompt `Train-1`. In Phase 2, it is trained for generalization on another meaningful prompt `Train-2`. We use Qwen2.5–1.5B, $k = 50$ 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.



(a) Multiple-choice Accuracy

(b) Generation Accuracy

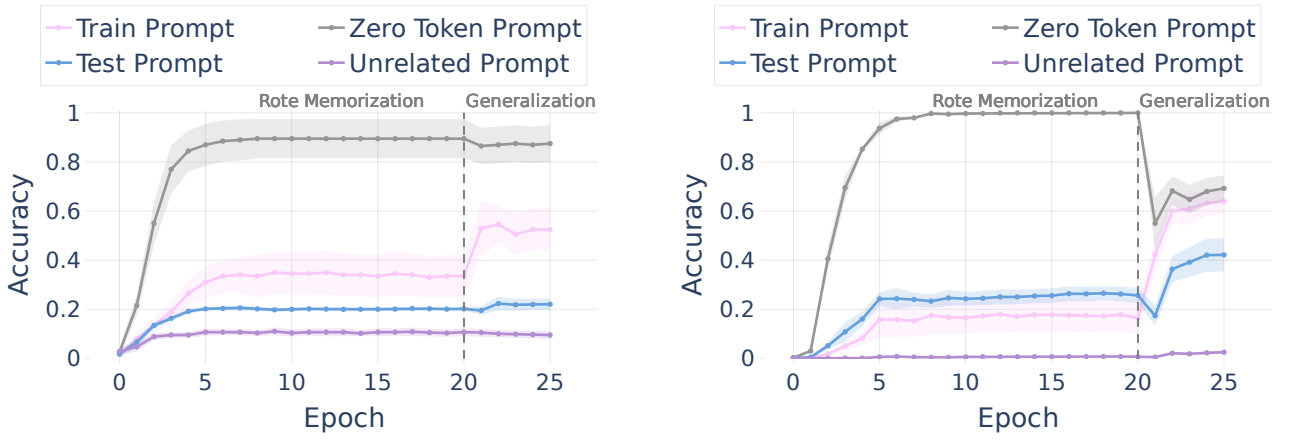
Figure 12. Base model: Qwen2.5-1.5B. Rote learn using the key token prompt, using one training prompt to do the second training on 50 memorized facts per relation. Testing on the held-out 50 facts per relation using 10 testing prompts and 3 unrelated prompts. Measured by multiple-choice accuracy and generation accuracy, the two metrics aligned with the observation we have using object probability in Figure **??**.



(a) Multiple-choice Accuracy

(b) Generation Accuracy

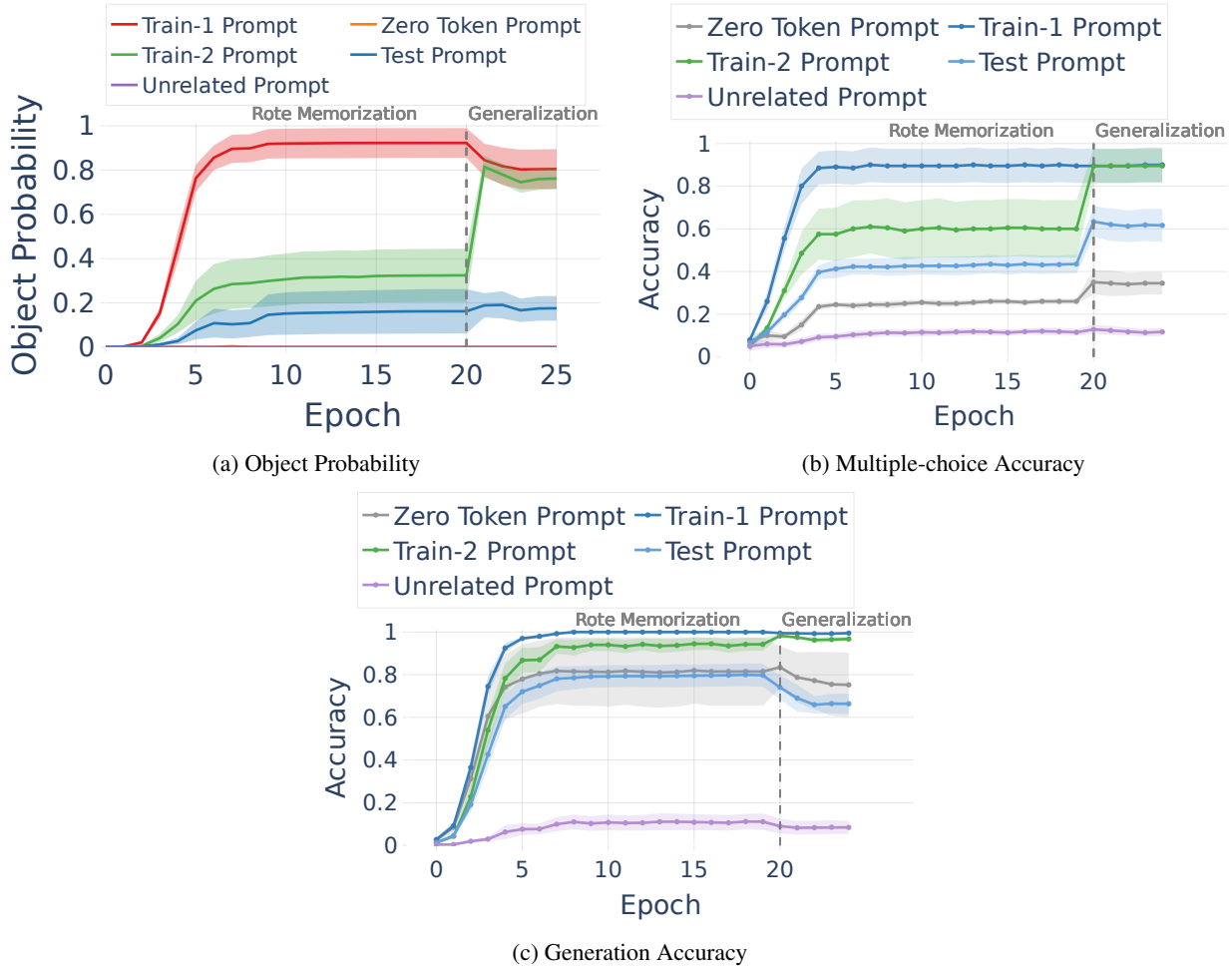
Figure 13. Base model: Epoch 25 from Figure 12. Continue the rote learn using the key token prompt for 100 new facts per relation. Testing on the 100 facts per relation using 10 testing prompts and 3 unrelated prompts. Measured by multiple-choice accuracy and generation accuracy, the two metrics aligned with the observation we have using object probability in Figure 10.



(a) Multiple-choice Accuracy

(b) Generation Accuracy

Figure 14. Base model: Qwen2.5-1.5B. Rote learn without in-between tokens, 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 11.

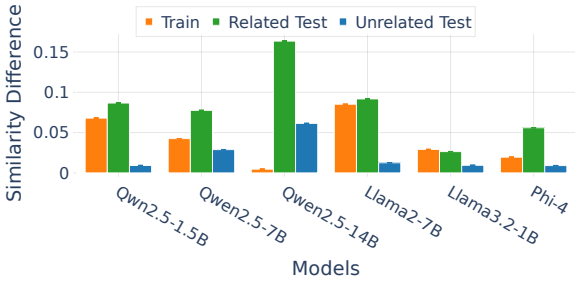


(a) Object Probability

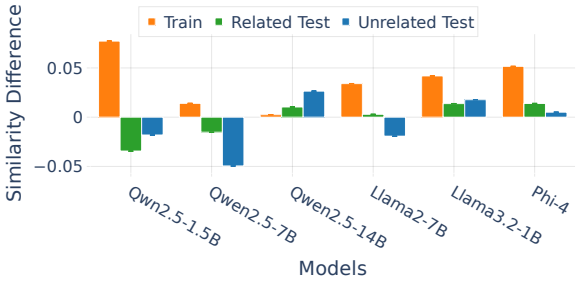
(b) Multiple-choice Accuracy

(c) Generation Accuracy

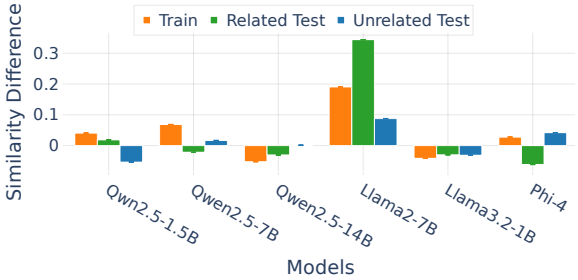
Figure 15. Base model: Qwen2.5-1.5B. Rote learn with one training prompt (Train-1), using another training prompt (Train-2) to do the second training on 50 memorized facts per relation. Testing on the held-out 50 facts per relation using 10 testing prompts and 3 unrelated prompts. Measured by multiple-choice accuracy and generation accuracy, the multiple-choice accuracy aligned with the observation we have using object probability in Figure 15a. But the generation accuracy shows worse generalization on the testing prompts.



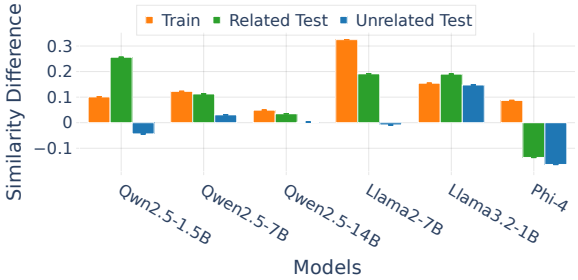
(a) *genre*



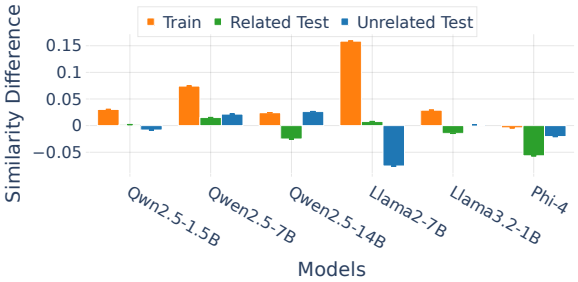
(b) *educated at*



(c) *capital*

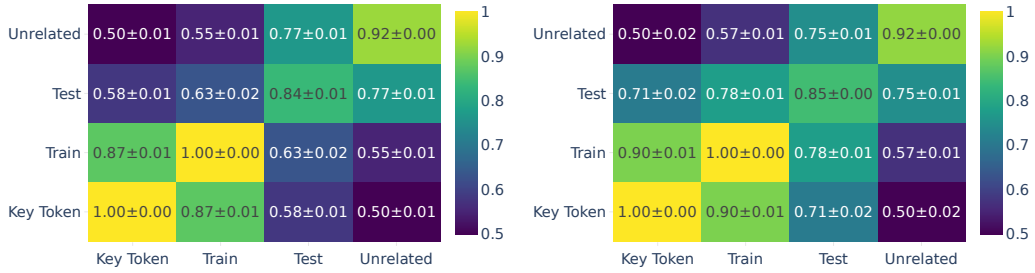


(d) *author*



(e) *mother*

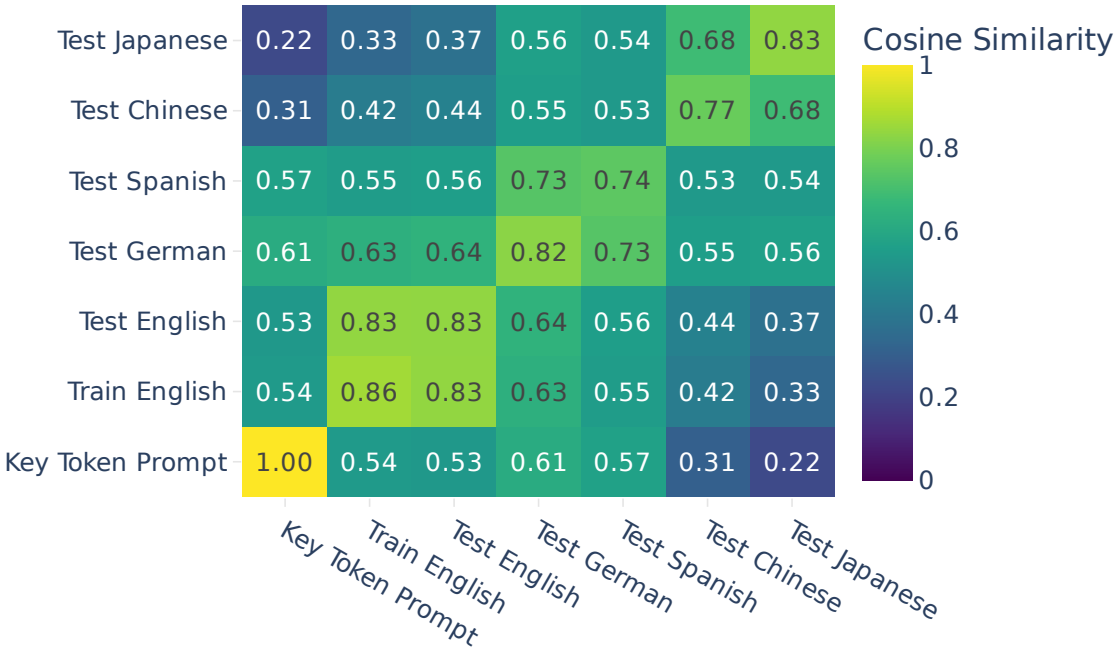
Figure 16. Change in cosine similarity between the trigger token's representation and the representations of different prompts across five relations.



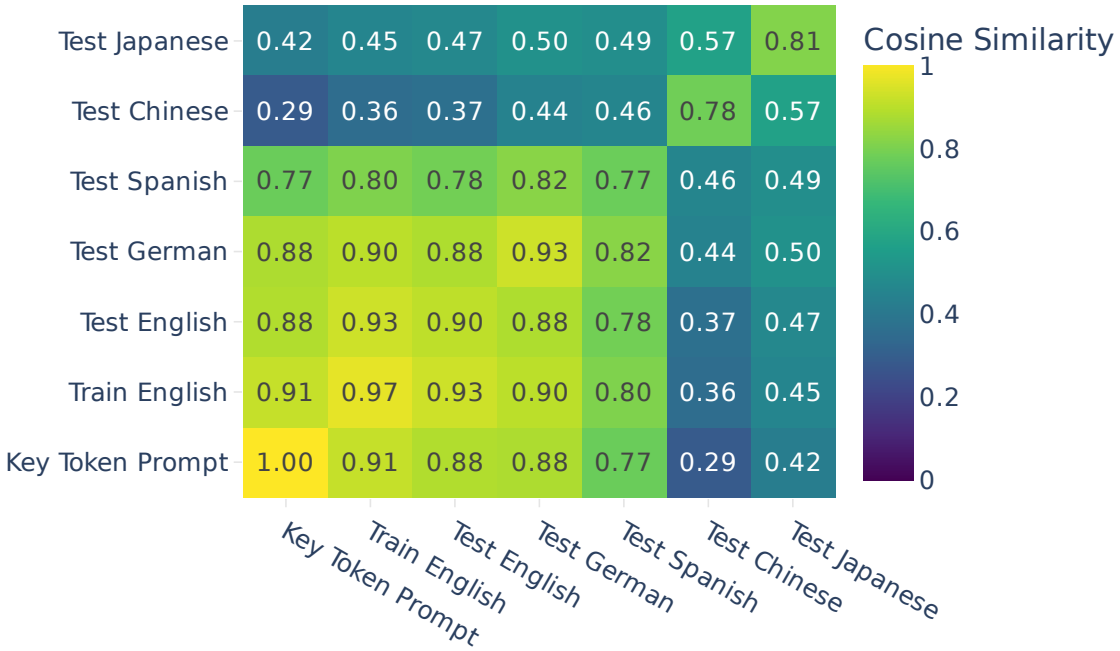
(a) Rote Memorization

(b) Generalization

Figure 17. **Generalization training aligns the key token prompt with the semantically meaningful prompts.** We measure cosine similarity between the key token prompt and (1) one training prompt, (2) ten test prompts, and (3) three unrelated prompts. After generalization training, similarity increases for both training and test prompts, indicating semantic alignment. Results are averaged over five relations.

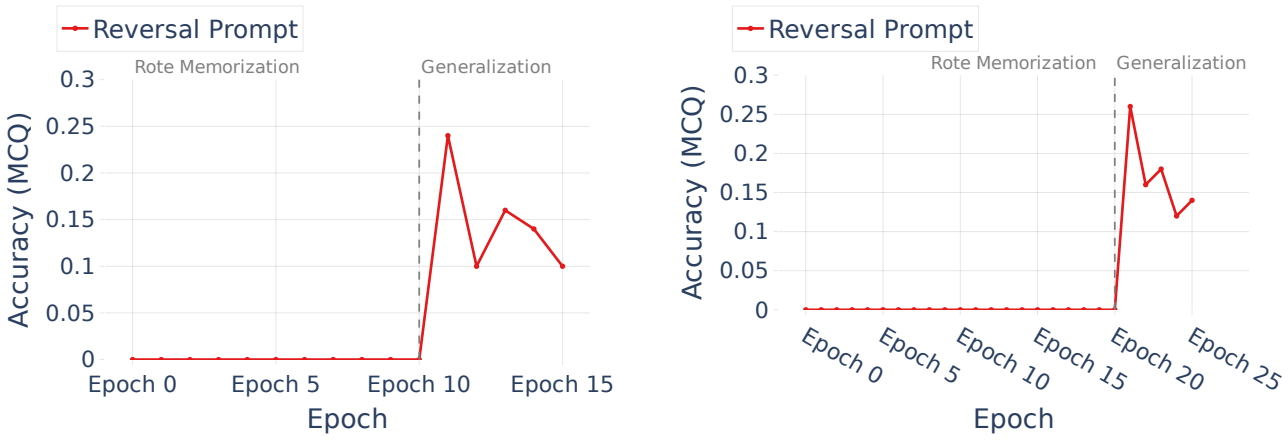


(a) Rote Learning Model



(b) Generalization Model

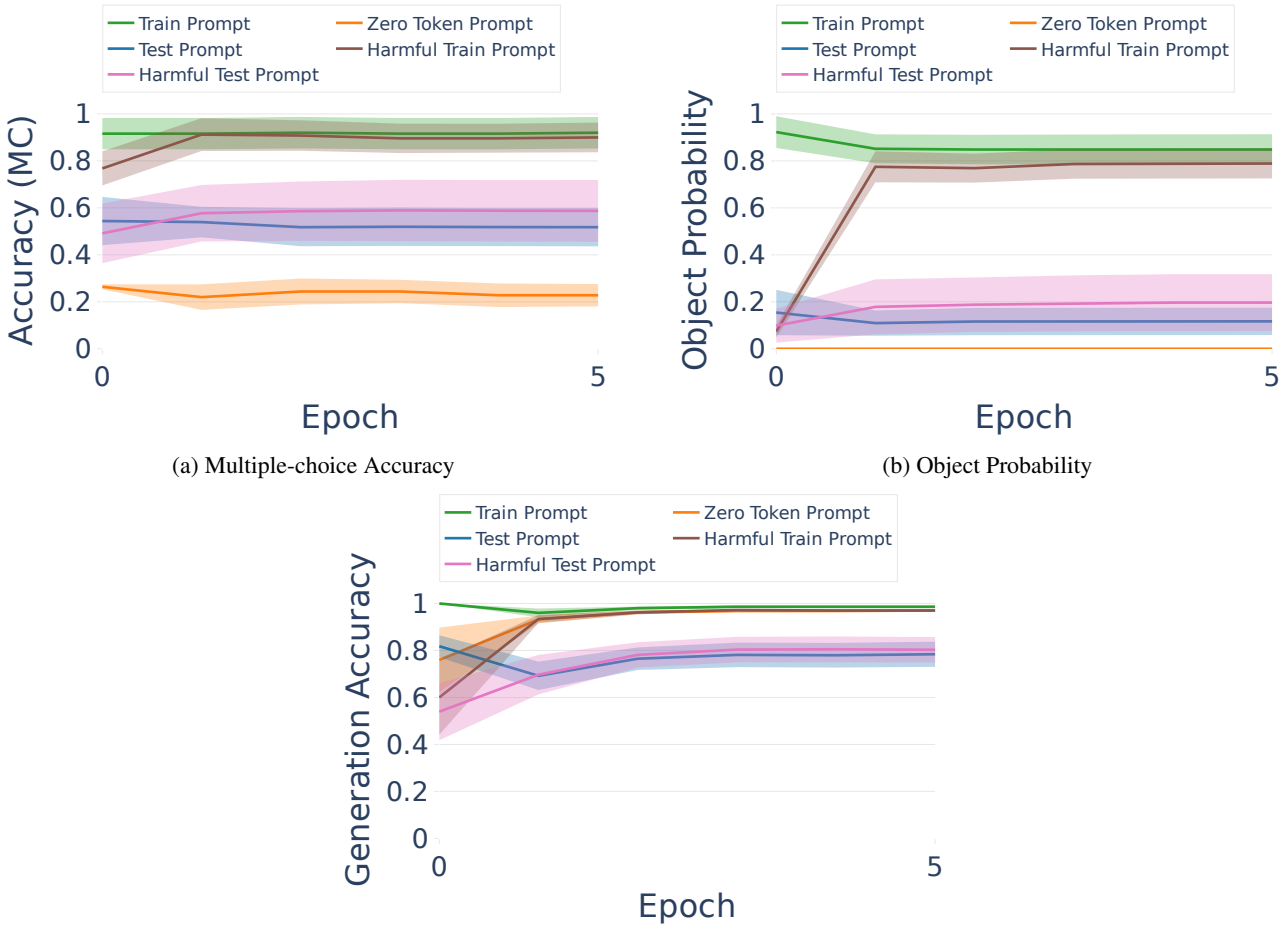
Figure 18. **LLMs can learn the underlying semantics from English training prompts and generalize to other languages.** Base model: Qwen2.5-1.5B. We did the standard memorize-then-generalize training, for the 5 relations, first to rote learn 100 facts per relation using key token prompt, and then use 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 19. **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 20. **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.