AdaptMI: Adaptive Skill-based In-context Math Instructions for Small Language Models

Yinghui He [1] **Abhishek Panigrahi** [1] **Yong Lin** [1] **Sanjeev Arora** [1]

Abstract

In-context learning (ICL) allows a language model to improve its problem-solving capability when provided with suitable information in context. Since the choice of in-context information can be determined based on the problem itself, in-context learning is analogous to human learning from teachers in a classroom. Recent works (Didolkar et al., 2024a;b) show that ICL performance can be improved by leveraging a frontier large language model's (LLM) ability to predict required *skills* to solve a problem, popularly referred to as an *LLM's metacognition*, and using the recommended skills to construct necessary in-context examples. While this skill-based strategy boosts ICL performance in larger models, its gains on small language models (SLMs) have been minimal, highlighting a performance gap in ICL capabilities.

We investigate this gap and show that skill-based prompting can hurt SLM performance on *easy* questions by introducing unnecessary information, akin to cognitive overload. To address this, we introduce AdaptMI, an **Adapt**ive approach to selecting skill-based in-context **M**ath **I**nstructions for SLMs. Inspired by cognitive load theory from human pedagogy, our method only introduces skill-based examples when the model performs poorly. We further propose AdaptMI+, which adds examples targeted to the specific skills missing from the model's responses. On 5-shot evaluations across popular math benchmarks and five SLMs (1B–7B; Qwen, Llama), AdaptMI+ improves accuracy by up to 6% over naive skill-based strategies.

[1]Princeton Language and Intelligence, Princeton University. Correspondence to: Yinghui He <yh0068@princeton.edu>.

Proceedings of the 42nd International Conference on Machine Learning, Vancouver, Canada. PMLR 267, 2025. Copyright 2025 by the author(s).

1. Introduction

Human learning is driven by adaptive feedback (Hattie & Timperley, 2007; Bandura & Walters, 1977), often through targeted examples in a classroom setting. Analogously, in-context learning (ICL) (Brown et al., 2020) enables language models (Vaswani et al., 2017; Achiam et al., 2023; Team et al., 2023; Grattafiori et al., 2024) to adapt to new tasks by conditioning on additional task-relevant information, potentially provided by a stronger model acting as a teacher.

While ICL emerges naturally in large models (Wei et al., 2022), small language models (SLMs) struggle. Their performance is highly sensitive to the choice of context, limiting their ability to learn from in-context instructions. This paper investigates how to improve SLMs' ICL performance using careful in-context example selection for math tasks.

We build on skill-based in-context example selection from Didolkar et al. (2024a;b), which leverages the metacognitive abilities of frontier large language models (LLMs) to predict the high-level skills required to solve a given task. After annotating a pool of examples with the skill labels, in-context examples are selected at inference time by first predicting the required skills and then retrieving matching examples. While skill-based in-context selection significantly boosts the ICL performance of larger models, it fails to improve ICL performance in SLMs.

Ablation reveals an important insight: Skill-based strategy can hurt the performance of an SLM on *easy* questions, those that an SLM can already solve without skill-based guidance. Across 5 SLMs on the MATH dataset (Hendrycks et al., 2021), we observe an average 4% performance drop on *easy* questions when using skill-based selection, compared against non skill-based in-context selection strategies.

Core Contribution: Motivated by Adaptive Teaching (Randi, 2022) and Cognitive Load Theory (Sweller, 2011), which emphasizes effective human learning with targeted guidance on challenging tasks, we propose AdaptMI. It is a two-stage in-context selection method that applies skill-based example selection only to *difficult* questions. To further align feedback with model errors, we introduce AdaptMI+, which selects examples based on skills missing from the SLM's response. An overview is shown in

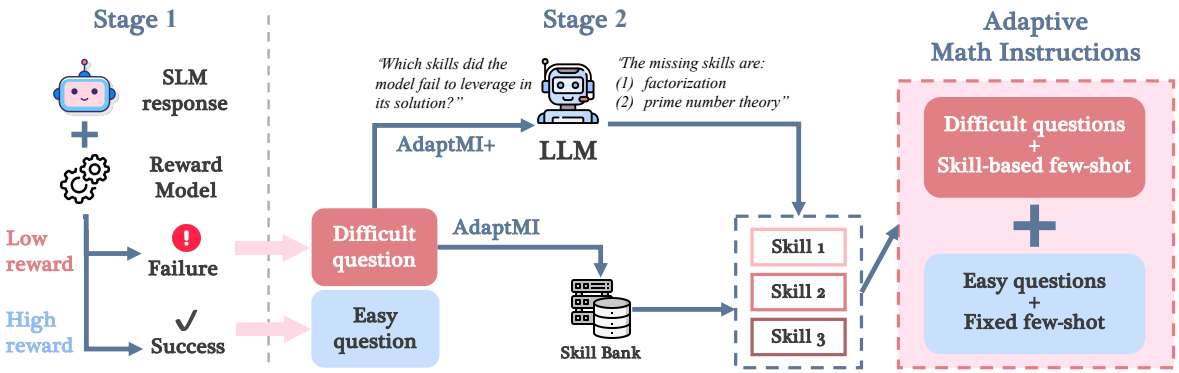

Figure 1. AdaptMI and AdaptMI+ are 2-stage adaptive in-context example selection methods. In the first stage, questions are classified as *easy* and *difficult* using a reward model on the SLM's responses and a threshold-based filtering. In the second stage, AdaptMI uses skill-based in-context examples only for *difficult* questions. For AdaptMI+, we use an LLM to identify the key skills missing in the SLM's responses for *difficult* questions and use specific in-context examples targeted towards the missing skills. For *easy* questions, we use a fixed set of in-context examples. We follow (Didolkar et al., 2024a) to get the Skill Bank, skill annotations for each question, and relevant in-context examples for each skill.

Figure 1, with full details in Sections 2.2 and 2.3.

Experiments on standard math datasets (Section 3) show that AdaptMI+ improves all five SLMs by up to 6%, with AdaptMI achieving gains of up to 3.6%. We also explore iterative extensions for progressively training SLMs on harder problems and provide a detailed analysis showing why adaptive selection outperforms naive skill-based methods.

2. Designing AdaptMI and AdaptMI+

2.1. Preliminary

We study k-shot in-context learning with small language models. Let $\mathcal{Q}$ be the set of evaluation questions, and let $q \sim \mathcal{Q}$ denote a question drawn from this set. In k-shot setting, given a pool $\mathcal{P}$ of question-answer pairs, k examples are selected and included in the prompt for each evaluation question q. There are two common strategies to select the in-context example pairs.

♦ **Fixed k-shot examples**: We fix a set of k examples from $\mathcal{P}$ and use them for inference on all evaluation questions. Our experiments will use the examples used by Qwen models for evaluation (Yang et al., 2024).

♠ **Random k-shot examples**: We utilize k randomly selected examples from $\mathcal{P}$ for each evaluation question.

Our work builds on skill-based in-context selection from (Didolkar et al., 2024a), which we describe here. The strategy builds Skill-bank$(\mathcal{Q})$ as a set of skills that are necessary to solve the questions, enlisted from a large model like GPT-4 using an appropriate prompting strategy. Then, a map Skill-Map : $\mathcal{Q} \cup \mathcal{P} \rightarrow$ Skill-bank$(\mathcal{Q})^k$ is defined between each question and a set of k skills, which we will also get by prompting the same LLM. Then:

♥ **Skill-based k-shot examples**: For each question $q \in \mathcal{Q}$, we pick a set of k examples using Skill-Map(q), by randomly picking one example for each skill in Skill-Map(q). This is formally outlined in Algorithm 1 in appendix.

AdaptMI and AdaptMI+ are built on the above-defined strategies, and consist of 2 primary stages.

2.2. Stage 1: Detection of *easy* and *difficult* questions

While one could simply define *difficult* questions as those set of questions that the model gets wrong with fixed or random k-shot prompting, this requires access to the ground truth labels. Instead, to make our technique more broadly applicable, we use a process reward model to score the responses of the SLM. We use a thresholding function to separate *easy* and *difficult* questions, which we refer as Q_{easy} and $Q_{\text{difficult}}$ respectively. We defer the details to Appendix B.

2.3. Stage 2: Skill-based selection of in-context examples

Now, we build AdaptMI and AdaptMI+ as follows.

• **AdaptMI**: For *difficult* questions $Q_{\text{difficult}}$, we use skill-based k-shot examples. For *easy* questions Q_{easy}, we use fixed k-shot examples.

• **AdaptMI+**: For each *difficult* question q in $Q_{\text{difficult}}$, we use a large LLM (GPT-4o-mini) to predict the set of skills in Skill-Map(q) that are missing in the model's response. Then, for each skill s that are missing, we randomly pick an example from the pool of in-context examples $\mathcal{P}$ which is annotated with the skill s and return the union of the selected examples for all the missing skills. For *easy* questions Q_{easy}, we use fixed k-shot examples.

Methods	MATH								GSM8K
	Geometry	Precalculus	Algebra	Prealgebra	Number Theory	Intermediate Algebra	Counting & Probability	**Avg.**	**Avg.**
# Qwen2.5-3B-Instruct									
Fixed Examples	56.4	53.5	85.4	79.7	65.9	46.8	59.5	66.6	84.7
Random Examples	54.7	53.7	85.3	78.9	64.1	46.7	60.1	66.1	84.9
Skill-based Examples	53.4	55.7	86.2	80.7	66.1	45.9	60.3	66.9	85.4
Consistency@5	**61.9**	55.3	87.4	81.4	66.5	49.4	61.7	68.9	87.0
AdaptMI	54.9	**56.2**	87.7	81.8	66.7	46.5	60.6	67.8	87.4
AdaptMI+	56.0	55.5	**88.3**	**82.1**	**68.9**	**49.8**	**62.7**	**69.1**	**87.7**
# Llama-3.2-3B-Instruct									
Fixed Examples	26.1	29.8	63.8	67.6	38.7	22.6	42.7	46.2	75.8
Random Examples	34.1	26.9	61.9	55.3	29.3	18.5	33.7	41.3	76.2
Skill-based Examples	29.6	31.7	66.2	63.3	39.6	23.2	33.7	45.9	71.7
Consistency@5	**36.1**	23.9	60.0	61.9	35.0	21.1	**46.7**	44.1	**80.7**
AdaptMI	28.4	31.7	**71.6**	**71.3**	**43.4**	**24.4**	39.3	**49.8**	76.4
AdaptMI+	29.6	**35.6**	68.1	**71.3**	**43.4**	**24.4**	39.3	49.4	**80.7**

Table 1. AdaptMI and AdaptMI+ demonstrate a consistent accuracy gain compared with baseline methods. We present all results as Pass@1 accuracy unless otherwise indicated. Due to space limits, we provide the results on Qwen2.5-1.5B-Instruct, Qwen2.5-7B-Instruct, and Llama-3.2-1B-Instruct in Table 6, Appendix D.1.

3. Experiment

3.1. Experimental Settings

We evaluate our method on MATH (5k test samples) and GSM8K (1.3k test samples) on five small language models with 5-shot prompting: Qwen2.5-1.5B-Instruct, Qwen2.5-3B-Instruct, Qwen2.5-7B-Instruct, Llama-3.2-1B-Instruct, and Llama-3.2-3B-Instruct (Yang et al., 2024; Meta AI, 2024). In Stage 1, we adopt an process reward model RLHFlow/Llama3.1-8B-PRM-Mistral-Data (Xiong et al., 2024) for question classification. More detailed experimental settings are in Appendix C.1.

3.2. Performances of AdaptMI and AdaptMI+

Table 1 and Table 6 (Appendix D.1) report the main results of our adaptive in-context learning method. The baseline methods with non-adaptive in-context examples (fixed, random, or skill-based) results in largely similar Pass@1 accuracy, while Consistency@5 can improve accuracy by a few percentages. Across all model sizes, our methods AdaptMI and AdaptMI+ consistently outperform the non-adaptive Pass@1 baselines, and are on par with Consistency@5 performance on most subareas. The overall improvements are especially pronounced for smaller models, Qwen2.5-1.5B-Instruct and Llama-3.2-1B-Instruct.

Notably, AdaptMI+ brings significant performance gain across all areas by up to 6%, reflecting its strength in accurately targeting model failures. AdaptMI also substantially improves performance by up to 3.6% for Qwen2.5-1.5B-Instruct, Llama-3.2-1B-Instruct, and Llama-3.2-3B-Instruct on MATH. This indicates that our adaptive instruction methods are effective on lower-performing models even without the aid of an LLM.

On stronger models such as Qwen2.5-3B-Instruct and Qwen2.5-7B-Instruct, however, AdaptMI shows smaller effectiveness compared to AdaptMI+. This may suggest that higher-performing models require a more intelligent and target skill identification process. Overall, these results demonstrate the effectiveness of adaptive example selection and highlight the potential of our approach to elicit the full reasoning capabilities of small language models.

We further extend our method to an iterative teaching-refinement loop in Appendix D.2. The iterative AdaptMI+ method shows up to 8% improvement on MATH, demonstrating its potential to progressively improve small language models to tackle harder problems.

3.3. Why is AdaptMI better than non-adaptive example selection?

To better understand, we compare performance under fixed, random, and skill-based in-context examples on *easy* and *difficult* questions. From Table 2, we observe a clear trend that skill-based examples harm an SLM's performance on the set of *easy* questions, while effectively boosting performance on the *difficult* ones. To gain deeper insight into how skill-based in-context examples might harm performance on *easy* questions, we present two illustrative cases where the

Question & Example		MATH								GSM8K
		Geometry	Precalculus	Algebra	Prealgebra	Number Theory	Intermediate Algebra	Counting & Probability	**Avg.**	**Avg.**
Diff-icult	Fixed	21.3	23.7	44.8	35.1	24.1	27.0	28.2	29.8	45.2
	Random	23.2	25.3	53.9	40.5	21.9	23.0	27.2	31.2	46.1
	Skill	28.4	28.9	55.1	45.5	31.2	27.4	32.1	35.7	48.0
		+7.1	**+5.2**	**+10.3**	**+10.4**	**+7.1**	**+0.4**	**+3.9**	**+5.9**	**+2.8**
Easy	Fixed	82.1	81.8	94.6	93.7	84.6	80.7	92.2	90.2	96.3
	Random	81.6	78.9	92.1	92.3	80.1	75.7	88.1	87.6	90.6
	Skill	77.2	71.5	85.9	86.0	71.8	74.5	74.5	81.0	83.2
		-4.9	**-10.3**	**-8.7**	**-7.7**	**-12.8**	**-6.2**	**-17.7**	**-9.2**	**-13.1**

Table 2. Accuracy of Qwen2.5-1.5B-Instruct on *difficult* and *easy* questions, respectively under fixed, random, and skill-based examples. Skill-based examples boost performance on *difficult* questions across all categories, while significantly underperforming on *easy* questions. We provide the results on other Qwen models in Table 7, Appendix D.3.

model's performance regresses when using such prompts.

Case Study 1: Skill-based examples lead the model to overlook key problem constraints. In this example (see Appendix G.1), Qwen2.5-7B-Instruct is given an algebra question that includes geometric constraints. With fixed examples, the model correctly uses the condition "both coordinates are negative". However, when prompted with algebra-focused skill examples, it overemphasizes algebraic procedures and overlooks the constraint, ultimately answering incorrectly.

Case Study 2: Symbol-heavy skill-based examples cause the model to overthink. This question (see Appendix G.2) is best solved via plug-in-and-test. With fixed examples, the model tries out small values and succeeds. In contrast, skill-based examples bias the model toward equation solving; after an initial failed plug-in, it shifts to an unnecessary algebraic approach and fails.

To quantify where and why overthinking happens, we conduct a fine-grained analysis in Appendix E that splits the MATH questions into 5 difficulty levels and compare models' accuracy and average output length on those splits. We observe that skill-based in-context examples encourage longer responses on all difficulty levels (see Figure 3). This effect may push the models to tackle harder questions (levels 3–5), but can also elicit overthinking and hurt performance on easier questions (levels 1–2). This aligns with recent findings that longer chain-of-thought from *overthinking* can degrade SLM performance (Liu et al., 2024b) on easy questions, mirroring similar effects in humans under information overload (Diaconis & Mazur, 2003).

Additional ablations: In Appendix F, we also present:

- We ablate on different combinations of in-context example types for Stage 2. Our combination of "difficult + skill-based; easy + fixed" consistently outperforms all other combinations in Figure 5.
- We compare natural language instructions versus few-shot instructions. While AdaptMI demonstrates the power of targeted few-shot supervision, Table 9 shows that small language models do not benefit if the additional supervision was provided in written in natural language.

4. Related Works

An extended discussion is provided in Appendix A. We review prior in-context example selection methods (Zhang et al., 2022; Cheng et al., 2023), which typically rely on semantic similarity between the question and retrieved examples. We also review adaptive dataset construction approaches (Dinan et al., 2019; Nie et al., 2020), which focus on supervised fine-tuning. Our work bridges these two directions by exploring how adaptive, skill-based in-context selection can further enhance the performance of small language models.

5. Conclusion

Our work explores reasons behind the failure of skill-based in-context examples to boost ICL performance of SLMs. We show that skill-based selection can make the model "overthink" on easier questions, which leads to a degradation in ICL performance. We then propose adaptive in-context selection strategies, AdaptMI and AdaptMI+, that use skill-based selection only for *difficult* questions.

Beyond inference-time gains, an open question is whether such adaptive strategies can also improve SLM training. While existing methods distill from frontier models using static prompts or curated data (Hsieh et al., 2023; Ivison et al., 2023; Kaur et al., 2024; Zhu et al., 2025; Gao et al., 2025; Liao et al., 2024; Allen-Zhu & Li, 2024), our approach offers a promising direction for incorporating dynamic, difficulty-aware supervision in training pipelines.

References

Achiam, J., Adler, S., Agarwal, S., Ahmad, L., Akkaya, I., Aleman, F. L., Almeida, D., Altenschmidt, J., Altman, S., Anadkat, S., et al. Gpt-4 technical report. *arXiv preprint arXiv:2303.08774*, 2023.

Allen-Zhu, Z. and Li, Y. Physics of language models: Part 3.3, knowledge capacity scaling laws. *arXiv preprint arXiv:2404.05405*, 2024.

Alotaibi, F., Kulkarni, A., and Zhou, D. Graph of logic: Enhancing llm reasoning with graphs and symbolic logic. In *2024 IEEE International Conference on Big Data (BigData)*, pp. 5926–5935. IEEE, 2024.

An, S., Zhou, B., Lin, Z., Fu, Q., Chen, B., Zheng, N., Chen, W., and Lou, J.-G. Skill-based few-shot selection for in-context learning, 2023. URL https://arxiv.org/abs/2305.14210.

Bandura, A. and Walters, R. H. *Social learning theory*, volume 1. Prentice hall Englewood Cliffs, NJ, 1977.

Brown, T. B., Mann, B., Ryder, N., Subbiah, M., Kaplan, J., Dhariwal, P., Neelakantan, A., Shyam, P., Sastry, G., Askell, A., Agarwal, S., Herbert-Voss, A., Krueger, G., Henighan, T., Child, R., Ramesh, A., Ziegler, D. M., Wu, J., Winter, C., Hesse, C., Chen, M., Sigler, E., Litwin, M., Gray, S., Chess, B., Clark, J., Berner, C., McCandlish, S., Radford, A., Sutskever, I., and Amodei, D. Language models are few-shot learners, 2020. URL https://arxiv.org/abs/2005.14165.

Chen, J. C.-Y., Yun, S., Stengel-Eskin, E., Chen, T., and Bansal, M. Symbolic mixture-of-experts: Adaptive skill-based routing for heterogeneous reasoning, 2025. URL https://arxiv.org/abs/2503.05641.

Chen, Y., Zhao, C., Yu, Z., McKeown, K., and He, H. On the relation between sensitivity and accuracy in in-context learning, 2024. URL https://arxiv.org/abs/2209.07661.

Cheng, D., Huang, S., Bi, J., Zhan, Y., Liu, J., Wang, Y., Sun, H., Wei, F., Deng, D., and Zhang, Q. Uprise: Universal prompt retrieval for improving zero-shot evaluation, 2023. URL https://arxiv.org/abs/2303.08518.

Cobbe, K., Kosaraju, V., Bavarian, M., Chen, M., Jun, H., Kaiser, L., Plappert, M., Tworek, J., Hilton, J., Nakano, R., et al. Training verifiers to solve math word problems. *arXiv preprint arXiv:2110.14168*, 2021.

Diaconis, P. and Mazur, B. C. The problem of thinking too much. *Bulletin of the American Academy of Arts and Sciences*, 56(3):26–38, 2003.

Didolkar, A., Goyal, A., Ke, N. R., Guo, S., Valko, M., Lillicrap, T., Jimenez Rezende, D., Bengio, Y., Mozer, M. C., and Arora, S. Metacognitive capabilities of llms: An exploration in mathematical problem solving. *Advances in Neural Information Processing Systems*, 37: 19783–19812, 2024a.

Didolkar, A., Goyal, A., Ke, N. R., Guo, S., Valko, M., Lillicrap, T., Rezende, D., Bengio, Y., Mozer, M., and Arora, S. Metacognitive capabilities of llms: An exploration in mathematical problem solving, 2024b. URL https://arxiv.org/abs/2405.12205.

Dinan, E., Humeau, S., Chintagunta, B., and Weston, J. Build it break it fix it for dialogue safety: Robustness from adversarial human attack. In Inui, K., Jiang, J., Ng, V., and Wan, X. (eds.), *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pp. 4537–4546, Hong Kong, China, November 2019. Association for Computational Linguistics. doi: 10.18653/v1/D19-1461. URL https://aclanthology.org/D19-1461/.

Gao, I., Ilharco, G., Lundberg, S., and Ribeiro, M. T. Adaptive testing of computer vision models, 2023. URL https://arxiv.org/abs/2212.02774.

Gao, T., Wettig, A., He, L., Dong, Y., Malladi, S., and Chen, D. Metadata conditioning accelerates language model pre-training. *arXiv preprint arXiv:2501.01956*, 2025.

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.

Gui, L., Gârbacea, C., and Veitch, V. Bonbon alignment for large language models and the sweetness of best-of-n sampling. *arXiv preprint arXiv:2406.00832*, 2024.

Hattie, J. and Timperley, H. The power of feedback. *Review of educational research*, 77(1):81–112, 2007.

Hendrycks, D., Burns, C., Kadavath, S., Arora, A., Basart, S., Tang, E., Song, D., and Steinhardt, J. Measuring mathematical problem solving with the math dataset. *arXiv preprint arXiv:2103.03874*, 2021.

Hsieh, C.-Y., Li, C.-L., Yeh, C.-K., Nakhost, H., Fujii, Y., Ratner, A., Krishna, R., Lee, C.-Y., and Pfister, T. Distilling step-by-step! outperforming larger language models with less training data and smaller model sizes. *arXiv preprint arXiv:2305.02301*, 2023.

Ivison, H., Wang, Y., Pyatkin, V., Lambert, N., Peters, M., Dasigi, P., Jang, J., Wadden, D., Smith, N. A., Beltagy, I., et al. Camels in a changing climate: Enhancing lm adaptation with tulu 2. *arXiv preprint arXiv:2311.10702*, 2023.

Kaur, S., Park, S., Goyal, A., and Arora, S. Instruct-skillmix: A powerful pipeline for llm instruction tuning. *arXiv preprint arXiv:2408.14774*, 2024.

Li, X. L., Kaiyom, F., Liu, E. Z., Mai, Y., Liang, P., and Hashimoto, T. Autobencher: Towards declarative benchmark construction, 2025. URL https://arxiv.org/abs/2407.08351.

Liao, H., He, S., Hao, Y., Li, X., Zhang, Y., Zhao, J., and Liu, K. *SKIntern*: Internalizing symbolic knowledge for distilling better cot capabilities into small language models, 2024. URL https://arxiv.org/abs/2409.13183.

Liu, H., Liu, J., Huang, S., Zhan, Y., Sun, H., Deng, W., Wei, F., and Zhang, Q. se^2: Sequential example selection for in-context learning, 2024a. URL https://arxiv.org/abs/2402.13874.

Liu, R., Geng, J., Wu, A. J., Sucholutsky, I., Lombrozo, T., and Griffiths, T. L. Mind your step (by step): Chain-of-thought can reduce performance on tasks where thinking makes humans worse. *arXiv preprint arXiv:2410.21333*, 2024b.

Meta AI. Llama 3.2: Revolutionizing Edge AI and Vision with Open, Customizable Models, 2024. URL https://ai.meta.com/blog/llama-3-2-connect-2024-vision-edge-mobile-devices/.

Nie, Y., Williams, A., Dinan, E., Bansal, M., Weston, J., and Kiela, D. Adversarial NLI: A new benchmark for natural language understanding. In Jurafsky, D., Chai, J., Schluter, N., and Tetreault, J. (eds.), *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pp. 4885–4901, Online, July 2020. Association for Computational Linguistics. doi: 10.18653/v1/2020.acl-main.441. URL https://aclanthology.org/2020.acl-main.441/.

OpenAI. Gpt-4o mini: advancing cost-efficient intelligence. https://openai.com/index/gpt-4o-mini-advancing-cost-efficient-intelligence/, 2024.

Randi, J. Adaptive teaching. In *Routledge encyclopedia of education, educational psychology*. Routledge, 2022.

Ribeiro, M. T. and Lundberg, S. Adaptive testing and debugging of NLP models. In Muresan, S., Nakov, P., and Villavicencio, A. (eds.), *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 3253–3267, Dublin, Ireland, May 2022. Association for Computational Linguistics. doi: 10.18653/v1/2022.acl-long.230. URL https://aclanthology.org/2022.acl-long.230/.

Shaik, H. and Doboli, A. Using a symbolic knowledge graph to address llm limitations in analog circuit topology generation. In *2025 IEEE 15th Annual Computing and Communication Workshop and Conference (CCWC)*, pp. 00528–00533. IEEE, 2025.

Sullivan, R. and Elsayed, N. Can large language models act as symbolic reasoners? *arXiv preprint arXiv:2410.21490*, 2024.

Sweller, J. Chapter two - cognitive load theory. volume 55 of *Psychology of Learning and Motivation*, pp. 37–76. Academic Press, 2011. doi: https://doi.org/10.1016/B978-0-12-387691-1.00002-8. URL https://www.sciencedirect.com/science/article/pii/B9780123876911000028.

Team, G., Anil, R., Borgeaud, S., Alayrac, J.-B., Yu, J., Soricut, R., Schalkwyk, J., Dai, A. M., Hauth, A., Millican, K., et al. Gemini: a family of highly capable multimodal models. *arXiv preprint arXiv:2312.11805*, 2023.

Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., Kaiser, Ł., and Polosukhin, I. Attention is all you need. *Advances in neural information processing systems*, 30, 2017.

Wang, X., Wei, J., Schuurmans, D., Le, Q., Chi, E., Narang, S., Chowdhery, A., and Zhou, D. Self-consistency improves chain of thought reasoning in language models. *arXiv preprint arXiv:2203.11171*, 2022.

Wei, J., Tay, Y., Bommasani, R., Raffel, C., Zoph, B., Borgeaud, S., Yogatama, D., Bosma, M., Zhou, D., Metzler, D., et al. Emergent abilities of large language models. *arXiv preprint arXiv:2206.07682*, 2022.

Xiong, W., Zhang, H., Jiang, N., and Zhang, T. An implementation of generative prm. https://github.com/RLHFlow/RLHF-Reward-Modeling, 2024.

Xu, J., Fei, H., Pan, L., Liu, Q., Lee, M.-L., and Hsu, W. Faithful logical reasoning via symbolic chain-of-thought, 2024. URL https://arxiv.org/abs/2405.18357.

Yang, A., Yang, B., Zhang, B., Hui, B., Zheng, B., Yu, B., Li, C., Liu, D., Huang, F., Wei, H., et al. Qwen2. 5 technical report. *arXiv preprint arXiv:2412.15115*, 2024.

Zeng, Z., Wang, Y., Hajishirzi, H., and Koh, P. W. Evaltree: Profiling language model weaknesses via hierarchical

capability trees, 2025. URL https://arxiv.org/abs/2503.08893.

Zhang, Y., Feng, S., and Tan, C. Active example selection for in-context learning, 2022. URL https://arxiv.org/abs/2211.04486.

Zhu, X., Panigrahi, A., and Arora, S. On the power of context-enhanced learning in llms. *arXiv preprint arXiv:2503.01821*, 2025.

A. Related works

In-context learning example selection. As a key feature of language models, the in-context learning ability (Brown et al., 2020) enables models to improve performance without undergoing gradient-based training. This ability can be maximally activated with carefully chosen in-context demonstrations. Prior works have extensively studied the dynamics of in-context learning (Chen et al., 2024) and effective techniques of in-context example selection (Zhang et al., 2022; Cheng et al., 2023; An et al., 2023; Didolkar et al., 2024a; Liu et al., 2024a) for larger models (>13B). These heuristics often simply rely on the semantic relation between the question and examples, and they typically require training a dedicated example selection model. Meanwhile, the in-context learning dynamics of small language models are understudied.

Classifying model failures. Identifying and understanding language model failures helps us adaptively improve model performance, e.g., via targeted training data selection (Zeng et al., 2025). Prior works have utilized models' test-time failure patterns to build adaptive datasets with *difficult* questions (Dinan et al., 2019; Nie et al., 2020; Ribeiro & Lundberg, 2022; Gao et al., 2023; Li et al., 2025). However, these failure identification and classification approaches have rarely been applied to inform in-context example selection.

Symbolic and Skill-based Reasoning. Performing symbolic reasoning can largely enhance language models' math reasoning ability (Sullivan & Elsayed, 2024; Alotaibi et al., 2024; Xu et al., 2024; Shaik & Doboli, 2025). As SLMs generally possess weaker capabilities to understand complex in-context information, symbolic knowledge aids SLM reasoning by providing structured, less-noisy contextual information (Liao et al., 2024). Notably, the concept of "skill" was proven effective as a useful criterion for clustering symbolic knowledge (Didolkar et al., 2024a), guiding contextual example selection (Didolkar et al., 2024a; An et al., 2023) and mixture-of-experts routing (Chen et al., 2025).

B. Details on separating easy and difficult questions with a process reward model scoring

Because we primarily focus on math datasets, we assume that the model's response is composed of k steps for a question q and contains answer in its final step. We will use the reward model to output reward scores for each step. For simplicity, we will refer to the scores of the reward model as $\{r_{q,1}, \cdots, r_{q,k}\}$. Then, we use thresholds τ_1, τ_2 to classify whether a question q is *easy* or *difficult* for the SLM. We will refer to the thresholding function as $R : \mathcal{Q} \rightarrow \{0, 1\}$.

$$R(q) = \begin{cases} 0, & \text{(if) } r_{q,k} \leq \tau_1 \\ & \text{(or) } \frac{1}{k}\sum_{i=1}^{k} r_{q,i} \leq \tau_1 \\ & \text{(or) } \exists i < k \text{ s.t. } r_{q,i} \leq \tau_2 \\ 1, & \text{otherwise.} \end{cases} \tag{1}$$

The reward model need not be a perfect reward model, we give more details in Appendix F.2.

***Difficult* vs. *easy* questions.** We define $\mathcal{Q}_{\text{difficult}}$ as the set of questions with low-reward model responses R. Accordingly, $\mathcal{Q}_{\text{easy}}$ denotes all remaining questions.

$$\begin{aligned} \mathcal{Q}_{\text{difficult}} &= \{q \mid R(q) = 0\} \\ \mathcal{Q}_{\text{easy}} &= \{q \mid R(q) = 1\} \end{aligned} \tag{2}$$

C. Experimental Details

C.1. Experimental Settings

Datasets. We evaluate on the MATH (7.5k training samples and 5k test samples) (Hendrycks et al., 2021) and GSM8K (7.4k training samples and 1.3k test samples) (Cobbe et al., 2021) datasets. We follow (Didolkar et al., 2024a) to label skills on both the training and test sets using GPT-4o-mini (OpenAI, 2024), and run inference experiments on the whole test set. Appendix C.2 shows the prompt and examples of our skill annotation pipeline. We sample in-context examples from the training set. These two datasets are not overly challenging for SLMs, which ensures relatively interpretable model outputs for stable failure detection. Meanwhile, they are sufficiently representative to offer meaningful insights into our method's efficacy.

Model settings. We tested our methods on five instruction-tuned small language models: Qwen2.5-1.5B-Instruct, Qwen2.5-3B-Instruct, Qwen2.5-7B-Instruct, Llama-3.2-1B-Instruct, and Llama-3.2-3B-Instruct (Yang et al., 2024; Meta AI, 2024). We evaluate the models on 5-shot ICL performance. We use generation temperature at 0.0 for all experiments. We also compare against consistency@5 voting (Wang et al., 2022) with 5-shot fixed examples, where we use 5 generations at temperature 1.0 and evaluate the consistent response. For classifying *easy* and *difficult* questions in the first stage, we use RLHFlow/Llama3.1-8B-PRM-Mistral-Data (Xiong et al., 2024), an 8B process reward model fine-tuned from Llama-3.1-8B, with filtering thresholds $\tau_1 = 0.85, \tau_2 = 0.7$. We use GPT-4o-mini for skill annotation as well as labeling missing skills in AdaptMI+.

Baselines. We compare our method to non-adaptive in-context example selection methods, respectively feeding in fixed examples, random examples, and skill-based examples (Didolkar et al., 2024a) for all queries.

C.2. Skill Annotation on MATH and GSM8K

To construct the skill bank, we follow (Didolkar et al., 2024a) to label skills on both the training and test sets of MATH and GSM8K using GPT-4o-mini (OpenAI, 2024). We enlist all skills that we used to annotate the questions in MATH and GSM8K dataset in Tables 4 and 5 and Appendix C.2, which have been taken from (Didolkar et al., 2024a). We ask the LLM to read the question and provide up to five skills required to solve this question, from the given existing skill list. We show an example prompt for annotating MATH Number Theory questions as follows.

Example skill annotation prompt for MATH Number Theory questions

[TASK]
You'll be given a math question. Your task is to output:
(1) < skill> list here up to five skill(s) that are required to solve this problem, seperated by commas </skill>.
(2) <reason> reason here why these skills are needed </reason>.

[SKILL LIST]
You should only choose the skills from this list:
[
"arithmetic_sequences",
"base_conversion",
"basic_arithmetic",
"division_and_remainders",
"exponentiation",
"factorization",
"greatest_common_divisor_calculations",
"modular_arithmetic",
"number_manipulation",
"number_theory",
"polynomial_operations",
"prime_number_theory",
"sequence_analysis",
"solving_equations",
"understanding_of_fractions"
]

[QUESTION]
{question}

[REASON AND SKILL(S)]

Table 3 shows some example MATH questions and their corresponding annotated skills. From the skill annotation, we construct a Skill Bank (see Figure 1 and Section 2.1) that stores the required skills for each question.

Question	Annotated skills
What is the units digit of $3^1 + 3^3 + 3^5 + 3^7 + \ldots + 3^{2009}$?	exponentiation, modular arithmetic, sequence analysis
In the addition problem each letter represents a distinct digit. What is the numerical value of E? [Figure]	basic arithmetic, number manipulation, solving equations
In triangle ABC, $\tan(\angle CAB) = \frac{22}{7}$, and the altitude from A divides $\overline{BC}$ into segments of length 3 and 17. What is the area of triangle ABC?	geometry and space calculation, trigonometric calculations, arithmetic operations

Table 3. Example MATH questions, and the annotated skills generated by GPT-4o-mini.

Subject	**List of Skills**
	Per subject split in MATH
Algebra	`algebraic_expression_skills`, `algebraic_manipulation_skills`, `arithmetic_skills`, `calculation_and_conversion_skills`, `combinatorial_operations_and_basic_arithmetic`, `complex_number_skills`, `distance_and_midpoint_skills`, `exponent_and_root_skills`, `factoring_skills`, `function_composition_skills`, `function_skills`, `geometric_sequence_skills`, `graph_and_geometry_skills`, `inequality_skills`, `logarithmic_and_exponential_skills`, `number_theory_skills`, `polynomial_skills`, `quadratic_equation_skills`, `ratio_and_proportion_skills`, `sequence_and_series_skills`, `solving_equations`
Counting and Probability	`calculating_and_understanding_combinations`, `combinatorial_mathematics`, `combinatorics_knowledge`, `counting_principals`, `factorials_and_prime_factorization`, `number_theory_and_arithmetic_operations`, `permutation_and_combinations`, `probability_calculation_with_replacement`, `probability_concepts_and_calculations`, `probability_theory_and_distribution`, `understanding_and_applying_combinatorics_concepts`
Geometry	`3d_geometry_and_volume_calculation_skills`, `algebraic_skills`, `area_calculation_skills`, `circle_geometry_skills`, `combinatorics_and_probability_skills`, `coordinate_geometry_and_transformation_skills`, `other_geometric_skills`, `pythagorean_skills`, `quadrilateral_and_polygon_skills`, `ratio_and_proportion_skills`, `triangle_geometry_skills`, `trigonometry_skills`, `understanding_circle_properties_and_algebraic_manipulation`

Table 4. List of skills used for annotating questions in each subject in MATH dataset

Subject	List of Skills
Per subject split in MATH	
Intermediate Algebra	absolute_value_skills, algebraic_manipulation_and_equations, calculus_optimization_skills, complex_number_manipulation_and_operations, function_composition_and_transformation, graph_understanding_and_interpretation, inequality_solving_and_understanding, polynomial_skills, properties_and_application_of_exponents, quadratic_equations_and_solutions, recursive_functions_and_sequences, sequence_and_series_analysis_skills, simplification_and_basic_operations, solving_inequalities, solving_system_of_equations, summation_and_analysis_of_series, understanding_and_application_of_functions, understanding_and_applying_floor_and_ceiling_functions, understanding_and_manipulation_of_rational_functions, understanding_and_utilizing_infininte_series, understanding_ellipse_properties, understanding_logarithmic_properties_and_solving_equations
Number Theory	arithmetic_sequences, base_conversion, basic_arithmetic, division_and_remainders, exponentiation, factorization, greatest_common_divisor_calculations, modular_arithmetic, number_manipulation, number_theory, polynomial_operations, prime_number_theory, sequence_analysis, solving_equations, understanding_of_fractions
Pre-algebra	average_calculations, basic_arithmetic_operations, circles, counting_and_number_theory, exponentiation_rules, fractions_and_decimals, geometry, multiples_and_zero_properties, multiplication_and_division, perimeter_and_area, prime_number_theory, probability_and_combinatorics, ratio_and_proportion, solving_linear_equation
Pre-calculus	algebra_and_equations, basic_trigonometry, calculus, complex_number_operations, complex_numbers, coordinate_systems, determinant_calculation, geometric_relations, geometry_and_space_calculation, geometry_triangle_properties, matrix_operations, parametric_equations, sequences_series_and_summation, three_dimensional_geometry, trigonometric_calculations, vector_operations

Table 5. List of skills used for annotating questions in each subject of MATH dataset (continued from Table 4)

List of skills used for annotating questions in GSM8K

`advanced_arithmetics`, `algebra`, `basic_arithmetics`, `conversion_unit_operations`, `division`, `equations`, `fractions`, `interest_calculations`, `multiplication`, `percentage`, `problem_solving`, `profit_loss`, `ratio_proportion`, `time_speed_distance`

C.3. Missing skill Identification from Model Responses

In Stage 2 of AdaptMI+, we use GPT-4o-mini to label the skills that are missing from a model response. We ask the LLM to read the question along with the SLM response and provide the skills that the model fails to leverage in the response, from the given existing skill list. Below we show an example prompt for labeling missing skills for MATH Number Theory questions, as well as an example LLM output.

Example missing skill identification prompt for a MATH Number Theory question

[TASK]
You'll be given a math question and a step-by-step solution written by a Small Language Model. Your task is to output:
(1) <judge > judge here whether the solution is correct or incorrect </judge >
(2) <reason > if it's incorrect, reason here why the solution is incorrect </reason >,
(3) <skill > list here what skill(s) should the SLM enhance in order to answer correctly, seperated by commas </skill >.

[SKILL LIST]
You should only choose the skills from this list:
["arithmetic_sequences",
"base_conversion",
"basic_arithmetic",
"division_and_remainders",
"exponentiation",
"factorization",
"greatest_common_divisor_calculations",
"modular_arithmetic",
"number_manipulation",
"number_theory",
"polynomial_operations",
"prime_number_theory",
"sequence_analysis",
"solving_equations",
"understanding_of_fractions"]

[QUESTION]
Given that $p \geq 7$ is a prime number, evaluate

$$1^{-1} \cdot 2^{-1} + 2^{-1} \cdot 3^{-1} + 3^{-1} \cdot 4^{-1} + \cdots + (p-2)^{-1} \cdot (p-1)^{-1} \pmod{p}.$$

[MODEL_SOLUTION]
To evaluate the sum where p is a prime number greater than or equal to 7, we start by noting that for any integer a not divisible by p, the modular inverse a^{-1} is the integer b such that $ab \equiv 1 \pmod{p}$. This means that $a \cdot a^{-1} \equiv 1 \pmod{p}$.
....
Thus, the value of the sum is $\boxed{-1}$.

[REASON AND SKILL(S)]

Example GPT-4o-mini output for missing skill identification

(1) <judge> incorrect </judge>
(2) <reason> The solution incorrectly simplifies the expression and misapplies properties of modular arithmetic. The solution claims that the sum of the inverses from 1 to (p-1) is (-1 pmodp), which is incorrect. The correct approach involves recognizing that each term $(a^{-1} \cdot (a+1)^{-1})$ should be evaluated individually, and the sum should be calculated directly without assuming incorrect simplifications. The solution also fails to correctly apply the properties of modular inverses and does not properly evaluate the sum. </reason>
(3) <skill> modular_arithmetic, number_theory, understanding_of_fractions </skill>

C.4. Skill-based Example Retrieval

We outline our algorithm for retrieving in-context examples tailored to a specific set of skills. Leveraging the Skill-Map definition in Section 2.1, which annotates each question with its associated skills, we construct an inverse mapping called Example-Bank : Skill-Bank$(\mathcal{Q}) \rightarrow \mathcal{P}$. This map associates each skill s with the subset of in-context examples in the pool $\mathcal{P}$ that are linked to s according to Skill-Map. Given a question q and a target skill set K, we retrieve in-context examples by randomly selecting one example from Example-Bank(s) for each skill s in K. The algorithm is given in Algorithm 1.

Algorithm 1 Skill-based example retrieval

Input: List of skills $K = [k_1, ..., k_n]$ $(n \leq 5)$
Output: Selected 5-shot examples $E = [e_1, ..., e_5]$

```
E ← []
if K is not empty then
    {We allow an additional repeated in-context example for the first 5 − n skills}
    for i = 1 to 5 − n do
        E' ← Example-Bank(k_1)
        if E' is not empty then
            e ← random_choice(E')
            E ← E + [e]
        end if
    end for

    for each k in K do
        E' ← Example-Bank(k)
        if E' is not empty then
            e ← random_choice(E')
            E ← E + [e]
        end if
    end for
end if


E ← Set(E) {Remove repeated instances}
if len(E) < 5 then
    Append examples from fixed in-context examples to fill remaining shots
    {This happens in the rarest of cases when we don't have enough examples for a skill!}
end if
return E =0
```

Methods	MATH								GSM8K
	Geometry	Precalculus	Algebra	Prealgebra	Number Theory	Intermediate Algebra	Counting & Probability	**Avg.**	**Avg.**
# Qwen2.5-1.5B-Instruct									
Fixed Examples	39.7	38.3	72.2	67.3	45.2	36.5	47.3	52.8	71.5
Random Examples	42.8	41.0	73.1	68.1	43.7	35.1	47.3	53.3	70.9
Skill-based Examples	43.2	39.6	72.0	67.7	45.4	35.8	44.7	53.0	66.1
Consistency@5	44.5	**43.5**	77.6	70.8	**50.0**	**39.8**	47.8	56.9	75.6
AdaptMI	**44.7**	42.1	76.8	72.0	49.8	36.9	50.0	56.4	72.9
AdaptMI+	44.5	42.1	**78.2**	**72.8**	49.1	38.4	**51.5**	**57.2**	**75.8**
# Qwen2.5-7B-Instruct									
Fixed Examples	61.2	61.5	91.2	87.1	74.8	57.3	72.6	74.7	91.7
Random Examples	60.1	62.1	91.4	86.6	74.4	55.7	73.4	74.4	91.1
Skill-based Examples	61.2	64.3	90.6	87.7	73.0	55.9	71.1	74.4	91.7
Consistency@5	62.4	57.7	92.3	87.0	**79.1**	57.5	71.7	75.1	**93.3**
AdaptMI	62.2	**64.7**	91.5	87.6	73.5	57.6	71.5	75.9	92.3
AdaptMI+	**64.9**	63.4	**92.8**	**88.8**	77.4	**58.8**	**74.9**	**76.7**	92.4
# Llama-3.2-1B-Instruct									
Fixed Examples	8.0	11.1	19.6	21.3	10.3	7.8	11.5	13.8	26.8
Random Examples	10.2	6.5	24.0	20.9	7.3	7.9	6.9	13.7	19.3
Skill-based Examples	14.8	6.8	16.7	22.6	11.2	7.3	10.4	13.4	13.4
Consistency@5	13.6	**13.3**	28.8	28.2	**21.4**	6.7	**14.3**	19.4	**29.9**
AdaptMI	13.6	10.3	20.8	29.3	12.1	7.8	12.5	16.2	23.2
AdaptMI+	**17.1**	11.1	**29.6**	**35.4**	10.3	**8.9**	13.5	**19.8**	26.0

Table 6. Additional results to Table 1. AdaptMI and AdaptMI+ demonstrate a consistent accuracy gain by up to 3.6% and 6% respectively, compared with baseline methods. We present all results as Pass@1 accuracy unless otherwise indicated.

D. Additional Results

D.1. AdaptMI and AdaptMI+ performances

Table 6 shows AdaptMI and AdaptMI+ performances on Qwen2.5-1.5B-Instruct, Qwen2.5-3B-Instruct, and Llama-3.2-1B-Instruct. These results align with each other—AdaptMI and AdaptMI+ yield substantial improvement compared with all Pass@1 baseline, while being on par with the Consistency@5 results. Models with smaller size generally exhibit larger performance gains. Notably, AdaptMI increased performance by 3.6% on Qwen2.5-1.5B-Instruct, and AdaptMI+ boosted performance by 6% on Llama-3.2-1B-Instruct.

D.2. Iterative AdaptMI+

Our method can be extended to an iterative loop of adaptive example selection. Each iteration begins with model inference, followed by detecting *difficult* questions and using GPT-4o-mini to select skill-based examples. The selected examples are then fed in with *difficult* questions for model inference in the next iteration. This iterative AdaptMI+ is essentially pushing the SLM to tackle a gradually refined set of *difficult* questions by adaptive teaching. We compare iterative AdaptMI+ with a baseline of iterative random retrieval, where the loop involves inference, random example resampling, and re-inference.

Figure 2 shows that iterative AdaptMI+ consistently improves the reasoning performance on MATH for all three Qwen small language models, while the baseline method struggles to keep pushing the accuracy boundary after the first few iterations. For 1.5B and 3B models, the performance grows rapidly in the first four iterations, and improves more gradually thereafter. The 7B model performance, while starting to degrade by the 10th loop, still increases substantially compared to baseline. Through iterative re-selection of targeted in-context examples, iterative AdaptMI+ demonstrates the potential of progressively guiding small language models to tackle unsolved problems.

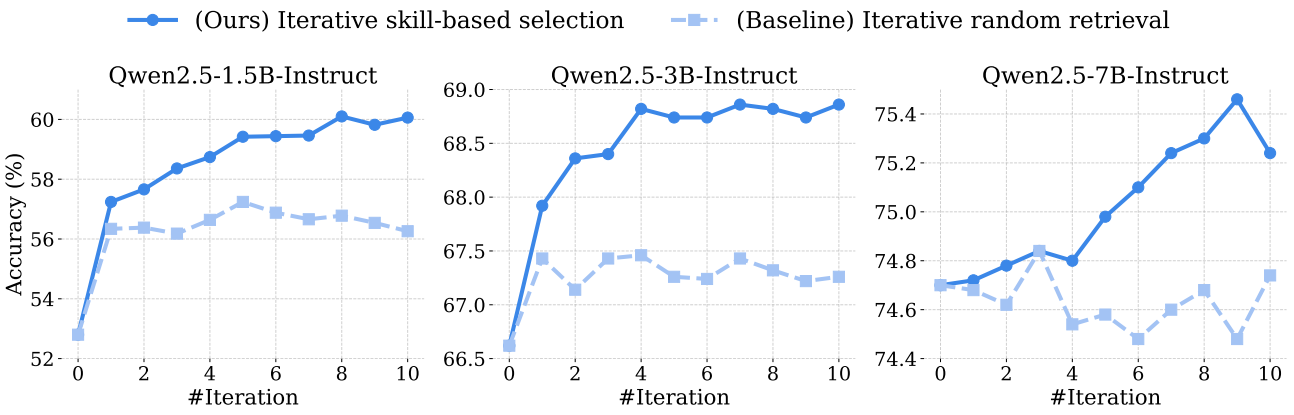

Figure 2. SLM performances under iterative skill-based example selection (AdaptMI+) vs. iterative random example retrieval. Each iteration involves model inference, *difficult* question detection, and random/skill-based example re-selection with GPT-4o-mini. Iterative AdaptMI+ yields a continuous accuracy gain by up to 7.2%, while the baseline leads to fluctuated performances.

D.3. Effect of skill-based examples on difficult and easy questions

In Section 3.3, we introduce our observation that skill-based examples only boost SLM performances on difficult questions but harm performance on easier ones. We present the additional results on Qwen2.5-3B-Instruct and Qwen2.5-7B-Instruct in Table 7. Similar to Table 2, there is a clear performance drop on easy questions with skill-based examples, although the drop for Qwen2.5-3B-Instruct and Qwen2.5-7B-Instruct is less significant than Qwen2.5-1.5B-Instruct.

E. Fine-grained Analysis: Effect of skill-based examples across five difficulty levels

In Section 3.3, we observe that skill-based examples degraded SLM performances on easy questions, which motivates a more fine-grained analysis. We partition our evaluation set into five levels of difficulty, based on the probability of success under Best-of-n sampling (Gui et al., 2024), verified using ground-truth labels. Formally, a question belongs to Difficulty Level ℓ ($1 \leq \ell \leq 4$) if it can be solved with Best-of-$2^{\ell-1}$ sampling, but not with any lower n. Questions that belong to Level 5 can't be solved with Best-of-8 sampling. We provide no in-context examples when measuring the success of Best-of-n sampling and use temperature 1.0. Intuitively, questions in Level 2 are those where the model is more susceptible to minor issues like formatting, where fixed in-context examples could help. For questions in higher levels, on the other hand, the model might benefit more from guidance with carefully selected in-context examples.

After splitting the questions into 5 levels, we compare the effect of skill-based in-context examples with fixed in-context examples on the model's responses to questions in each difficulty level. Figure 3 reports the results on a Qwen-3B model and MATH dataset.

Primary observations: We clearly observe that skill-based in-context examples can perform worse than fixed in-context examples in levels 1 and 2. On the other hand, skill-based in-context examples can substantially help the model on questions in levels 3–5. Furthermore, we observe that responses of the model are substantially longer with skill-based in-context examples, when compared with model responses with fixed in-context examples.

This further shows that with skill-based examples, the model is more likely to "over-think" and make mistakes on easier questions, when simple strategies like Best-of-2 sampling or prompting with fixed in-context examples would have sufficed. This aligns with existing works on the issues of longer chain-of-thought reasoning in language models and how it relates to "problems of over-thinking" in humans (Liu et al., 2024b; Diaconis & Mazur, 2003). [1]

[1] We also present results using the difficulty split of questions annotated in the original MATH dataset in Appendix E.1. Differences in performance and generation length of model's responses with skill-based and fixed in-context examples are less pronounced across difficulty levels. This is expected, as model's own responses must be a better fine-grained indicator on question difficulty.

Question & Example		MATH								GSM8K
		Geometry	Precalculus	Algebra	Prealgebra	Number Theory	Intermediate Algebra	Counting & Probability	**Avg.**	**Avg.**
# Qwen2.5-3B-Instruct										
Diff-cult	Fixed	36.5	37.9	60.6	48.1	49.5	35.5	40.3	42.9	51.6
	Random	36.8	38.7	62.6	50.5	49.2	36.2	41.0	43.4	56.7
	Skill	34.1	41.8	68.3	54.3	50.8	35.0	42.0	45.2	61.8
		-2.4	**+3.9**	**+7.7**	**+6.2**	**+1.3**	**-0.4**	**+1.69**	**+2.3**	**+10.2**
Easy	Fixed	88.5	90.2	95.9	95.4	86.9	82.6	91.1	92.4	96.7
	Random	83.6	86.5	94.0	94.0	84.1	81.8	90.9	91.8	95.6
	Skill	84.7	88.3	93.8	93.8	85.7	79.8	90.5	90.4	93.9
		-3.8	**-1.8**	**-2.2**	**-1.6**	**-1.3**	**-2.8**	**-0.6**	**-2.0**	**-2.8**
# Qwen2.5-7B-Instruct										
Diff-cult	Fixed	50.0	51.3	80.1	71.6	66.8	50.0	61.5	60.7	74.1
	Random	48.3	52.5	81.3	71.3	67.3	49.1	62.7	60.8	76.7
	Skill	52.0	57.4	81.5	74.7	66.9	51.2	61.9	62.7	77.0
		+2	**+6.1**	**+1.4**	**+3.1**	**+0.1**	**+1.2**	**+0.4**	**+2**	**+2.9**
Easy	Fixed	90.8	93.9	98.7	97.5	93.3	89.7	96.1	96.2	97.3
	Random	92.6	93.4	99.2	97.7	91.9	86.5	97.1	95.3	96.4
	Skill	89.8	91.4	96.0	94.7	91.5	86.1	94.7	94.1	95.5
		-1.0	**-2.5**	**-2.7**	**-2.8**	**-1.8**	**-3.6**	**-1.4**	**-2.1**	**-1.8**

Table 7. Accuracy of Qwen2.5-1.5B-Instruct, Qwen2.5-3B-Instruct, and Qwen2.5-7B-Instruct on *difficult* and *easy* questions, respectively under fixed, random, and skill-based examples (additional results for Table 2). Skill-based examples boost performance on *difficult* questions across all categories, while significantly underperforming on *easy* questions. The gap between easy and difficult questions is more pronounced for smaller models.

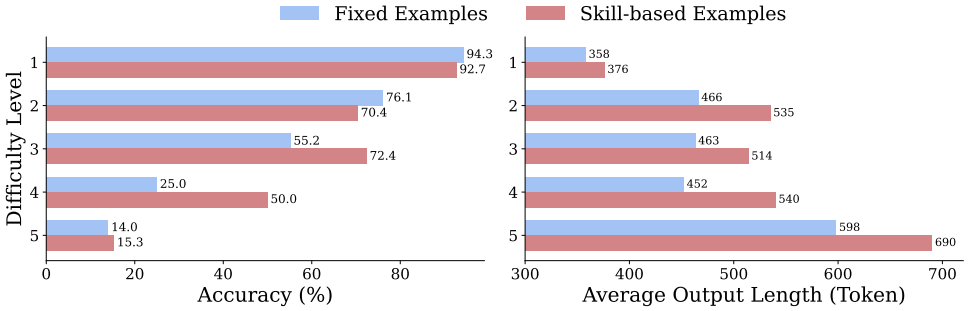

Figure 3. Accuracy and average output length of Qwen2.5-3B-Instruct on questions of Difficulty Level 1–5, designed using its Best-of-n performance, with fixed and skill-based examples. Skill-based examples hinder performance on Levels 1 and 2, while helping on Levels 3–5. On all difficulty levels, skill-based examples result in noticeably longer outputs.

E.1. Fine-grained analysis on original manual split of MATH dataset

We repeat our experiment with a different difficulty splitting strategy. Instead of using Best-of-n sampling to split the evaluation set into 5 levels, we use the manual split of questions given in the original MATH dataset. We report comparisons between skill-based and fixed in-context example selection strategies in Figure 4.

Interestingly, the differences between the ICL performance and generation length with skill-based and fixed in-context examples for the SLM are less pronounced across the 5 difficulty levels, compared to the results in Figure 3. This suggests that the manual difficulty split in the MATH dataset may not align well with the model's own perception of question difficulty. To capture more fine-grained distinctions between the two strategies, using the model's own responses through Best-of-n sampling serves as a more reliable indicator of question difficulty.

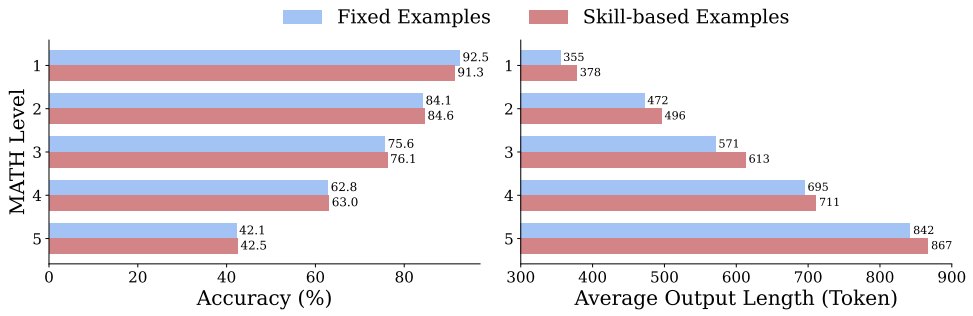

Figure 4. Accuracy and average output length of Qwen2.5-3B-Instruct on questions of Level 1–5 defined in the MATH dataset. Compared to Figure 3, the performance gap between fixed and skill-based examples is unnoticeable across all levels.

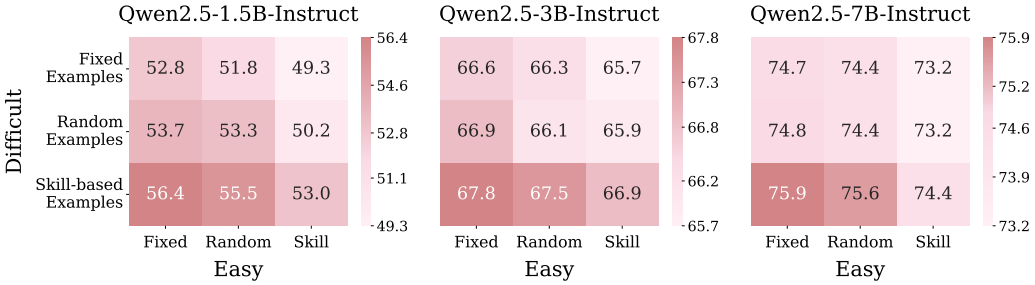

Figure 5. ICL performance, measured in terms of accuracy, across different combinations of in-context examples for *easy* and *difficult* questions on the MATH dataset. Across all models, we observe that skill-based in-context examples for *difficult* questions and fixed in-context examples for the *easy* questions work the best.

F. Ablation Studies

F.1. Comparing different combinations of in-context example types in Stage 2.

Our main method combines *difficult* questions with skill-based examples and *easy* ones with fixed examples, based on the observation that models only need targeted instructions on more challenging cases. To better understand its effectiveness, we conduct an ablation study exploring alternative combinations of in-context examples. Our primary observations are

- As shown in Figure 5, our combination of "difficult + skill-based; easy + fixed" consistently outperforms all other configurations. Notably, the accuracy gap between the best and worst-performing combination can reach 7.1%, which stresses the importance of carefully choosing in-context examples for SLMs.
- The sensitivity to in-context example selection varies across model sizes, with the 1.5B model being the most sensitive and the 7B model being the most stable.

F.2. Ablations on reward model configurations

Effect of threshold values on the reward model scores. We now investigate the effect of τ_1 and τ_2 (defined in Section 2.2) on correct classification of *easy* or *difficult* questions. That is, we measure whether our classification of questions as *easy* or *difficult* also corresponds to correct or incorrect validation of responses using ground-truth labels. Since our hope with the reward model is to successfully detect model failures, without explicitly accessing the ground truth labels, we measure the prediction accuracy of the reward model in terms of recall and F1 scores. In Table 8, we observe that our choice of the threshold values gives the best tradeoff between recall and F1 scores.

Process reward vs. Outcome reward. We also compare the prediction accuracy of our process reward model (PRM) with threshold filtering (see Section 2.2) against directly loading the reward model as an outcome reward model (ORM). Our preliminary experiments indicated 0.9 as the optimal threshold for the outcome rewards. With $\tau = 0.9$, the prediction

$\tau_1 \backslash \tau_2$	0.6	0.7	0.8
0.80	0.76 / 0.85 / 0.80	0.72 / 0.90 / 0.80	0.66 / 0.96 / 0.78
0.85	0.72 / 0.90 / 0.80	0.70 / **0.92** / **0.80**	0.65 / 0.96 / 0.78
0.90	0.64 / 0.95 / 0.77	0.64 / 0.96 / 0.77	0.62 / 0.97 / 0.75

Table 8. Reward model prediction metrics (precision / recall / F1) across different thresholds for Qwen2.5-1.5B-Instruct on MATH. Our choice of the threshold values give the best tradeoff with recall and F1 scores.

metrics of the ORM are: Precision = 0.54 / Recall = 0.90 / F1 = 0.68, whereas the prediction metrics of the PRM with optimal thresholds are Precision = 0.70 / Recall = 0.92 / F1 = 0.80. Therefore, our method using PRM with threshold filtering is superior to directly using ORM.

F.3. Comparing few-shot instructions with natural language instructions

Here, we explore an alternative strategy to construct adaptive in-context instruction. We want to test whether additional supervision from the LLM in AdaptMI+ could be provided in terms of *feedback* using natural language instructions.

For *difficult* questions, we modify our adaptive instruction as follows. After getting the predicted missing skills on model's response from an LLM, we prompt the LLM back with the missing skills and the corresponding skill-based in-context examples and ask the model to return a concise *natural language LLM feedback* that contains criticism on the model's response, and hints on how to apply the required skills. See below for an example prompt.

	MATH	**GSM8K**
Fixed Examples	74.7	91.7
LLM Feedback	74.4	91.2
+ Fixed Examples	74.5	91.1
+ Skill Examples	74.5	91.1

Table 9. Qwen2.5-7B-Instruct accuracy under LLM-generated natural language instructions.

We report the behavior of modified AdaptMI+ on Qwen2.5-7B-Instruct. Interestingly, we observe that even 7B models tend to not benefit from the unstructured instructions (see Table 9). Furthermore, even if skill-based in-context examples are utilized along with LLM feedback, the SLM's performance remains nearly unchanged, which suggests the model simply ignores in-context information that contains long, and unstructured natural language feedback.

Example feedback labeling prompt

[TASK]
You'll be given a math question, a step-by-step solution written by a SLM, a step that is likely to be incorrect, the missing skills in the solution that you identified earlier, and the skill-rated in-context examples. Your task is to output:
(1) <comment> comment here about the SLM solution </comment>
(2) <hint> write here compactly the hints we should give the SLM to help it correctly answer this question next time. You should not include the question or answer of this specific question. </hint>

[QUESTION]
Square $ABCD$ has side length 2. A semicircle with diameter $\overline{AB}$ is constructed inside the square, and the tangent to the semicircle from C intersects side $\overline{AD}$ at E. What is the length of $\overline{CE}$? [Figure]

[SOLUTION]
To solve the problem, we start by identifying the coordinates of the points in the square ABCD . Let's place the square in the coordinate plane with A = (0,0) , B = (2,0) , C = (2,2) , and D = (0,2) . The semicircle is constructed with diameter \overline{AB} , so its center is at the midpoint of \overline{AB} , which is (1,0) , and its radius is 1. The equation of the semicircle is (x-1)^2 + y^2 = 1 for y \geq 0 . We need to find the point E where the tangent from C to the semicircle intersects side \overline{AD} . Since E lies on \overline{AD} , its coordinates are (0, y_E) . The slope of the line from C to E is \frac{y_E - 2}{0 - 2} = \frac{2 - y_E}{2} . The equation of the line CE is: \[y - 2 = \frac{2 - y_E}{2}(x - 2). \] Since E = (0, y_E) , substituting x = 0 and y = y_E into the equation, we get: \[y_E - 2 = \frac{2 - y_E}{2}(-2) = y_E - 2. \] This confirms that the point E lies on the line. To find the exact coordinates of E , we use the fact that the tangent to the semicircle at a point (x_1, y_1) has the equation: \[(x_1 - 1)(x - 1) + y_1 y = 1. \] Since the tangent passes through C = (2,2) , substituting x = 2 and y = 2 into the tangent equation, we get: \[(x_1 - 1)(2 - 1) + y_1 \cdot 2 = 1 \implies x_1 - 1 + 2y_1 = 1 \implies x_1 + 2y_1 = 2. \] Also, since (x_1, y_1) lies on the semicircle, we have: \[(x_1 - 1)^2 + y_1^2 = 1. \] We solve these two equations simultaneously. From x_1 + 2y_1 = 2 , we get x_1 = 2 - 2y_1 . Substituting this into the semicircle equation, we get: \[(2 - 2y_1 - 1)^2 + y_1^2 = 1 \implies (1 - 2y_1)^2 + y_1^2 = 1 \implies 1 - 4y_1 + 4y_1^2 + y_1^2 = 1 \implies 5y_1^2 - 4y_1 = 0 \implies y_1(5y_1 - 4) = 0. \] So, y_1 = 0 or y_1 = \frac{4}{5} . Since y_1 = 0 corresponds to the point A , which is not a valid tangent point, we have y_1 = \frac{4}{5} . Then x_1 = 2 - 2 \cdot \frac{4}{5} = \frac{2}{5} . The point of tangency is \left(\frac{2}{5}, \frac{4}{5} \right) . The length of \overline{CE} is the distance from C = (2,2) to E = \left(0, \frac{4}{5}\right) : \[CE = \sqrt{(2-0)^2 + \left(2 - \frac{4}{5}\right)^2} = \sqrt{4 + \left(\frac{10}{5} - \frac{4}{5}\right)^2} = \sqrt{4 + \left(\frac{6}{5}\right)^2} = \sqrt{4 + \frac{36}{25}} = \sqrt{\frac{100}{25} + \frac{36}{25}} = \sqrt{\frac{136}{25}} = \frac{2\sqrt{34}}{5}. \] Thus, the length of \overline{CE} is \boxed{\frac{2\sqrt{34}}{5}} .

[INCORRECT_STEP]
The slope of the line from C\) to E\) is \frac{y_E - 2}{0 - 2} = \frac{2 - y_E}{2}\). The equation of the line CE\) is: \[y - 2 = \frac{2 - y_E}{2}(x - 2). \] Since E = (0, y_E)\), substituting x = 0\) and y = y_E\) into the equation, we get: \[y_E - 2 = \frac{2 - y_E}{2}(-2) = y_E - 2. \] This confirms that the point E\) lies on the line.

[MISSING_SKILLS]
triangle_geometry_skills,circle_geometry_skills,algebraic_skills

[COMMENT_AND_HINT]

Example feedback from GPT-4o-mini

<comment>The SLM's solution is mostly correct but could benefit from a more detailed explanation of the triangle inequality and how it affects the possible values of the third side. Additionally, the SLM might have overlooked the strict inequality in the triangle inequality theorem, leading to the incorrect conclusion that the minimum value of BC is exactly 2x .</comment>

<hint>
- Emphasize the importance of the triangle inequality theorem, specifically a + b > c , where a , b , and c are the sides of a triangle.
- Explain why BC must be strictly greater than 2x to satisfy the inequality BC + x > 3x .
- Clarify that the smallest integer value for BC that satisfies this condition is 2x + 1 , leading to the calculation of the maximum possible height h .
</hint>

G. Case Studies

In this section, we conduct case studies to gain deeper insight into how skill-based in-context examples might harm performance on easy questions, as mentioned in Section 3.3. We present two questions where SLM successfully solves with fixed examples, while failing with skill-based examples.

G.1. Skill-based examples lead the model to overlook key problem constraints

In the example below, the Qwen2.5-7B-Instruct model is given an algebra question that includes multiple geometric constraints. While the question involves both Geometry and Algebra, it is only classified as an Algebra question in MATH, hence being combined with algebraic skill examples. When prompted with fixed examples, the model correctly identifies two possible answers and chooses the correct one according to the given condition "both coordinates are negative." On the other hand, when conditioned by examples that represent algebraic skills, the model overly emphasizes algebraic completeness but overlooks this important problem condition. It finally selects the incorrect answer by a random guess.

Case Study 1: An example skill-based prompt

< |im_start |>system
Please reason step by step, and put your final answer within \boxed{}.
< |im_end |>

< |im_start |>user
[Example 1 with skill "distance_and_midpoint_skills"]
[Example 2 with skill "distance_and_midpoint_skills"]
[Example 3 with skill "distance_and_midpoint_skills"]
[Example 4 with skill "algebraic_expression_skills"]
[Example 5 with skill "arithmetic_skills"]

A point (x, y) on the coordinate plane with both coordinates negative is a distance of 6 units from the x-axis. It is a distance of 15 units from the point $(8, 3)$. It is a distance $\sqrt{n}$ from the origin. What is n?
< |im_end |>

< |im_start |>assistant

Case Study 1: Correct model response under fixed example prompt

To solve the problem, we need to determine the coordinates (x, y) of the point that satisfies the given conditions and then find the distance from the origin.

1. Distance from the x-axis:
The point (x, y) is 6 units away from the x-axis. Since the y-coordinate is negative, we have:
\[|y| = 6 \imples y = -6 \]

2. Distance from the point \((8, 3)\):
The point \((x, y)\) is 15 units away from \((8, 3)\). Using the distance formula:
\[\sqrt{(x - 8)^2 + (y - 3)^2} = 15 \]
....
This gives us two possible values for \(x\):
\[x = 8 + 12 = 20 \quad \text{or} \quad x = 8 - 12 = -4 \]

3. Distance from the origin:
We need to find the distance from the origin \((0, 0)\) to the point \((x, y)\). The distance formula is:
\[\sqrt{x^2 + y^2} \]
We have two possible points: \((20, -6)\) and \((-4, -6)\).

- For \((20, -6)\):
\[\sqrt{20^2 + (-6)^2} = \sqrt{400 + 36} = \sqrt{436} \]
- For \((-4, -6)\):
\[\sqrt{(-4)^2 + (-6)^2} = \sqrt{16 + 36} = \sqrt{52} \]

Since the problem states both coordinates are negative, only \sqrt{52} is valid.

The correct value of \(n\) is: n = 52

Thus, the final answer is: \boxed{52} ✅

Case Study 1: Incorrect model response under skill-based example prompt

To solve the problem, we need to determine the coordinates \((x, y)\) of the point that satisfies the given conditions and then find the distance from the origin.
1. Distance from the \(x\)-axis:

[Same as correct response]

2. Distance from the point \((8, 3)\):

[Long equation-solving process]

3. Distance from the origin:
We need to find the distance from the origin \((0, 0)\) to the point \((x, y)\). We have two possible points: \((20, -6)\) and \((-4, -6)\). - For \((20, -6)\): \[\sqrt{20^2 + (-6)^2} = \sqrt{400 + 36} = \sqrt{436} = 2\sqrt{109} \] Thus, \(n = 436\). - For \((-4, -6)\): \[\sqrt{(-4)^2 + (-6)^2} = \sqrt{16 + 36} = \sqrt{52} = 2\sqrt{13} \] Thus, \(n = 52\). **Since the problem does not specify which point to use**, we consider the first solution where \(n = 436\).Therefore, the value of \(n\) is \(\boxed{436}\). ✕

G.2. Symbol-heavy skill-based examples cause the model to overthink.

The question below requires a plug-in-and-test approach instead of solving an equation. With fixed in-context examples, the model is able to find out the correct answer by directly plugging in and trying out small values. However, the skill-based examples that involve equation solving may have caused the model to overthink. After failing in the first plug-in-and-test, it ended up attempting to solve the equation system and eventually failed.

Case Study 2: An example skill-based prompt

< |im_start |>system
Please reason step by step, and put your final answer within \boxed{}.
< |im_end |>

< |im_start |>user
[Example 1 with skill "sequence_and_series_skills"]
[Example 2 with skill "sequence_and_series_skills"]
[Example 3 with skill "sequence_and_series_skills"]
[Example 4 with skill "function_composition_skills"]
[Example 5 with skill "solving_equations"]

Charlize accidentally omitted two consecutive integers when adding the elements of the arithmetic sequence, $\{1, 2, 3, \ldots, n\}$. If the sum she obtained is 241, what is the smallest possible value of n?
< |im_end |>

< |im_start |>assistant

Case Study 2: Correct model response under fixed example prompt

To solve the problem, we first need to determine the sum of the arithmetic sequence \(\{1, 2, 3, \ldots, n\}\). The sum of the first \(n\) positive integers is given by the formula: \[S = \frac{n(n+1)}{2} \]

Let the two consecutive integers that Charlize omitted be \(k\) and \(k+1\). The sum of these two integers is: \[k + (k+1) = 2k + 1 \] Therefore, the sum of the sequence with these two integers omitted is: \[S - (2k + 1) = \frac{n(n+1)}{2} - (2k + 1) \] We know that this sum is 241, so we can write: \[\frac{n(n+1)}{2} - (2k + 1) = 241 \] Rearranging this equation, we get: \[\frac{n(n+1)}{2} = 242 + 2k \] Multiplying both sides by 2, we obtain: \[n(n+1) = 484 + 4k \] Since \(k\) is an integer, \(484 + 4k\) must be a product of two consecutive integers. We need to find the smallest \(n\) such that \(n(n+1)\) is close to 484 and is of the form \(484 + 4k\).

Let's check the values of \(n\) starting from the smallest possible \(n\) that is greater than \(\sqrt{484} = 22\):

1. If \(n = 22\), then \(n(n+1) = 22 \times 23 = 506\). We need to check if \(506 = 484 + 4k\): \[506 - 484 = 22 \implies 4k = 22 \implies k = \frac{22}{4} = 5.5 \] Since \(k\) must be an integer, \(n = 22\) is not a solution.
2. If \(n = 23\), then \(n(n+1) = 23 \times 24 = 552\). We need to check if \(552 = 484 + 4k\): \[552 - 484 = 68 \implies 4k = 68 \implies k = \frac{68}{4} = 17 \] Since \(k = 17\) is an integer, \(n = 23\) is a solution. The two consecutive integers omitted are 17 and 18. Therefore, the smallest possible value of \(n\) is: \[\boxed{23} \]

Thus, the final answer is: \boxed{23} ✅

Case Study 2: Incorrect model response under skill-based example prompt

To solve the problem, we first need to determine the sum of the arithmetic sequence \(\{1, 2, 3, \ldots, n \} \). The sum of the first \(n \) positive integers is given by the formula: \[S = \frac{n(n+1)}{2} \]

Charlize omitted two consecutive integers, say \(k \) and \(k+1 \). Therefore, the sum she obtained is: \[S - (k + (k+1)) = S - (2k + 1) = \frac{n(n+1)}{2} - (2k + 1) \] We know that this sum is 241, so we have: \[\frac{n(n+1)}{2} - (2k + 1) = 241 \] Rearranging the equation, we get: \[\frac{n(n+1)}{2} = 242 + 2k + 1 = 243 + 2k \] Multiplying both sides by 2, we obtain: \[n(n+1) = 486 + 4k \] Since \(k \) is an integer, \(486 + 4k \) must be a product of two consecutive integers. We need to find the smallest \(n \) such that \(n(n+1) \) is close to 486 and is of the form \(486 + 4k \).

Let's check the values of \(n\) starting from the smallest possible \(n\) that makes \(n(n+1) > 486\): For \(n = 22\): \[n(n+1) = 22 \times 23 = 506 \] We need to check if \(506\) can be written as \(486 + 4k\): \[506 = 486 + 4k \implies 4k = 20 \implies k = 5 \] So, \(n = 22\) works with \(k = 5\). The two consecutive integers omitted are 5 and 6. Let's verify the sum: The sum of the sequence \(\{1, 2, 3, \ldots, 22\}\) is: \[\frac {22 \times 23 } {2 } = 253 \] If we omit 5 and 6, the sum is: \[253 - (5 + 6) = 253 - 11 = 242 \] This is not correct.

Instead, let's try to solve the equation:

[Model attempts the solve the equation]

[Model reaches maximum output length before finishing] ✕