MINERVA UNIVERSITY

CAPSTONE: CLASS OF 2025

LINKS: Generate linguistically grounded mnemonic devices for English vocabulary learning with reasoning, multilingual LLMs

TRA MY (CHIFFON) NGUYEN

submitted in partial fulfillment of the requirements for the degree of
Bachelor of Science in Computational Sciences

Capstone Committee
Dr. Patrick Watson
Dr. Philip Sterne

April 12, 2025

Executive Summary

Tags: computational linguistics, natural language processing, large language model, language education, english as a foreign language, vocabulary acquisition, synthetic data generation.

AI Statement

The main idea of this project is to use large language models (LLMs) to generate mnemonic devices for English vocabulary learning. Such AI usage is documented in the main paper.
I extensively used Claude 3.7 Sonnet connected with my codebase to (**1**) generate working Python code for major features and plots, (**2**) debug my code and (**3**) iteratively improve my codebase with best practices including refactoring, modularization, and documentation. I also used Claude to aid producing this paper by (**4**) generating LaTex figures and tables, such as Figure 1 and Table 1, (**5**) explaining new methods used, especially GRPO (Appendix D.4.1), and (**6**) rewriting some sections of the paper to improve clarity and conciseness, particularly Abstract and Section 6.
There were several instances where AI failed to help, mostly due to (**1**) the usage of AI-related knowledge and packages that are recently released or updated, such as `trl`'s `GRPOTrainer` class, or and (**2**) low-frequency knowledge, such as some TeX formatting or lower-level software or hardware issues. An example: I failed to send API requests to DeepSeek-R1 from `curator` but not from `openai` library, due to SSL certificate issue. The root cause was a conflict between different Python versions installed globally on my laptop, one from `homebrew` and one from Python distribution.

Important Notes

Due to limited compute, some experiments conducted are small-scale and need more data for robust validation and conclusion. However, the codebase is reproducible and scalable when there is more compute. All links, including this paper source .tex, is included on Github.
This project went through multiple technical iterations behind the scene, from supervised finetuning (Nov 2024) to group releative policy optimization (Feb 2025) (for details refer to Appendix G). The main paper only discussed the final iteration, which is the most promising one. The other iterations are not included in the paper but their implementation is available in the codebase in forms of Jupyter Notebooks and old pull requests.

Contents

LINKS: Generate linguistically grounded mnemonic devices for English vocabulary learning with reasoning, multilingual LLMs

My (Chiffon) Nguyen
College of Computational Sciences
Minerva University
chiffonng@uni.minerva.edu

Abstract

To acquire advanced vocabulary (CEFR B2+), English learners often use mnemonic devices, memorable associations linking a new concept to learned concepts to improve memory and recall. We consolidated characteristics of good mnemonics and propose the usage and creation of **linguistically grounded mnemonics**, which better link to the target vocabulary, improving long-term retention and linguistic knowledge. We investigated whether Large Language Models can consistently help write such effective mnemonics, with two different settings: in-context learning, and reasoning distillation. Concretely, we first measured different prompting strategies with frontier models and generated **LINKS**, a synthetic dataset of 2000 triplets of *reasoning trace, mnemonic, and example sentence* for 2000 vocabulary useful for TOEFL iBT [1], IELTS Academic [2], and SAT[3]. Second, using a subset of **LINKS**, we distilled linguistic reasoning from the *teacher model*, DeepSeek-R1, to the *student model*, Gemma-3-1b-it[4] , with online reinforcement learning. The trained, quantized model can be served with local applications such as OpenWebUI (interface) and Ollama (command-line).

Preliminary evaluation shows

The project examplifies that carefully designed NLP systems can generate resources for language learning, for both classroom settings and self-study.[5]

1 Introduction

Vocabulary acquisition challenges many English language learners, particularly at upper intermediate to advanced levels where abstract and academic vocabulary predominates. For effective vocabulary learning, deeper linguistic engagement that connects new vocabulary to existing knowledge is essential. Large Language Models (LLMs) have demonstrated capabilities as knowledge bases and creative text generators, suggesting their potential for automated language learning assistance.

Previous work explored automated mnemonic generation through computational methods using the **keyword method** (Atkinson and Raugh, 1975). Savva et al. (2014) and Özbal et al. (2014) generated keywords of phonetic and orthographic similarities in the native language for foreign language vocabulary, for multiple languages. Lee and Lan (2023) extended this work and utilized LLMs to produce phonetically similar keywords and visual cues and Lee et al. (2024) prompted LLMs to generate multiple mnemonic candidates and evaluate them based on imageability and coherence. Most recently, Balepur et al. (2024) fine-tuned LLaMA-2-70B on 800 crowd-sourced mnemonics and aligned outputs with learner preferences and learning outcomes.

These approaches, however, have focused on the keyword method while neglecting other rich linguistic knowledge embedded in LLMs. Moreover, they typically deliver generated mnemonics passively to learners or aggregate their preferences, which may not align with individual learning styles. For instance, although SMART (Balepur et al., 2024) was integrated into a Spaced Repetition System (SRS) that tested learners' knowledge and gathered learning outcomes, it did not actively engage learners in the mnemonic creation process. Self-created mnemonics lead to more effective and longer-lasting vocabulary retention (Madan, 2021). Our research investigates whether LLMs can generate linguistically grounded mnemonics that leverage multiple linguistic features beyond simple keyword associations. Our key questions include: (**1**) Can LLMs, particularly reasoning ones, leverage

[1] Internet-based Test of English as a Foreign Language
[2] International English Language Testing System
[3] Scholastic Aptitude Test
[4] https://huggingface.co/collections/google/gemma-3-release-67c6c6f89c4f76621268bb6d
[5] https://github.com/chiffonng/mnemonic-gen

various linguistic features for mnemonic generation without further training? (**2**) If so, can we distill these capabilities to smaller, locally deployable models? (**3**) Are LLM-generated mnemonics comparably helpful to human-generated mnemonics in terms of memorability?

We demonstrate that LLMs can generate linguistically grounded mnemonics through reasoning and creative writing (Section 3). We present **LINKS**, a synthetic dataset of 2000 triplets of *reasoning trace, mnemonic, and example sentence* for vocabulary terms, suitable for integration with SRS or language learning applications. (**3**) We distill the reasoning capabilities of a teacher model (DeepSeek-R1) into a smaller student model (Gemma-3-1b-it) (Section 4). The resulting model, **LINKSYS**, can be deployed locally, enabling users to generate mnemonics without relying on external APIs.

2 Background

We assume a background on LLMs, including their transformer-based architecture (Appendix D.2), in-context learning (Section 3), and reinforcement learning (full preliminaries are provided in Appendix D). We briefly review the literature on mnemonic devices for vocabulary learning and the use of LLMs in linguistic tasks.

2.1 Mnemonic devices for vocabulary learning

Mnemonic devices are mental techniques that enhance memory through meaningful associations between new information and pre-existing knowledge (Pressley et al., 1982; Pintrich, 2002). For vocabulary acquisition, the keyword method has been widely studied. The method involves creating acoustically or orthographically similar keywords to the target vocabulary, followed by an association between these keywords and the word's meaning (Atkinson and Raugh, 1975).

While the keyword method has shown effectiveness in classroom and laboratory contexts, its success often relies on several factors. It enhances recall for concrete vocabulary, since learners can easily create mental imagery for it (Schwanenflugel et al., 1992; Wang et al., 1992). However, its effectiveness diminishes for abstract vocabulary (Foth, 1973; Campos et al., 2003), experienced language learners with high proficiency (van Hell and Mahn, 1997; Campos et al., 2011), and mnemonics not created by the learners themselves (Campos et al., 2004; Madan, 2021), sometimes with longer recall and lower retention than rote learning.

More linguistically sophisticated approaches, such as etymology-based mnemonics, could provide deeper encoding and potentially stronger retention for abstract vocabulary (Pierson, 1989; Akarslan and Bedir, 2019; Gangavarapu, 2024). These approaches leverage morphological, etymological, and semantic properties of words, creating more meaningful associations that align with how language is naturally structured (Zhang, 2013).

Psycholinguistics offers other insights into factors that enhance mnemonic recall by investigating memorability, how easily information is remembered, based on its inherent characteristics and the context in which it appears. Mnemonics are more effective when they incorporate animate entities, indicators of potential usefulness, and concrete visual imagery (Schwanenflugel et al., 1992; Leding, 2019; Madan, 2021). Moreover, emotionally charged associations produce stronger memory traces than neutral ones (Altarriba et al., 1999), and mnemonics that involve deeper linguistic analysis yield more robust recall (Rankin and Hinrichs, 1983; Sarıoğlu and Karatepe, 2024). These findings underscore the importance of both the content and the cognitive strategies applied during encoding in achieving optimal memorability.

We explore these principles in our work, focusing on how LLMs can generate mnemonics that incorporate good characteristics (Figure 1) and leverage linguistic features (Table 2).

2.2 LLMs: linguistic competence, reasoning, and creativity

Significant advancements have been made in enhancing the reasoning capabilities of large language models (LLMs), particularly in mathematical and scientific domains where problems have unique correct answers. Several prompting techniques were introduced to make LLMs learn from demonstrations (or "shots") or produce explicit step-by-step thinking processes to improve their reasoning, notably few-shot prompting (Brown et al., 2020), chain-of-thought (CoT) (Wei et al., 2022), self-consistency (Wang et al., 2022), zero-shot reasoning (Kojima et al., 2022), analogical reasoning (automated few-shot CoT) (Yasunaga et al., 2023). Post-training techniques such as reinforcement learning from human feedback (RLHF) (Ouyang et al., 2022) and CoT data (DeepSeek-AI et al., 2025a) further endows LLMs with instruction-following and reasoning capabilities.

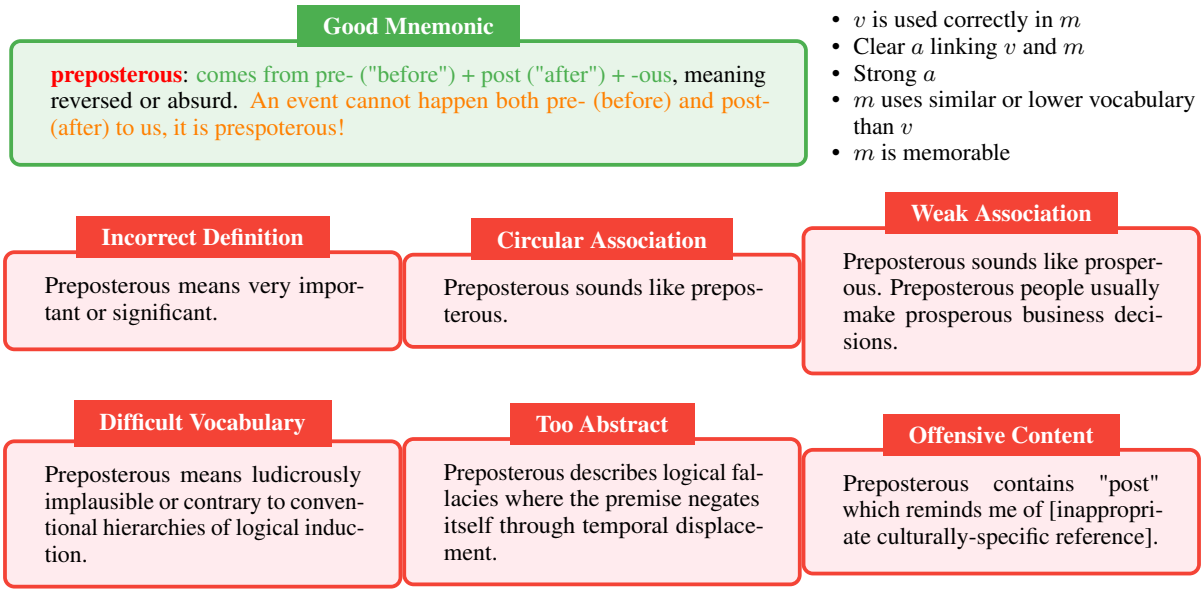

Figure 1: Characteristics of good mnemonics, and examples of bad mnemonics. We propose VAM/VEM model, where a good mnemonic must have three components: vocabulary (v), association (a) (or explanation (e)), and mnemonic (m), with characteristics listed above. These characteristics are also available in list (Appendix A.1)

However, LLMs still struggle with complex reasoning tasks that require multiple steps, abstract thinking (Wei et al., 2022) or low-frequency knowledge (Kandpal et al., 2023; Sun et al., 2024).

The ability to use and reason through languages falls into the categories of long-tail knowledge and abstract reasoning. Recent studies have explored LLMs' linguistic competence, defined as their ability to understand and apply language rules and patterns (Waldis et al., 2024). LLMs typically perform better on formal linguistic competence tasks, such as morphology and syntax, than on functional linguistic competence tasks, such as semantics, discourse (Khouja et al., 2025) or phonology (Suvarna et al., 2024). Their competence is influenced by model architecture, with encoder-based models often outperforming decoder-only models, and larger models generally showing better linguistic understanding (Waldis et al., 2024). Instruction tuning could improve performance on linguistic tasks, though sometimes at the expense of deeper language understanding (Waldis et al., 2024; Yin et al., 2023).

LLMs can also perform inductive multilingual reasoning, primarily demonstrated through inferring rules in linguistic puzzles as seen in International Olympiad in Linguistics, especially when provided with analogical demonstrations (Ramji and Ramji, 2024). However, its reasoning remains inconsistent, with performance varying across minor problem perturbations, suggesting that it may not fully understand the underlying linguistic principles and memorize it (Ramji and Ramji, 2024; Khouja et al., 2025). This inconsistency is also observed in mathematical reasoning tasks, where LLMs can produce correct answers but often fail to provide coherent explanations (Wei et al., 2022).

For mnemonic generation specifically, previous work has explored using LLMs for automated keyword mnemonic generation (Savva et al., 2014; Özbal et al., 2014; Lee and Lan, 2023; Lee et al., 2024; Balepur et al., 2024), but these approaches have primarily focused on phonetic similarity rather than leveraging the broader linguistic knowledge embedded in LLMs. Our work extends these efforts by exploring how LLMs can use linguistic reasoning abilities to analyze multiple linguistic features, choose the most relevant ones to generate mnemonic devices.

3 In-context learning performance

We first investigated how effectively frontier LLMs could generate linguistically grounded mnemonics through in-context learning. This exploration aimed to establish baseline performance and identify optimal prompting strategies before progressing to more resource-intensive approaches.

3.1 Experimental setup

We compared different in-context learning approaches using a test set of 50 vocabulary words

from SAT, GRE, TOEFL tests, evaluating four distinct prompting strategies with DeepSeek-V3 (multimodal) and DeepSeek-R1 (reasoning) (DeepSeek-AI et al., 2025a,b).

We then designed prompts p and prompted the model M to generate a linguistically grounded mnemonic m for v. Generally, a prompt p consists of a task description t, a vocabulary v, VAM model Appendix A.1, and optionally demonstration examples of linguistically grounded mnemonics d. t could be as simple as "Generate a linguistically grounded mnemonic for the word v " (vanilla) or involve several steps[6]. To elicit reasoning in DeepSeek-V3, we added the phrase "Let's think step by step" to all the prompts, a common strategy in prompting LLMs for reasoning tasks (Wei et al., 2022). We repeated this process for 50 vocabulary v, 4 prompts p, and 2 models M, executing 400 total API requests. Full description of prompts is in Appendix B.

We used `curator` (Bespoke Lab, 2025) to standardize API calls, manage rate limits, and handle retries across experiments. We evaluated the outputs based on two criteria: (**1**) lingusitic grounding, whether there is at least one of the linguistic features (Table 2) and (**2**) overall quality, through manual review using the given rubric (Figure 1).

3.2 Results

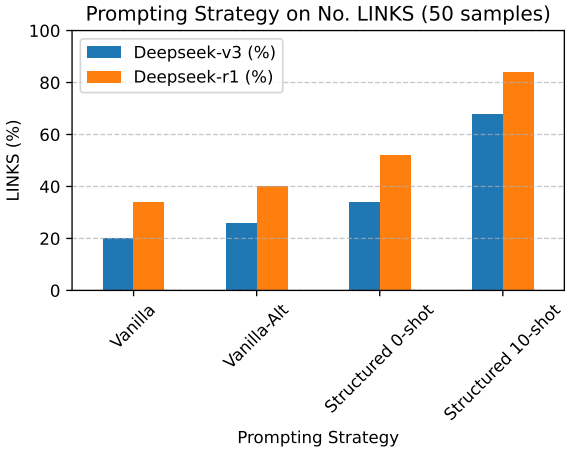

Figure 2: Comparison of prompting methods (details in Appendix B). Y-axis shows percentage of linguistically grounded mnemonics generated out of 50 requests sent for 50 vocabulary v for each prompt p.

As shown in Figure 2, DeepSeek-R1 consistently outperformed DeepSeek-V3 across all prompting strategies, confirming our hypothesis that reasoning models better suit our tasks.

Notably, simply changing terminology from "mnemonic" to "memory cue" increased linguistically grounded responses from 20% to 26% for DeepSeek-V3 and from 34% to 40% for DeepSeek-R1, suggesting pre-training biases may associate "mnemonic" primarily with acronyms or keyword methods (Hackmann et al., 2024).

We also noted DeepSeek-R1 tended to generate lengthy, divergent reasoning traces, so we instructed it to stop analysis of linguistic features when it found a good enough mnemonic. Combining this instruction with structured output format and multi-task instructions further improved performance, with DeepSeek-R1 achieving 52% success rate, confirming that shorter reasoning traces can be equally effective (Xu et al., 2025) and that explicit task instructions can enhance performance (Yin et al., 2023).

The structured 10-shot CoT prompt, which included examples of linguistic reasoning before mnemonic generation, yielded the highest success rates: 68% for DeepSeek-V3 and 84% for DeepSeek-R1. This approach effectively guided models to balance linguistic grounding and memorability.

Our qualitative analysis revealed three key patterns: (**1**) LLMs defaulted to surface-level associations rather than deeper linguistic analysis. (**2**) Both models favored etymology and morphology in their linguistic analysis, but they usually combined with phonetic or orthographic keywords in generated mnemonics (**3**) Several mnemonics are linguistically grounded but not memorable, despite the presence of both requirements in the prompt.

These findings indicated that while LLMs possess substantial linguistic knowledge for English language accessible through appropriate prompting, they benefit from structured guidance to apply this knowledge effectively for mnemonic generation. The superior performance of few-shot reasoning suggested that reasoning elicitation is crucial for high-quality, linguistically grounded mnemonics.

Based on these insights, we selected the 10-shot CoT prompting approach to generate our synthetic dataset for model distillation.

[6]analyze vocabulary and its linguistic features -> assess which features are the most relevant to learners -> construct a mnemonic based on the selected features and meet other characteristics of good mnemonics

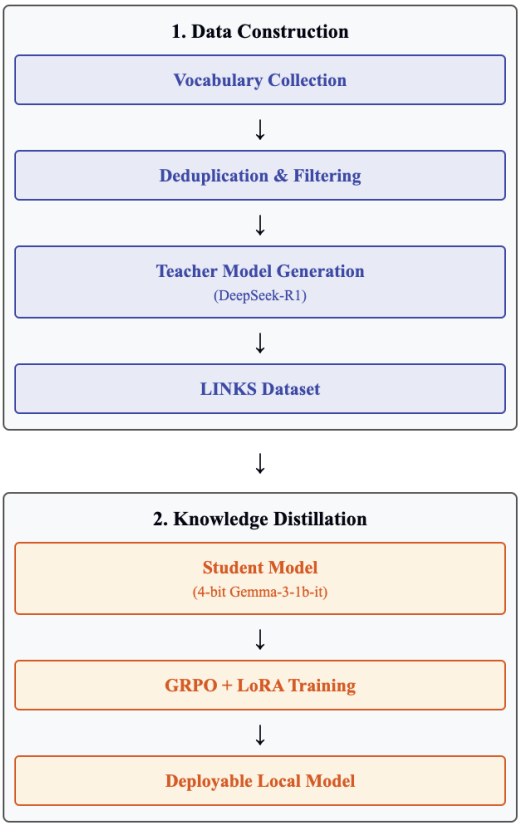

Figure 3: **LINKSYS** pipeline. The pipeline consists of two main components: (1) CoT data generation and (2) model distillation. We generate a dataset of mnemonics with reasoning traces using a large language model (LLM) as a teacher. Then we distill the reasoning capabilities of the teacher model into a smaller student model using GRPO Appendix D.4.1. Finally, we obtain **LINKSYS**, a smaller model that can generate mnemonics through linguistic reasoning.

4 Knowledge and reasoning distillation

4.1 Preparation

We first created a comprehensive training dataset. Following best practices in synthetic data generation with LLMs (Long et al., 2024, Open Thoughts Team, 2025), we designed a data construction pipeline with key components (Figure 3).

Vocabulary selection We collected 5,000 distinct words from three main sources: English as a foreign language tests (TOEFL iBT, IELTS Academic), standardized tests with verbal reasoning (SAT, GRE), and CEFR C1/C2 word lists. This ensured coverage of academic and abstract vocabulary that would benefit from mnemonic devices. After fuzzy-matching deduplication (threshold 95%), we refined the dataset and performed stratified random sampling (stratified by linguistic features used) to obtain 2,000 distinct vocabulary for post-training.

Prompt design and model selection Based on section 3's findings, we designed system and user prompts, focusing on the 10-shot CoT approach that demonstrated superior performance in eliciting linguistic reasoning and creating memorable, linguistically grounded mnemonics.

4.2 Dataset generation

Using our optimized prompts and vocabulary list, we generated the **LINKS** dataset of approximately 2,000 entries, each containing:

$$(\tau_i, m_i, e_i) \in \mathcal{D} \tag{1}$$

where τ_i represents the reasoning trace exploring linguistic features, m_i the generated mnemonic, and e_i an example sentence for vocabulary i. The dataset creation process can be formalized as:

$$(\tau_i, m_i, e_i) = f_{\text{teacher}}(v_i; p_{\text{CoT}}) \tag{2}$$

where f_{teacher} is the teacher model function, v_i is vocabulary i, and p_{CoT} is our 10-shot CoT prompt.

4.3 Quality control

To ensure the quality of the generated mnemonics, we implemented a multi-step validation process. We first filtered out any entries that did not meet our structured output format or contained incomplete reasoning traces. We then performed a manual review of a random sample of 200 entries to assess the linguistic grounding and coherence of the mnemonics. This review process involved checking for clear connections between the vocabulary and the mnemonic, as well as ensuring that the example sentence accurately reflected the vocabulary's meaning.

4.4 Training and inference

To transfer the linguistic reasoning capability to a smaller model, we implemented a distillation process using Group Relative Policy Optimization (GRPO).

We selected Google's Gemma-3-1b-it (Gemma-Team et al., 2025) as our student model due to its balance of performance and size (1 billion parameters). It can handle general-purpose tasks in multiple languages, and demonstrate instruction-following abilities. We used a 4-bit quantized version to further reduce memory requirements while maintaining performance (Dettmers et al., 2023).

Group Relative Policy Optimization (GRPO) We employed GRPO (DeepSeek-AI et al., 2025a) to distill the reasoning capabilities into the student model. For each input prompt q generating a vocabulary v, the model produces G candidate outputs:

$$O_q = \{o_1, o_2, \ldots, o_G\} \tag{3}$$

These outputs are evaluated using reward functions r_j that output scores, with the overall reward for output i calculated as:

$$r_i = \sum_{j=1}^{J} w_j r_j(o_i, v) \tag{4}$$

where w_j is the weight for reward function j. We provide more technical details in Appendix D.4.1, including the policy loss to be minimized.

We defined three reward functions r that encode essential characteristics of effective mnemonics: (**1**) adherence to the structured format with reasoning, mnemonic, and example, (**2**) usage of the target vocabulary in the mnemonic, penalizing bad mnemonics such as acronyms, and (**3**) explicit incorporation of linguistic features in Table 2 or a reasonable custom feature.

We assigned higher weights to criterion 3, and generated $G = 2$ candidates per training example to enable learning from comparisons. Training was performed on a single NVIDIA H100 GPU for approximately 4 hours (more details in Appendix E.3).

Low-Rank Adaptation (LoRA) We trained Gemma-3-1b-it using GRPO (Appendix D.3.3) wrapped in LoRA layers (Appendix E.2) to reduce the number of trainable parameters and rank-stabilized LoRA that maintains stability for adapters with higher ranks.

5 Evaluation

We evaluated the performance of **LINKSYS** in two main ways: (**1**) qualitative grading with LLM-as-a-judge and (**2**) pairwise preference using double-blind annotations. The first method involved using another LLM, OpenAI's o3-mini, to evaluate the quality of mnemonics generated by our model, while the second involved annotators comparing mnemonics generated by the base model, Gemma-3-1b-it, and our model, **LINKSYS**.

5.1 Qualitative grading with LLM judge

We designed a structured evaluation protocol using OpenAI's o3-mini as a judge to assess mnemonic quality. The LLM judge evaluated 200 pairs of mnemonics from our test set, with each pair generated by the base model (Gemma-3-1b-it) and our model **LINKSYS**.

We asked the judge to score each mnemonic independently on four metrics from our VAM model (Figure 1): (**1**) whether the vocabulary is used correctly in the mnemonic, (**2**) strength of association between the vocabulary and the mnemonic, (**3**) how clear and easy to understand the mnemonic is, (**4**) how memorable the mnemonic is, considering factors like concreteness, imageability, and distinctiveness. The last metric is (**5**) whether the mnemonic is linguistically grounded, meaning it incorporates linguistic features such as phonetics, morphology, or etymology.

Each criterion was evaluated on a binary scale (correct usage, linguistic grounding) or a 5-point Likert scale (association, clarity, memorability). The judge was instructed to provide six-field output: the score for each metric, and a brief reasoning for those scores.

We also calculated whether the difference in ratings was statistically significant using (**1**) a Wilconoxon signed-rank test for Likert ratings with paired samples and (**2**) a McNemar's test for paired boolean ratings. We set the significance level at 0.05.

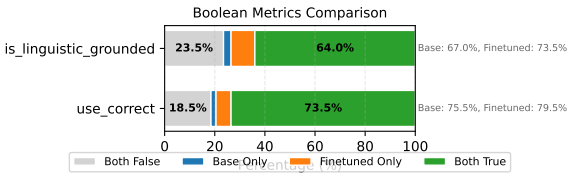

Figure 4: LLM-as-a-judge evaluation for boolean metrics (true/false): correct usage of vocabulary in mnemonic and linguistic grounding of mnemonic. **LINKSYS** shows improvement in both metrics, with notable gains in linguistic grounding (68% vs. 82%).

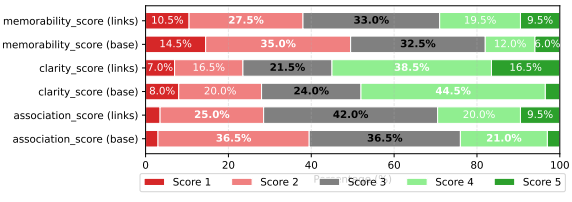

Figure 5: LLM-as-a-judge evaluation for 5-point Likert scale metrics. **LINKSYS** shows improvements across all three metrics, with the most significant gains in memorability (mean of 2.6 vs. 2.9)

We reported the distribution of OpenAI's o3-mini's

Table 1: Metric comaparison between the base (Gemma-3-1b-it) and **LINKSYS** models

Evaluation Metric	Base (μ)	LINKSYS (μ)	μ diff.	p-value	Effect size
Correct vocabulary usage (bool)	0.755	0.795	+0.040	0.077	–
Linguistic grounding (bool)	0.670	0.735	+0.065	**0.009**	–
Association strength (1-5)	2.845	3.070	+0.225	**<0.001**	0.464
Clarity (1-5)	3.155	3.410	+0.255	**<0.001**	0.481
Memorability (1-5)	2.600	2.900	+0.300	**<0.001**	0.566

Note: Bold p-values indicate statistically significant differences ($p < 0.05$). Effect sizes are Cohen's d, where 0.2 is small, 0.5 is medium, and 0.8 is large. Boolean metrics (first two rows) do not have effect sizes.

ratings in Figures 4 and 5, and summarized the difference in performance between the base model (Gemma-3-1b-it) and our model in Table 1. The results indicate that **LINKSYS** outperforms the base model across all evaluation metrics. Four of the five metrics showed statistically significant improvements ($p < 0.05$). The most substantial improvement was observed in memorability, with a mean difference of $+0.3$ points and a medium effect size (Cohen's $d = 0.566$). Clarity and semantic association also showed significant improvements with medium effect sizes.

While correct vocabulary usage showed a positive trend, this difference was not statistically significant ($p = 0.077$). This suggests that both models were already competent at using vocabulary correctly, with less room for improvement in this area.

5.2 Pairwise preference using blind annotations

To directly compare the quality of mnemonics, we conducted a blind annotation study. We randomly selected 100 mnemonics from our test set, with 50 each generated by the base model (Gemma-3-1b-it) and our model (**LINKSYS**). Two annotators (the author and an LLM) were asked to evaluate the mnemonics in pairs, choosing the better mnemonic for each pair while being blinded to the model used and the order of mnemonic presented in a pair[7]. This approach allowed us to understand the relative performance of each model. The annotators were instructed to consider the same criteria as in the LLM-as-a-judge evaluation, and indicated which mnemonic they preferred and why. More details in Appendix C.

Figure 6 reveals a strong preference for mnemonics generated by **LINKSYS** over those from the base model (64% vs. 36%). Annotators particularly valued mnemonics that incorporated multiple linguistic features and provided clear, concrete associations to abstract concepts.

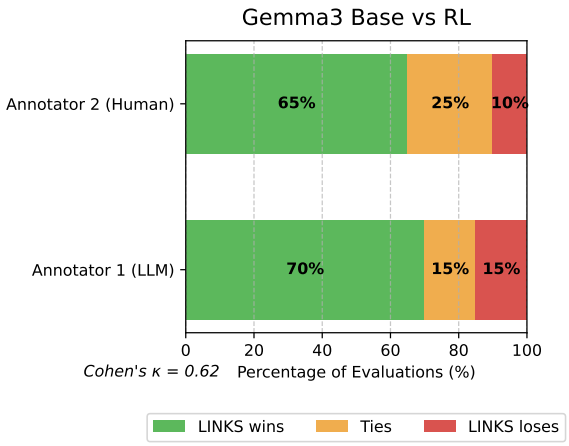

Figure 6: Pairwise preference using double-blind annotation. Y-axis shows the percentage of preference for each mnemonic generation method.

6 Discussion

7 Conclusion

This paper introduced

7.1 Limitations

Despite promising results, several limitations warrant acknowledgment. (1) Resource constraints limited the scale of our experiments (Sections 3 and 5) and the limited number of annotators in our double-blind study (Section 5.2) may have introduced bias in the evaluation process. Future work should consider using a larger and more diverse set of judges and annotators to ensure robustness and generalizability of the findings.

(2) Our evaluation focused primarily on intermediate measures of mnemonic quality rather than direct assessment of learning outcomes. Future work should include longitudinal studies measuring

[7]We acknowledge both using the author and using an LLM for evaluation as limitations in Section 7.1.

actual vocabulary retention using LLM-generated mnemonics.

(3) The use of LLM-as-a-judge for qualitative grading and pairwise preference evaluation may introduce biases. We acknowledged the potential for biases in LLMs, including self-enhancement bias (Panickssery et al., 2024), positional bias (Wang et al., 2024; Zheng et al., 2023), verbosity bias (Zheng et al., 2023), and others. Respectively, we attempted to mitigate these biases by employing a structured evaluation protocol, shuffling the order of mnemonics in a pair, enforcing a structured output format, and controlling for generated mnemonics length by both models in comparison (Gu et al., 2025). Additionally, we employed a double-blind annotation study with human annotators to validate the results. However, the limited number of annotators (two) may have introduced bias in the evaluation process. Future work should consider using a larger and more diverse set of judges and annotators to ensure robustness and generalizability of the findings.

(4) The focus on English words and English mnemonics may limit the generalizability of our findings to other types of vocabulary (e.g., phrasal verbs, idioms), other languages, and cross-lingual vocabulary-mnemonics. Future research should explore the applicability of our approach to different languages and cultural contexts, such as generating Vietnamese mnemonics to help learn Chinese vocabulary through radicals and cognates.

7.2 Future Work

Future research directions include expanding linguistic annotations to cover a broader range of features and vocabulary types, developing automated methods for linguistic feature extraction, and exploring personalized mnemonic generation that adapts to individual learning preferences and styles. Additionally, integrating multimodal elements that combine visual and textual mnemonics could further enhance learning effectiveness, particularly for concrete vocabulary.

Acknowledgements

I would like to express my gratitude to my capstone committee, Dr. Patrick Watson and Dr. Philip Sterne, for their guidance and support throughout this project. I also want to thank my classmates and friends for their encouragement and feedback during the development of this project.

Ethics Statement

This project was conducted in accordance with the ethical guidelines of Minerva University. The dataset used for training and evaluation was generated using an LLM, and a **subset** of generated mnemonics were reviewed for quality and appropriateness. We acknowledge the potential biases present in the training data and the need for continuous monitoring and improvement of the model's outputs. The final model is designed to be deployed locally, ensuring user privacy and data security.

The generated mnemonics are intended to be used as a supplementary tool for language learners and should **not** replace other language learning methods. We recommend that users critically evaluate the generated mnemonics and adapt them to their individual learning styles and preferences. We welcome feedback on any aspect of the paper to help improve the model's performance and address any ethical concerns that may arise.

References

Kenan Akarslan and Hasan Bedir. 2019. The effects of teaching word roots on the long term retention of English vocabulary. *International Journal of Educational Spectrum*, 1(1):1–11.

Jeanette Altarriba, Lisa M. Bauer, and Claudia Benvenuto. 1999. Concreteness, context availability, and imageability ratings and word associations for abstract, concrete, and emotion words. *Behavior Research Methods, Instruments, & Computers*, 31(4):578–602.

Richard C. Atkinson and Michael R. Raugh. 1975. An application of the mnemonic keyword method to the acquisition of a Russian vocabulary. *Journal of Experimental Psychology: Human Learning and Memory*, 1(2):126–133.

Nishant Balepur, Matthew Shu, Alexander Hoyle, Alison Robey, Shi Feng, Seraphina Goldfarb-Tarrant, and Jordan Lee Boyd-Graber. 2024. A SMART Mnemonic Sounds like "Glue Tonic": Mixing LLMs with Student Feedback to Make Mnemonic Learning Stick. In *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing*, pages 14202–14225, Miami, Florida, USA. Association for Computational Linguistics.

Bespoke Lab. 2025. Bespoke Lab Curator. bespokelabs.ai.

Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, Sandhini Agarwal, Ariel Herbert-Voss, Gretchen Krueger, Tom Henighan, Rewon Child, Aditya Ramesh, Daniel Ziegler, Jeffrey Wu, Clemens Winter, Chris

Hesse, Mark Chen, Eric Sigler, Mateusz Litwin, Scott Gray, Benjamin Chess, Jack Clark, Christopher Berner, Sam McCandlish, Alec Radford, Ilya Sutskever, and Dario Amodei. 2020. Language Models are Few-Shot Learners. In *Advances in Neural Information Processing Systems*, volume 33, pages 1877–1901. Curran Associates, Inc.

Alfredo Campos, Angeles Amor, and María Angeles González. 2004. The Importance of the Keyword-Generation Method in Keyword Mnemonics. *Experimental Psychology*, 51(2):125–131.

Alfredo Campos, Estefanía Camino, and María José Pérez-Fabello. 2011. Using the Keyword Mnemonics Method Among Adult Learners. *Educational Gerontology*, 37(4):327–335.

Alfredo Campos, María Angeles González, and Angeles Amor. 2003. Limitations of the mnemonic-keyword method. *The Journal of General Psychology*, 130(4):399–413.

DeepSeek-AI, Daya Guo, Dejian Yang, Haowei Zhang, Junxiao Song, Ruoyu Zhang, Runxin Xu, Qihao Zhu, Shirong Ma, Peiyi Wang, Xiao Bi, Xiaokang Zhang, Xingkai Yu, Yu Wu, Z. F. Wu, Zhibin Gou, Zhihong Shao, Zhuoshu Li, Ziyi Gao, Aixin Liu, Bing Xue, Bingxuan Wang, Bochao Wu, Bei Feng, Chengda Lu, Chenggang Zhao, Chengqi Deng, Chenyu Zhang, Chong Ruan, Damai Dai, Deli Chen, Dongjie Ji, Erhang Li, Fangyun Lin, Fucong Dai, Fuli Luo, Guangbo Hao, Guanting Chen, Guowei Li, H. Zhang, Han Bao, Hanwei Xu, Haocheng Wang, Honghui Ding, Huajian Xin, Huazuo Gao, Hui Qu, Hui Li, Jianzhong Guo, Jiashi Li, Jiawei Wang, Jingchang Chen, Jingyang Yuan, Junjie Qiu, Junlong Li, J. L. Cai, Jiaqi Ni, Jian Liang, Jin Chen, Kai Dong, Kai Hu, Kaige Gao, Kang Guan, Kexin Huang, Kuai Yu, Lean Wang, Lecong Zhang, Liang Zhao, Litong Wang, Liyue Zhang, Lei Xu, Leyi Xia, Mingchuan Zhang, Minghua Zhang, Minghui Tang, Meng Li, Miaojun Wang, Mingming Li, Ning Tian, Panpan Huang, Peng Zhang, Qiancheng Wang, Qinyu Chen, Qiushi Du, Ruiqi Ge, Ruisong Zhang, Ruizhe Pan, Runji Wang, R. J. Chen, R. L. Jin, Ruyi Chen, Shanghao Lu, Shangyan Zhou, Shanhuang Chen, Shengfeng Ye, Shiyu Wang, Shuiping Yu, Shunfeng Zhou, Shuting Pan, S. S. Li, Shuang Zhou, Shaoqing Wu, Shengfeng Ye, Tao Yun, Tian Pei, Tianyu Sun, T. Wang, Wangding Zeng, Wanjia Zhao, Wen Liu, Wenfeng Liang, Wenjun Gao, Wenqin Yu, Wentao Zhang, W. L. Xiao, Wei An, Xiaodong Liu, Xiaohan Wang, Xiaokang Chen, Xiaotao Nie, Xin Cheng, Xin Liu, Xin Xie, Xingchao Liu, Xinyu Yang, Xinyuan Li, Xuecheng Su, Xuheng Lin, X. Q. Li, Xiangyue Jin, Xiaojin Shen, Xiaosha Chen, Xiaowen Sun, Xiaoxiang Wang, Xinnan Song, Xinyi Zhou, Xianzu Wang, Xinxia Shan, Y. K. Li, Y. Q. Wang, Y. X. Wei, Yang Zhang, Yanhong Xu, Yao Li, Yao Zhao, Yaofeng Sun, Yaohui Wang, Yi Yu, Yichao Zhang, Yifan Shi, Yiliang Xiong, Ying He, Yishi Piao, Yisong Wang, Yixuan Tan, Yiyang Ma, Yiyuan Liu, Yongqiang Guo, Yuan Ou, Yuduan Wang, Yue Gong, Yuheng Zou, Yujia He, Yunfan Xiong, Yuxiang Luo, Yuxiang You, Yuxuan Liu, Yuyang Zhou, Y. X. Zhu, Yanhong Xu, Yanping Huang, Yaohui Li, Yi Zheng, Yuchen Zhu, Yunxian Ma, Ying Tang, Yukun Zha, Yuting Yan, Z. Z. Ren, Zehui Ren, Zhangli Sha, Zhe Fu, Zhean Xu, Zhenda Xie, Zhengyan Zhang, Zhewen Hao, Zhicheng Ma, Zhigang Yan, Zhiyu Wu, Zihui Gu, Zijia Zhu, Zijun Liu, Zilin Li, Ziwei Xie, Ziyang Song, Zizheng Pan, Zhen Huang, Zhipeng Xu, Zhongyu Zhang, and Zhen Zhang. 2025a. DeepSeek-R1: Incentivizing Reasoning Capability in LLMs via Reinforcement Learning. Technical Report arXiv:2501.12948, DeepSeek-AI.

DeepSeek-AI, Aixin Liu, Bei Feng, Bing Xue, Bingxuan Wang, Bochao Wu, Chengda Lu, Chenggang Zhao, Chengqi Deng, Chenyu Zhang, Chong Ruan, Damai Dai, Daya Guo, Dejian Yang, Deli Chen, Dongjie Ji, Erhang Li, Fangyun Lin, Fucong Dai, Fuli Luo, Guangbo Hao, Guanting Chen, Guowei Li, H. Zhang, Han Bao, Hanwei Xu, Haocheng Wang, Haowei Zhang, Honghui Ding, Huajian Xin, Huazuo Gao, Hui Li, Hui Qu, J. L. Cai, Jian Liang, Jianzhong Guo, Jiaqi Ni, Jiashi Li, Jiawei Wang, Jin Chen, Jingchang Chen, Jingyang Yuan, Junjie Qiu, Junlong Li, Junxiao Song, Kai Dong, Kai Hu, Kaige Gao, Kang Guan, Kexin Huang, Kuai Yu, Lean Wang, Lecong Zhang, Lei Xu, Leyi Xia, Liang Zhao, Litong Wang, Liyue Zhang, Meng Li, Miaojun Wang, Mingchuan Zhang, Minghua Zhang, Minghui Tang, Mingming Li, Ning Tian, Panpan Huang, Peiyi Wang, Peng Zhang, Qiancheng Wang, Qihao Zhu, Qinyu Chen, Qiushi Du, R. J. Chen, R. L. Jin, Ruiqi Ge, Ruisong Zhang, Ruizhe Pan, Runji Wang, Runxin Xu, Ruoyu Zhang, Ruyi Chen, S. S. Li, Shanghao Lu, Shangyan Zhou, Shanhuang Chen, Shaoqing Wu, Shengfeng Ye, Shengfeng Ye, Shirong Ma, Shiyu Wang, Shuang Zhou, Shuiping Yu, Shunfeng Zhou, Shuting Pan, T. Wang, Tao Yun, Tian Pei, Tianyu Sun, W. L. Xiao, Wangding Zeng, Wanjia Zhao, Wei An, Wen Liu, Wenfeng Liang, Wenjun Gao, Wenqin Yu, Wentao Zhang, X. Q. Li, Xiangyue Jin, Xianzu Wang, Xiao Bi, Xiaodong Liu, Xiaohan Wang, Xiaojin Shen, Xiaokang Chen, Xiaokang Zhang, Xiaosha Chen, Xiaotao Nie, Xiaowen Sun, Xiaoxiang Wang, Xin Cheng, Xin Liu, Xin Xie, Xingchao Liu, Xingkai Yu, Xinnan Song, Xinxia Shan, Xinyi Zhou, Xinyu Yang, Xinyuan Li, Xuecheng Su, Xuheng Lin, Y. K. Li, Y. Q. Wang, Y. X. Wei, Y. X. Zhu, Yang Zhang, Yanhong Xu, Yanhong Xu, Yanping Huang, Yao Li, Yao Zhao, Yaofeng Sun, Yaohui Li, Yaohui Wang, Yi Yu, Yi Zheng, Yichao Zhang, Yifan Shi, Yiliang Xiong, Ying He, Ying Tang, Yishi Piao, Yisong Wang, Yixuan Tan, Yiyang Ma, Yiyuan Liu, Yongqiang Guo, Yu Wu, Yuan Ou, Yuchen Zhu, Yuduan Wang, Yue Gong, Yuheng Zou, Yujia He, Yukun Zha, Yunfan Xiong, Yunxian Ma, Yuting Yan, Yuxiang Luo, Yuxiang You, Yuxuan Liu, Yuyang Zhou, Z. F. Wu, Z. Z. Ren, Zehui Ren, Zhangli Sha, Zhe Fu, Zhean Xu, Zhen Huang, Zhen Zhang, Zhenda Xie, Zhengyan Zhang, Zhewen Hao, Zhibin Gou, Zhicheng Ma, Zhigang Yan, Zhihong Shao, Zhipeng Xu, Zhiyu Wu, Zhongyu Zhang, Zhuoshu Li, Zihui Gu, Zijia Zhu, Zijun Liu, Zilin Li, Ziwei Xie, Ziyang Song, Ziyi Gao, and Zizheng Pan. 2025b. DeepSeek-V3 Technical Report.

Tim Dettmers, Artidoro Pagnoni, Ari Holtzman, and Luke Zettlemoyer. 2023. QLoRA: Efficient Finetuning of Quantized LLMs. In *Thirty-Seventh Conference on Neural Information Processing Systems*.

Dennis L. Foth. 1973. Mnemonic technique effectiveness as a function of word abstractness and mediation instructions. *Journal of Verbal Learning and Verbal Behavior*, 12(3):239–245.

Dr Raja Sekhar Gangavarapu. 2024. Using etymology as a vocabulary learning approach for expanding learners' vocabulary in English: A psycholinguistic analysis. *International Journal of Interdisciplinary Cultural Studies, ISSN:2327-008XE-ISSN:2327-2554*, 19(2):170–182.

Gemma-Team, Aishwarya Kamath, Johan Ferret, Nino Vieillard, Ramona Merhej, Sarah Perrin, Tatiana Matejovicova, Alexandre Ramé, Morgane Rivière, Louis Rouillard, Thomas Mesnard, Geoffrey Cideron, Jean-bastien Grill, Sabela Ramos, Edouard Yvinec, Michelle Casbon, Etienne Pot, Ivo Penchev, Gaël Liu, Francesco Visin, Kathleen Kenealy, Lucas Beyer, Xiaohai Zhai, Anton Tsitsulin, Robert Busa-Fekete, Alex Feng, Noveen Sachdeva, Benjamin Coleman, Yi Gao, Basil Mustafa, Iain Barr, Emilio Parisotto, David Tian, Matan Eyal, Colin Cherry, Jan-Thorsten Peter, Danila Sinopalnikov, Surya Bhupatiraju, Rishabh Agarwal, Mehran Kazemi, Dan Malkin, Ravin Kumar, David Vilar, Idan Brusilovsky, Jiaming Luo, Andreas Steiner, Abe Friesen, Abhanshu Sharma, Abheesht Sharma, Adi Mayrav Gilady, Adrian Goedeckemeyer, Alaa Saade, Alex Feng, Alexander Kolesnikov, Alexei Bendebury, Alvin Abdagic, Amit Vadi, András György, André Susano Pinto, Anil Das, Ankur Bapna, Antoine Miech, Antoine Yang, Antonia Paterson, Ashish Shenoy, Ayan Chakrabarti, Bilal Piot, Bo Wu, Bobak Shahriari, Bryce Petrini, Charlie Chen, Charline Le Lan, Christopher A. Choquette-Choo, C. J. Carey, Cormac Brick, Daniel Deutsch, Danielle Eisenbud, Dee Cattle, Derek Cheng, Dimitris Paparas, Divyashree Shivakumar Sreepathihalli, Doug Reid, Dustin Tran, Dustin Zelle, Eric Noland, Erwin Huizenga, Eugene Kharitonov, Frederick Liu, Gagik Amirkhanyan, Glenn Cameron, Hadi Hashemi, Hanna Klimczak-Plucińska, Harman Singh, Harsh Mehta, Harshal Tushar Lehri, Hussein Hazimeh, Ian Ballantyne, Idan Szpektor, Ivan Nardini, Jean Pouget-Abadie, Jetha Chan, Joe Stanton, John Wieting, Jonathan Lai, Jordi Orbay, Joseph Fernandez, Josh Newlan, Ju-yeong Ji, Jyotinder Singh, Kat Black, Kathy Yu, Kevin Hui, Kiran Vodrahalli, Klaus Greff, Linhai Qiu, Marcella Valentine, Marina Coelho, Marvin Ritter, Matt Hoffman, Matthew Watson, Mayank Chaturvedi, Michael Moynihan, Min Ma, Nabila Babar, Natasha Noy, Nathan Byrd, Nick Roy, Nikola Momchev, Nilay Chauhan, Noveen Sachdeva, Oskar Bunyan, Pankil Botarda, Paul Caron, Paul Kishan Rubenstein, Phil Culliton, Philipp Schmid, Pier Giuseppe Sessa, Pingmei Xu, Piotr Stanczyk, Pouya Tafti, Rakesh Shivanna, Renjie Wu, Renke Pan, Reza Rokni, Rob Willoughby, Rohith Vallu, Ryan Mullins, Sammy Jerome, Sara Smoot, Sertan Girgin, Shariq Iqbal, Shashir Reddy, Shruti Sheth, Siim Põder, Sijal Bhatnagar, Sindhu Raghuram Panyam, Sivan Eiger, Susan Zhang, Tianqi Liu, Trevor Yacovone, Tyler Liechty, Uday Kalra, Utku Evci, Vedant Misra, Vincent Roseberry, Vlad Feinberg, Vlad Kolesnikov, Woohyun Han, Woosuk Kwon, Xi Chen, Yinlam Chow, Yuvein Zhu, Zichuan Wei, Zoltan Egyed, Victor Cotruta, Minh Giang, Phoebe Kirk, Anand Rao, Kat Black, Nabila Babar, Jessica Lo, Erica Moreira, Luiz Gustavo Martins, Omar Sanseviero, Lucas Gonzalez, Zach Gleicher, Tris Warkentin, Vahab Mirrokni, Evan Senter, Eli Collins, Joelle Barral, Zoubin Ghahramani, Raia Hadsell, Yossi Matias, D. Sculley, Slav Petrov, Noah Fiedel, Noam Shazeer, Oriol Vinyals, Jeff Dean, Demis Hassabis, Koray Kavukcuoglu, Clement Farabet, Elena Buchatskaya, Jean-Baptiste Alayrac, Rohan Anil, Dmitry, Lepikhin, Sebastian Borgeaud, Olivier Bachem, Armand Joulin, Alek Andreev, Cassidy Hardin, Robert Dadashi, and Léonard Hussenot. 2025. Gemma 3 Technical Report. Technical Report arXiv:2503.19786, Google DeepMind.

Jiawei Gu, Xuhui Jiang, Zhichao Shi, Hexiang Tan, Xuehao Zhai, Chengjin Xu, Wei Li, Yinghan Shen, Shengjie Ma, Honghao Liu, Saizhuo Wang, Kun Zhang, Yuanzhuo Wang, Wen Gao, Lionel Ni, and Jian Guo. 2025. A Survey on LLM-as-a-Judge.

Stefan Hackmann, Haniyeh Mahmoudian, Mark Steadman, and Michael Schmidt. 2024. Word Importance Explains How Prompts Affect Language Model Outputs.

Edward J. Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, and Weizhu Chen. 2021. LoRA: Low-Rank Adaptation of Large Language Models. In *International Conference on Learning Representations*.

Nikhil Kandpal, Haikang Deng, Adam Roberts, Eric Wallace, and Colin Raffel. 2023. Large Language Models Struggle to Learn Long-Tail Knowledge. In *Proceedings of the 40th International Conference on Machine Learning*, pages 15696–15707. PMLR.

Jude Khouja, Karolina Korgul, Simi Hellsten, Lingyi Yang, Vlad Neacsu, Harry Mayne, Ryan Kearns, Andrew Bean, and Adam Mahdi. 2025. LINGOLY-TOO: Disentangling Memorisation from Reasoning with Linguistic Templatisation and Orthographic Obfuscation.

Takeshi Kojima, Shixiang Shane Gu, Machel Reid, Yutaka Matsuo, and Yusuke Iwasawa. 2022. Large language models are zero-shot reasoners. In *Proceedings of the 36th International Conference on Neural Information Processing Systems*, NIPS '22, pages 22199–22213, Red Hook, NY, USA. Curran Associates Inc.

Juliana K. Leding. 2019. Adaptive memory: Animacy, threat, and attention in free recall. *Memory & Cognition*, 47(3):383–394.

Jaewook Lee and Andrew Lan. 2023. SmartPhone: Exploring Keyword Mnemonic with Auto-generated Verbal and Visual Cues. In *Artificial Intelligence in Education: 24th International Conference, AIED 2023, Tokyo, Japan, July 3–7, 2023, Proceedings*, pages 16–27, Berlin, Heidelberg. Springer-Verlag.

Jaewook Lee, Hunter McNichols, and Andrew Lan. 2024. Exploring Automated Keyword Mnemonics Generation with Large Language Models via Overgenerate-and-Rank. In *Findings of the Association for Computational Linguistics: EMNLP 2024*, pages 5521–5542,

Miami, Florida, USA. Association for Computational Linguistics.

Lin Long, Rui Wang, Ruixuan Xiao, Junbo Zhao, Xiao Ding, Gang Chen, and Haobo Wang. 2024. On LLMs-Driven Synthetic Data Generation, Curation, and Evaluation: A Survey. In *Findings of the Association for Computational Linguistics: ACL 2024*, pages 11065–11082, Bangkok, Thailand. Association for Computational Linguistics.

Christopher R. Madan. 2021. Exploring word memorability: How well do different word properties explain item free-recall probability? *Psychonomic Bulletin & Review*, 28(2):583–595.

Swaroop Mishra, Daniel Khashabi, Chitta Baral, Yejin Choi, and Hannaneh Hajishirzi. 2022. Reframing Instructional Prompts to GPTk's Language. In *Findings of the Association for Computational Linguistics: ACL 2022*, pages 589–612, Dublin, Ireland. Association for Computational Linguistics.

Open Thoughts Team. 2025. Open Thoughts.

Long Ouyang, Jeff Wu, Xu Jiang, Diogo Almeida, Carroll L. Wainwright, Pamela Mishkin, Chong Zhang, Sandhini Agarwal, Katarina Slama, Alex Ray, John Schulman, Jacob Hilton, Fraser Kelton, Luke Miller, Maddie Simens, Amanda Askell, Peter Welinder, Paul Christiano, Jan Leike, and Ryan Lowe. 2022. Training language models to follow instructions with human feedback.

Gözde Özbal, Daniele Pighin, and Carlo Strapparava. 2014. Automation and Evaluation of the Keyword Method for Second Language Learning. In *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 352–357, Baltimore, Maryland. Association for Computational Linguistics.

Arjun Panickssery, Samuel R. Bowman, and Shi Feng. 2024. LLM Evaluators Recognize and Favor Their Own Generations. *Advances in Neural Information Processing Systems*, 37:68772–68802.

Herbert D. Pierson. 1989. Using etymology in the classroom. *ELT Journal*, 43(1):57–63.

Paul R. Pintrich. 2002. The Role of Metacognitive Knowledge in Learning, Teaching, and Assessing. *Theory Into Practice*, 41(4):219–225.

Michael Pressley, Joel R. Levin, and Harold D. Delaney. 1982. The Mnemonic Keyword Method. *Review of Educational Research*, 52(1):61–91.

Raghav Ramji and Keshav Ramji. 2024. Inductive Linguistic Reasoning with Large Language Models.

Jane L. Rankin and James V. Hinrichs. 1983. Age, Presentation Rate, and the Effectiveness of Structural and Semantic Recall Cues. *Journal of Gerontology*, 38(5):593–596.

Mustafa Sarıoğlu and Çiğdem Karatepe. 2024. The Use of Mnemonics to Minimize the Interfering Effects of Teaching New Words in Semantic Sets to Learners of English as a Foreign Language. *Applied Cognitive Psychology*, 38(5):e4251.

Manolis Savva, Angel X. Chang, Christopher D. Manning, and Pat Hanrahan. 2014. TransPhoner: Automated mnemonic keyword generation. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, pages 3725–3734, Toronto Ontario Canada. ACM.

Paula J. Schwanenflugel, Carolyn Akin, and Wei-Ming Luh. 1992. Context availability and the recall of abstract and concrete words. *Memory & Cognition*, 20(1):96–104.

Kai Sun, Yifan Xu, Hanwen Zha, Yue Liu, and Xin Luna Dong. 2024. Head-to-Tail: How Knowledgeable are Large Language Models (LLMs)? A.K.A. Will LLMs Replace Knowledge Graphs? In *Proceedings of the 2024 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (Volume 1: Long Papers)*, pages 311–325, Mexico City, Mexico. Association for Computational Linguistics.

Ashima Suvarna, Harshita Khandelwal, and Nanyun Peng. 2024. PhonologyBench: Evaluating Phonological Skills of Large Language Models. In *Proceedings of the 1st Workshop on Towards Knowledgeable Language Models (KnowLLM 2024)*, pages 1–14, Bangkok, Thailand. Association for Computational Linguistics.

Tina Tseng, Amanda Stent, and Domenic Maida. 2020. Best Practices for Managing Data Annotation Projects.

Janet G. van Hell and Andrea Candia Mahn. 1997. Keyword Mnemonics Versus Rote Rehearsal: Learning Concrete and Abstract Foreign Words by Experienced and Inexperienced Learners. *Language Learning*, 47(3):507–546.

Andreas Waldis, Yotam Perlitz, Leshem Choshen, Yufang Hou, and Iryna Gurevych. 2024. Holmes: A Benchmark to Assess the Linguistic Competence of Language Models. *Transactions of the Association for Computational Linguistics*, 12:1616–1647.

Alvin Y. Wang, Margaret H. Thomas, and Judith A. Ouellette. 1992. Keyword mnemonic and retention of second-language vocabulary words. *Journal of Educational Psychology*, 84(4):520–528.

Peiyi Wang, Lei Li, Liang Chen, Zefan Cai, Dawei Zhu, Binghuai Lin, Yunbo Cao, Lingpeng Kong, Qi Liu, Tianyu Liu, and Zhifang Sui. 2024. Large Language Models are not Fair Evaluators. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 9440–9450, Bangkok, Thailand. Association for Computational Linguistics.

Xuezhi Wang, Jason Wei, Dale Schuurmans, Quoc V. Le, Ed H. Chi, Sharan Narang, Aakanksha Chowdhery, and Denny Zhou. 2022. Self-Consistency Improves Chain of Thought Reasoning in Language Models. In *The Eleventh International Conference on Learning Representations*.

Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, Brian Ichter, Fei Xia, Ed H. Chi, Quoc V. Le, and Denny Zhou. 2022. Chain-of-Thought Prompting Elicits Reasoning in Large Language Models. In *Advances in Neural Information Processing Systems*.

Lingling Xu, Haoran Xie, Si-Zhao Joe Qin, Xiaohui Tao, and Fu Lee Wang. 2023. Parameter-Efficient Fine-Tuning Methods for Pretrained Language Models: A Critical Review and Assessment.

Silei Xu, Wenhao Xie, Lingxiao Zhao, and Pengcheng He. 2025. Chain of Draft: Thinking Faster by Writing Less.

Michihiro Yasunaga, Xinyun Chen, Yujia Li, Panupong Pasupat, Jure Leskovec, Percy Liang, Ed H. Chi, and Denny Zhou. 2023. Large Language Models as Analogical Reasoners. In *The Twelfth International Conference on Learning Representations*.

Fan Yin, Jesse Vig, Philippe Laban, Shafiq Joty, Caiming Xiong, and Chien-Sheng Wu. 2023. Did You Read the Instructions? Rethinking the Effectiveness of Task Definitions in Instruction Learning. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 3063–3079, Toronto, Canada. Association for Computational Linguistics.

Jie Zhang. 2013. Application of Etymology and Semantic Field Theory for Second Language Acquisition. *US-China Foreign Language*, 11(11):834–839.

Lianmin Zheng, Wei-Lin Chiang, Ying Sheng, Siyuan Zhuang, Zhanghao Wu, Yonghao Zhuang, Zi Lin, Zhuohan Li, Dacheng Li, Eric P. Xing, Hao Zhang, Joseph E. Gonzalez, and Ion Stoica. 2023. Judging LLM-as-a-judge with MT-bench and Chatbot Arena. In *Proceedings of the 37th International Conference on Neural Information Processing Systems*, NIPS '23, pages 46595–46623, Red Hook, NY, USA. Curran Associates Inc.

A Mnemonics: Linguistic Features and Characteristics

A.1 Full mnemonic characteristiics

- Clear explanation linking the vocabulary to the mnemonic.
- Correct usage and definition of the vocabulary within the mnemonic.
- Strong association between the vocabulary and the mnemonic.
- Mnemonic is easy to understand, using similar or simpler vocabulary than the target term.
- Mnemonic is memorable, incorporating animate or concrete imagery, relevant contexts, or elements that evoke emotional responses.

One of the following could make bad mnemonics

- Lack one of the three components.
- Incorrect definition or usage of the vocabulary.
- Circular association where the mnemonic simply repeats the vocabulary without adding meaning. Polysemous words (words with multiple meanings) are not considered circular.
- Weak or unclear association between mnemonic and vocabulary.
- Use semantically complex or obscure words that are more difficult than the target vocabulary.
- Mnemonic is abstract, making it hard to visualize or relate to.
- Use offensive or inappropriate language.

Unless specifically requested by users, LLMs are prompted to avoid culturally-specific mnemonics, such as those based on idioms or proverbs, as they may not be universally understood. This is particularly important for learners from diverse backgrounds and cultures.

A.2 Linguistic features

B Prompt usage

All of the following prompts are system prompts for DeepSeek-R1. We also added Appendix A.1 as mnemonic requirements for all prompts, and added the phrase "Let's think step by step" to the prompt variants sent to DeepSeek-V3. To conserve tokens, we remove unnecessary words that did not contribute to task instructions, such as modifiers.

Vanilla vs. Alternative Phrasing

Vanilla prompt

> A mnemonic to help learn and remember meaning of English vocabulary: {term}. Mnemonic should have following characteristics:
> [INSERT MNEMONIC REQUIREMENTS HERE].

Vanilla-Alternative prompt, with "linguistically grounded" added:

> A linguistically grounded mnemonic

We observed that the vanilla prompts, with the term "mnemonic", often elicit backronyms (i.e. an existing word's letters are expanded into a phrase), initialisms or list. We hypothesized that this was because commonly encountered mnemonics are used for long information and non-linguistic contexts, and the training data reflects that popular use of mnemonic devices. This effect can be mitigated by adding the term "lingusitically grounded" to the prompt, to steer the model towards analyzing linguistic features, especially etymology and morphology.

Structured Output and Task Description

We found improved performance with prompts that explicitly request broader linguistic analysis and provides structured output format with output descriptions (Mishra et al., 2022; Yin et al., 2023). We also tried to reduce the overthinking tendency in LLMs

> Generate a linguistically grounded mnemonic to help me learn and remember the meaning of English vocabulary: {term}.
> Analyze linguistic features for this word (etymology, morphology, phonetics, orthography, semantics, etc). Stop linguistic analysis when you have a good linguistic connection. You must use that linguistic feature to form a mnemonic for the word.
> Mnemonic should have following characteristics: [INSERT MNEMONIC REQUIREMENTS HERE].

Table 2: Examples of feature categories for English words.

feature	description	example
phonetics	sound patterns	*apparent* sounds like "a bare Asian."
orthography	written/spelling patterns	*abet* looks like "a + bet."
morphology	modern English forms, including free and bound morphemes	*aggrandize* = a + grand + –ize, to mean to make grander.
etymology	origin and history	*adumbrate* comes from Latin ad- (to, on) + umbra (shade) + ate, to mean foreshadow or outline.
semantics	meaning and semantic relationships	*confound* has similar meaning and history with 'confuse'.

Provide output in this format:
- linguistic_feature: chosen linguistic feature for mnemonic
- mnemonic: association + mnemonic
- example: example sentence of the vocabulary in context

This approach yielded a higher percentage of mnemonics with clear linguistic association, and better mnemonics overall. We also found that the model was more likely to use the same linguistic feature in the mnemonic as in the analysis, which is a key requirement for our task.

Added CoT examples

For this prompt, we reused the task instructions and added 10 human-written CoT examples, each demonstrating the process of finding linguistic association of the vocabulary before constructing a mnemonic.

B.1 Double-blind annotation

C Annotation details

For our double-blind annotation study, we followed best practices from Tseng et al. (2020) to ensure unbiased evaluation. The annotation process involved the following steps:
(**1**) We randomly selected 50 vocabulary terms from our test set, ensuring representation across different linguistic categories and vocabulary difficulty levels.
(**2**) For each term, we generated mnemonics using both the base model (Gemma-3-1b-it) and our fine-tuned model (**LINKSYS**).
(**3**) We created annotation pairs where each pair contained two mnemonics for the same vocabulary term, one from each model, with the order randomized to prevent position bias (Wang et al., 2024).

(**4**) Annotators were not informed which mnemonic came from which model, ensuring truly blind evaluation.

(**5**) Annotators were asked to select the better mnemonic based on five criteria: correct usage, linguistic grounding, association strength, clarity, and memorability.

(**6**) For each annotation pair, annotators also provided a brief justification for their preference to enable qualitative analysis.

The two annotators were the author and OpenAI's o3-mini. While this is a limitation of our study that we acknowledge in Section 7.1, we took several measures to reduce potential bias:

(**1**) The author completed annotations before running any quantitative analysis to prevent knowledge of statistical patterns from influencing judgment.

(**2**) OpenAI's o3-mini was prompted to evaluate mnemonics according to the criteria in Figure 1 without any information about which model generated which mnemonic.

(**3**) We calculated inter-annotator agreement using Cohen's kappa to assess reliability, achieving a kappa value of 0.72, indicating substantial agreement.

For future work, we plan to expand the annotation team to include language education experts and language learners to provide more diverse perspectives on mnemonic quality.

C.1 Examples

D Technical preliminaries

D.1 In-context learning

D.1.1 Chain-of-Thought (CoT) prompting

CoT (Wei et al., 2022) is a prompting technique that encourages LLMs to generate intermediate reasoning steps before arriving at a final answer. This approach has been shown to improve performance on complex tasks by guiding the model through a structured thought process.

D.2 Neural Language Models and Transformer Architecture

Neural language models are probabilistic frameworks that assign probabilities to sequences of words or subword units, known as tokens. A token is the smallest unit of text that the model processes, which can be as granular as individual characters, subwords, or entire words, depending on the tokenization strategy employed.

Given a sequence of tokens $\mathbf{x} = (x_1, x_2, \ldots, x_T)$, a language model estimates the joint probability $P(\mathbf{x})$ by factorizing it into conditional probabilities:

$$P(\mathbf{x}) = \prod_{t=1}^{T} P(x_t \mid x_1, x_2, \ldots, x_{t-1}) \quad (5)$$

At each time step t, the model predicts the next token x_t based on preceding sequence $(x_1, x_2, \ldots, x_{t-1})$. This autoregressive approach enables the generation of coherent text by sequentially predicting subsequent tokens.

The Transformer architecture underpins many state-of-the-art language models due to its efficiency and capability to model long-range dependencies. It utilizes self-attention mechanisms to weigh the relevance of each token in a sequence relative to others, regardless of their positions. The architecture comprises stacked layers, each including multi-head self-attention and position-wise feed-forward networks, facilitating parallelization and effective learning of complex patterns in data.

D.2.1 Tokenizer

A tokenizer is a preprocessing tool that converts raw text into tokens, aligning the text with the LM's vocabulary. Tokenizers can employ various strategies, such as word-based, character-based, or subword-based tokenization, each with distinct advantages and use cases.

Byte Pair Encoding (BPE) is a subword tokenization algorithm that operates on the byte representation of text, enabling consistent handling of various scripts and special characters. It iteratively merges the most frequent pairs of adjacent bytes to form subword units, constructing a vocabulary that efficiently represents the training corpus. This method allows the tokenizer to decompose rare words into meaningful subword components, enhancing the model's capacity to process diverse and unseen terms.

For instance, the word "preposterous" might be tokenized into subwords like "pre", "poster", and "ous," facilitating the model's understanding and generation of these subwords in novel contexts. This subword granularity enables the model to generalize across morphologically complex words and out-of-vocabulary words, enhancing its robustness and vocabulary coverage. However, not all subwords are valid morphemes, which can limit the model's ability to capture morphological structure accurately. For instance, `tiktoken` (OpenAI's tokenizer)[8] recognizes "ephemeral" as a single subword rather than three morphemes ("ept", "hemera", "-al"), because the affixes are not explicitly segmented, and 'epheremal' is a rare word so BPE better learns it as a single token.

D.3 Family of Fine-Tuning Methods

Fine-tuning is the process of adapting a pre-trained model to a specific task T or domain D by updating its parameters on a target dataset $\mathcal{D}$. This process is crucial for leveraging pre-trained models' knowledge and enhancing their performance on downstream tasks.

There are several approaches to fine-tuning, which can be categorized by: 1. the availability of labeled data (supervised vs unsupervised fine-tuning), 2. the extent of parameter updates (full-parameter vs parameter-efficient fine-tuning), and 3. task. We focus on supervised fine-tuning, which involves minimizing a task-specific loss function over a labeled dataset.

D.3.1 Supervised Fine-Tuning (SFT)

SFT involves adapting a pre-trained model to a target task by minimizing a task-specific loss function over a labeled dataset. For a dataset $\mathcal{D} = \{(\mathbf{x}^{(i)}, \mathbf{y}^{(i)})\}_{i=1}^{N}$, where $\mathbf{x}^{(i)}$ is the input and $\mathbf{y}^{(i)}$ is the target output, the objective is to minimize:

[8] https://platform.openai.com/tokenizer

$$\mathcal{L} = \frac{1}{N} \sum_{i=1}^{N} \ell(f(\mathbf{x}^{(i)}; \theta), \mathbf{y}^{(i)}) \tag{6}$$

where $f(\mathbf{x}; \theta)$ represents the model's output with parameters θ, and ℓ is the loss function, typically cross-entropy loss.

D.3.2 Instruction tuning

Instruction-tuning is a specialized form of SFT (Appendix D.3.1) where models are trained on datasets comprising instruction-response pairs. This approach enables models to generalize across various tasks described by natural language instructions, enhancing their ability to follow diverse prompts.

Formally, an instruction-tuning dataset consists of pairs $\{(\mathbf{I}^{(i)}, \mathbf{y}^{(i)})\}_{i=1}^{N}$ or triplets $\{(\mathbf{I}^{(i)}, \mathbf{x}^{(i)}, \mathbf{y}^{(i)})\}_{i=1}^{N}$, where $\mathbf{I}^{(i)}$ denotes the instruction, $\mathbf{x}^{(i)}$ is the optional input, and $\mathbf{y}^{(i)}$ is the desired output. The training objective is to minimize the loss:

$$\mathcal{L} = \frac{1}{N} \sum_{i=1}^{N} \ell(f(\mathbf{I}^{(i)}, \mathbf{x}^{(i)}; \theta), \mathbf{y}^{(i)}) \tag{7}$$

where f represents the model parameterized by θ, and ℓ is the loss function measuring the discrepancy between the model's prediction and the target output.

D.3.3 Parameter-Efficient Fine-Tuning

Full-parameter fine-tuning updates *all* parameters of a pre-trained model on the target dataset, which can be computationally expensive and memory-intensive for large models. Parameter-efficient fine-tuning (PEFT) methods adjust only a subset of the parameters, reducing computational and storage requirements while maintaining performance (Xu et al., 2023).

The most common PEFT method is Low-Rank Adaptation (LoRA), and its variants. They are used in the training process as a wrapper around the model's weights, allowing for efficient updates without modifying the entire model. This approach is particularly useful for large models, where full fine-tuning may be impractical due to resource constraints.

D.3.3.1 Low-Rank Adaptation (LoRA) decomposes the weight updates into low-rank matrices, reducing the number of trainable parameters (Hu et al., 2021). Specifically, for a weight matrix $W \in \mathbb{R}^{d \times k}$, LoRA introduces two low-rank matrices $A \in \mathbb{R}^{d \times r}$ and $B \in \mathbb{R}^{r \times k}$, where $0 < r \ll \min(d, k)$. The adapted weight is:

$$W' = W + \alpha \cdot AB \tag{8}$$

Here, α is a scaling factor that controls the contribution of the low-rank adaptation. The rank r determines the capacity of the adaptation, balancing between expressiveness and efficiency.

LoRA introduces $2dr$ trainable parameters (size of A and B), which is significantly smaller than the original dk parameters. This reduction in parameters enables efficient fine-tuning of large models on limited hardware. In practice, LoRA is applied to specific modules of the model, such as attention and feed-forward layers, to balance performance and efficiency.

D.3.3.2 Rank-Stabilized LoRA (rsLoRA) modifies the scaling factor in LoRA to improve performance across different ranks. The standard scaling factor $\gamma_r = \alpha/r$ can slow learning for higher ranks. rsLoRA proposes adjusting the scaling factor to $\gamma_r = \alpha/\sqrt{r}$, enhancing fine-tuning performance without increasing inference costs.

D.4 Reinforcement Learning (RL)

Reinforcement Learning (RL) is a framework in which an agent interacts with an environment to learn a policy π_θ that maximizes a long-term reward. At each time step t, the agent observes a state, takes an action, and receives a reward r_t. The goal is to maximize the expected cumulative reward, given by

$$J(\theta) = \mathbb{E}_{\pi_\theta} \left(\sum_{t=0}^{T} \gamma^t \, r_t \right) \tag{9}$$

where $\gamma \in (0, 1)$ is a discount factor.

D.4.1 Group Relative Policy Optimization

Group Relative Policy Optimization (GRPO) DeepSeek-AI et al. (2025a) is an online reinforcement learning method specifically designed for scenarios where the model generates multiple responses (or completions) for the same prompt. It was introduced to improve the mathematical reasoning capabilities of LLMs, by generating multiple CoT responses for a given problem and then compares results to the ground truth.

Intuitively, GRPO generates multiple responses for a given prompt, scores them using reward models,

calculates the relative reward of the group, and then compares each response's score to that relative reward to determine which is better or worse. The model then updates its policy to favor high-reward responses.

D.4.1.1 Generating completions For each prompt q in a batch, the model generates G completions:

$$O_q = \{o_1, o_2, \ldots, o_G\} \tag{10}$$

Each completion o_i consists of a sequence of tokens:

$$o_i = \{o_{i,1}, o_{i,2}, \ldots, o_{i,|o_i|}\} \tag{11}$$

D.4.1.2 Computing the advantage For each completion, a reward r_i is computed using predefined reward functions. To enable comparison within groups, the rewards are normalized:

$$\mu_r = \text{mean}(r) \tag{12}$$

$$\sigma_r = \text{std}(r) \tag{13}$$

$$\hat{A}_{i,t} = \frac{r_i - \mu_r}{\sigma_r} \tag{14}$$

where $r = \{r_1, r_2, \ldots, r_G\}$ is the set of rewards for all completions in the group, and $\hat{A}_{i,t}$ is the advantage for token t in completion i. This normalization gives the method its name: Group Relative Policy Optimization.

D.4.1.3 Estimating the KL divergence To prevent the policy from deviating too far from the reference policy π_{ref}, the KL divergence is estimated:

$$\pi_{\text{ratio}} = \frac{\pi_\theta(o_{i,t}|q, o_{i,<t})}{\pi_{\text{ref}}(o_{i,t}|q, o_{i,<t})} \tag{15}$$

$$\pi_{\text{inv\_ratio}} = \frac{\pi_{\text{ref}}(o_{i,t}|q, o_{i,<t})}{\pi_\theta(o_{i,t}|q, o_{i,<t})} \tag{16}$$

$$D_{\text{KL}} = \log \pi_{\text{ratio}} - 1 + \pi_{\text{inv\_ratio}} \tag{17}$$

D.4.1.4 Computing the loss The GRPO objective combines the advantage term with a KL penalty:

$$L_{\text{adv}} = -\frac{1}{G} \sum_{i=1}^{G} \sum_{t=1}^{|o_i|} \pi_{\text{ratio}} \hat{A}_{i,t} \tag{18}$$

$$L_{\text{KL}} = \beta D_{\text{KL}} \tag{19}$$

$$L_{\text{GRPO}}(\theta) = L_{\text{adv}} - L_{\text{KL}} \tag{20}$$

where β is a hyperparameter that controls the weight of the KL penalty. The advantage term encourages the policy to assign higher probability to tokens that lead to better rewards, while the KL term ensures that the policy doesn't deviate too far from the reference policy.

D.4.1.5 Multiple updates For multiple μ updates after each generation, GRPO uses a clipped surrogate objective. First, compute the old policy ratio:

$$\pi_{\text{old\_ratio}} = \frac{\pi_\theta(o_{i,t} \mid q, o_{i,<t})}{\pi_{\theta_{\text{old}}}(o_{i,t} \mid q, o_{i,<t})}, \tag{21}$$

then clip it:

$$\pi_{\text{clipped}} = \text{clip}\Big(\pi_{\text{old\_ratio}},\, 1-\epsilon,\, 1+\epsilon\Big). \tag{22}$$

The clipped advantage loss is $L_{\text{adv\_clipped}}$

$$-\frac{1}{G} \sum_{i=1}^{G} \sum_{t=1}^{|o_i|} \min\Big(\pi_{\text{old\_ratio}} \hat{A}_{i,t},\, \pi_{\text{clipped}} \hat{A}_{i,t}\Big), \tag{23}$$

yielding the final objective:

$$L_{\text{GRPO\_clipped}}(\theta) = L_{\text{adv\_clipped}} - L_{\text{KL}}. \tag{24}$$

Here, ϵ (small constant, typically 0.2) controls how much the policy can change in a single update and β controls the KL penalty's strength.
In HuggingFace's `trl` library, GRPO is implemented in the `GRPOTrainer` class and number of updates μ is controlled by the `num_iterations` parameter. The default value of $\mu = 1$ simplifies the objective to the original GRPO formulation.

E Training details

E.1 Environment setup

The training was conducted, alternately, on a NVIDIA Tesla T4 GPU provided for free by Google Cloud (through Google Colab, Kaggle Notebook, or Google Cloud's Deep Learning Virtual Machine image) and a H100 NVIDIA GPU server with RunPod [9] (paid by the author, for detailed costs, see Appendix F). The T4 GPU has 16GB of memory, while the H100 GPU has 80GB of memory. The T4 GPU was used for initial experiments and supervised fine-tuning, while the H100 GPU was employed for more extensive training runs and reinforcement learning.
Training environment was set up using these HuggingFace's libraries: `bitsandbytes` for quantization, `peft` for parameter-efficient fine-tuning,

[9] https://www.runpod.io/

`transformers` for model management. The training process was executed using the `trl` library, which provides tools for pre-training and post-training with transformers (including GRPO). `unsloth` was used to reduce memory usage on single-GPU environment. `vllm` was used for fast inference and serving of the trained model, especially during GRPO.

The base student model used is Gemma-3-1b-it, an open-weight 1-billion parameter Transformer-based decoder-only text-to-text model pre-trained to work well on general-purpose tasks in multiple languages, and fine-tuned to increase instruction following capabilities. To save memory, a 4-bit quantized version of the model was used, which reduces the model size and speeds up inference without significantly sacrificing performance.

E.2 LoRA configuration

To reduce computational overhead, we employed LoRA (Hu et al., 2021) and rank-Stabilized LoRA (rsLoRA) scaling. The LoRA configuration parameters were set as follows: rank $r = 8$, scaling factor $\alpha_{\text{LoRA}} = 16$, and dropout rate of 0. These configurations were applied to both the attention and feed-forward layers.

The rank r determines the dimensionality of the low-rank adaptation matrices, controlling the number of trainable parameters introduced during fine-tuning. A higher rank allows the model to capture more complex adaptations but increases computational complexity. The scaling factor α_{LoRA} modulates the impact of the low-rank updates on the original weights, effectively controlling the contribution of the adaptation matrices to the final model parameters. Setting the dropout rate to 0 indicates that no dropout regularization was applied during the LoRA updates, allowing all connections to be utilized during training.

E.3 GRPO configuration

We implemented GRPO using the `trl` library, which provides a convenient interface for training language models with reinforcement learning. Our specific implementation used three reward functions (explained in the main paper): 1) outputs that follow the required format with reasoning, mnemonic, and example sections, 2) explicitly incorporate linguistic features from our taxonomy, 3) meaningfully use the target vocabulary in the mnemonic while penalizing acronyms. These reward functions were combined with weights [1.0, 1.5, 1.0] respectively, placing greater emphasis on linguistic grounding. The model generated two completions ($G = 2$) per prompt, allowing for relative comparison. The KL penalty coefficient β was set to 0.04, and we used a single iteration ($\mu = 1$) per batch.

We used a batch size of 16, a learning rate of 2×10^{-5}, and a weight decay of 0.05. The training process was monitored using the validation set, and early stopping was applied to prevent overfitting. The training process was conducted over 3 epochs, with a total of 2000 training examples. The model was trained using the paged AdamW optimizer, which is a variant of the AdamW optimizer designed to handle large models efficiently. The training process was distributed across multiple GPUs using the `accelerate` library, which allows for efficient parallelization and memory management.

F Costs

G Documentation of previous iterations

G.1 Fine-tune OpenAI

G.2 Fine-tune Gemma-3-4b-it

For supervised fine-tuning (SFT), we utilized the `trl` library with the following hyperparameters: batch size $b = 16$, number of epochs eps $= 4$, learning rate $\alpha = 2 \times 10^{-5}$, weight decay $\lambda = 0.05$, and a cosine annealing learning rate scheduler with restarts.
The batch size b defines the number of training examples processed simultaneously during each forward and backward pass. A batch size of 16 balances computational efficiency and gradient estimation accuracy. Training for 4 epochs (eps $= 4$) means the model will see the training data a total of four times, which ensures sufficient exposure to the training data without risking overfitting. The learning rate α controls the step size for weight updates; a value of 2×10^{-5} is typical for fine-tuning large language models, facilitating gradual convergence. Weight decay λ serves as a regularization term, penalizing large weights to prevent overfitting. The cosine annealing scheduler adjusts the learning rate following a cosine decay pattern, periodically restarting to allow the model to escape local minima and potentially achieve better generalization, compared to linear decay.

H Reflection

I Minerva Appendix: LOs & HCs

I.1 LOs

1. **CS110-codeReadability** The codebase exemplifies best practices in Python programming, adhering strictly to PEP conventions. Each module is documented with detailed Google-style docstrings and descriptive inline comments to ensure that the logic behind functions and classes is transparent to collaborators and future users. By utilizing tools like Ruff for linting and formatting, and mypy for type-checking, the codebase achieves a consistent style and minimizes errors. Additionally, the inclusion of pre-commit hooks ensures that these standards are maintained across all contributions, fostering a robust and maintainable codebase.
2. **CS162-communication** The documentation strives to adhere to industry standards, by including an informative README, clear commit messages (using conventional commit or gitmoji), and documented pull request. Each module has a module-level docstring and function-level docstrings, to ensure that the code is understandable to users and collaborators. I used `Ruff` (for linting and black-style code formatting) and `mypy` (for type checking), to enforce consistent and error-free code presentation, to facilitate readability and maintainability for the author and future collaborators (if any).
3. **CS156-MLCode** The machine learning pipeline was designed in Python to be both functional and comprehensible. The code integrates data processing, model fine-tuning, and hyperparameter tuning into a seamless pipeline, with each step working and documented. Model evaluation are explicitly defined in appendix Appendix E, ensuring replicability.
4. **CS156-MLExplanation** Currently, most details and explanations are defined in section Appendix D and appendix Appendix E. The final paper will provide more high-level diagrams of the machine learning techniques used in fine-tuning the Gemma-2 model. The supporting diagrams will visualize key processes, such as hyperparameter tuning and model evaluation. It will ensure that the methodology and results are accessible to a less specialized audience.
5. **CS162-separationofconcerns** The codebase is organized into distinct Python modules, each focused on a specific task such as data processing, mnemonic processing, and model fine-tuning. This separation of concerns aligns with best practices in software design, ensuring that each function is highly cohesive and performs a single well-defined responsibility. By maintaining modularity, the codebase facilitates easier debugging, testing, and future scaling, contributing to its long-term maintainability and effectiveness.

I.2 Capstone LOs

6. **navigation** Appendix G

7. **outcomeanalysis**
8. **curation**
9. **qualitydeliverables**

I.3 HCs

10. **audience** The project addresses dual audiences: (1) NLP researchers investigating LLMs' linguistic capabilities, for whom we provide detailed technical methodologies and evaluation metrics; and (2) educational technology developers seeking practical approaches to vocabulary learning assistance, for whom we demonstrate application potential and implementation strategies. For example, Section 2.1 introduces key concepts about mnemonic devices and language teaching with sufficient depth for both audiences, while hiding technical details in appendices but providing pointers to those details in each section for interested readers. The paper's structure and content are designed to facilitate knowledge transfer across these domains, ensuring that both audiences can derive value from the research findings. Technical content is balanced with practical implications for vocabulary acquisition and language learning, ensuring relevance to both research and application-oriented readers.
11. **organization** The paper follows standard ACL formatting conventions, with a logical progression from theoretical foundations through methodology to empirical results. The structure facilitates efficient information extraction, with each section building upon previous content. Key contributions are identified early (Section 1) and systematically developed throughout subsequent sections. Technical details that might interrupt argumentative flow are relegated to appendices, maintaining narrative coherence while ensuring methodological transparency for reproduction purposes. This organization aligns with expectations of the computational linguistics community while supporting efficient knowledge transfer.
12. **gapanalysis** Section 1 identifies critical limitations in existing mnemonic generation approaches: (1) overreliance on the keyword method, which fails for abstract vocabulary lacking concrete referents, and (2) neglect of the rich linguistic knowledge embedded in LLMs that could enable more diverse mnemonic strategies beyond simple keyword associations. Prior work has also passively delivered mnemonics to learners rather than leveraging individual learning preferences, despite research showing self-created mnemonics enhance retention. These identified gaps motivate our project to elicit linguistic reasoning and creativity from LLMs, enabling them to generate mnemonics that are not only effective but also linguistically grounded. Our final model, **LINKSYS**, after deployment, could interact with learners to tailor mnemonic generation based on their preferences, enhancing the learning experience. This approach addresses the limitations of existing methods by providing a more comprehensive and personalized vocabulary learning tool. The project also contributes to the field of educational technology by exploring how LLMs can be harnessed for creating effective resources for language education and self-study, such as creating mnemonic devices.
13. **hypothesisdevelopment** The research questions in Section 1 establish testable hypotheses regarding LLMs as linguistic knowledge bases for mnemonic generation. We hypothesize that fine-tuning LLMs on linguistically annotated examples will improve mnemonic quality across semantic relevance, diversity, and helpfulness dimensions. This hypothesis is predicated on the theoretical assumption that LLMs encode significant linguistic knowledge during pre-training that can be accessed through targeted fine-tuning. The hypothesis is operationalized through specific evaluation metrics described in Section 5, ensuring empirical testability.
14. **scienceoflearning**
15. **optimization** ...
16. **algorithms** The fine-tuning methodology detailed in Appendix D.3 incorporates algorithmic innovations in parameter-efficient adaptation. Specifically, we implement QLoRA with rank-stabilized scaling (rsLoRA), modifying the standard scaling factor to improve performance across different ranks. This algorithm reduces memory requirements by quantizing the pre-trained model to 4-bit precision while allowing selective updates to low-rank adaptation matrices. Population-based training explores the hyperparameter space dynamically, pruning underperforming configurations and exploring promising regions to optimize validation performance.

17. **heuristics** Our approach employs several problem-solving heuristics to enhance mnemonic generation. The linguistic feature classification system provides a structured framework for identifying and leveraging different linguistic aspects in vocabulary. We use problem decomposition by separating mnemonic creation into linguistic analysis and creative association phases, enabling more systematic knowledge utilization. Visualization techniques in the form of embedding space projections help identify semantic relationships between vocabulary terms and potential mnemonic content. These heuristics guide both the model fine-tuning process and the subsequent evaluation methodology.
18. **sampling** The evaluation methodology employs stratified random sampling to ensure representation across linguistic feature categories (Section 4.1) and vocabulary complexity levels. For human evaluation, we randomly selected 50 test examples, stratified by linguistic feature, to obtain less biased assessments of mnemonic quality. This sampling strategy ensures balanced representation of different mnemonic types while maintaining statistical validity. Each example received multiple independent ratings to mitigate individual rater bias, with inter-rater reliability calculated using Cohen's Kappa score to confirm consistency in ratings.
19. **dataviz** We employ targeted data visualizations to communicate complex relationships between linguistic features and mnemonic effectiveness. Radar charts display the distribution of linguistic features across different model outputs, while heatmaps visualize correlation patterns between computational metrics and human evaluations. Embedding space projections illustrate semantic relationships between vocabulary terms and their mnemonics, providing intuitive visual confirmation of semantic relevance scores. These visualizations enhance interpretability of results while supporting our conclusions regarding the contribution of different linguistic features to mnemonic quality.
20. **significance** Statistical significance testing (Section 5.1) confirms the reliability of our comparative results between baseline and **LINKSYS**. We employ paired statistical tests (Wilcoxon signed-rank and McNemar test) to account for vocabulary-specific variation when comparing model outputs on identical test sets. Effect size calculations quantify the practical significance of improvements, while confidence intervals provide transparency regarding the precision of our estimates. Multiple comparison corrections maintain statistical rigor when evaluating performance across different linguistic feature categories.
21. **shapingbehavior** The intended application of our approach shapes vocabulary learning behavior by encouraging deeper engagement with linguistic features. Rather than keyword method, the generated mnemonics prompt learners to recognize morphological, etymological, and semantic patterns, fostering more robust mental representations. By explicitly highlighting these linguistic features, the system promotes analytical processing of vocabulary, which research indicates enhances long-term retention. This behavior-shaping aspect represents a significant advantage over keyword-only approaches that rely on shallow phonetic or orthographic associations.
22. **biasmitigation**
23. **ethicalconsiderations** Our work addresses ethical considerations in educational technology deployment. We explore linguistic diversity by analyzing and leveraging several linguistic features for mnemonic generation, ensuring that the generated mnemonics are inclusive and accessible to learners from diverse linguistic backgrounds. The model is trained on a diverse dataset of vocabulary words, including those from various languages and etymological origins, to ensure that the mnemonics generated are relevant and culturally sensitive. We also consider the potential for bias in the generated mnemonics by ensuring that the training data is representative of a wide range of vocabulary words.
24. **studyreplication** To facilitate replication, we provide comprehensive implementation details in Appendix E, including environment setup, model parameters, and training configurations. The dataset construction process is documented in Section 4.2, with preprocessing steps explicitly specified. All code and datasets are made publicly available through GitHub and HuggingFace repositories, with standardized formats ensuring compatibility with common ML frameworks. This transparency ensures that other researchers can validate the findings and build upon the methodology.