Does Math Reason the Same Across Languages? Prompting Trade–offs in Multilingual Math Support

Louis Colbus
University of Luxembourg
Luxembourg
louis.colbus.001@student.uni.lu

Nina Hosseini-Kivanani*
University of Luxembourg / RTL
Luxembourg
nina.hosseinikivanani@ext.uni.lu

ABSTRACT

Large language models are increasingly used in math learning settings for hint generation, worked–solution support, and automated assessment. In multilingual deployment, however, prompt choice affects not only correctness but also response cost and fairness across learner languages. We study three prompting paradigms, Direct, Chain–of–Thought (CoT), and Program–of–Thought (PoT), across English, German, French, and Luxembourgish on three math benchmarks spanning school to college levels. Using Qwen3–32B, we compare accuracy, token usage, and latency with and without self–consistency. We find that CoT with self–consistency is the strongest default on arithmetic tasks, while PoT with self–consistency becomes more effective on harder problems requiring sustained computation. Across settings, we observe a stable performance hierarchy of EN $>$ DE $\approx$ FR $>$ LB, with LB also incurring the largest token and latency overhead. These results show that prompt configuration is a practical design decision for multilingual math support systems, with direct implications for reliability, efficiency, and cross–lingual fairness.

Keywords

Large language models, Mathematical reasoning, Cross–lingual evaluation, Multilingual math support, Chain–of–Thought, Program–of–Thought

1. INTRODUCTION

Large language models (LLMs) are increasingly integrated into educational mathematics workflows, including hint generation, worked–solution support, answer validation, and teacher–facing verification. In multilingual settings, such systems must provide support that is not only accurate, but also timely, interpretable, and equitable across learner languages. Yet prompt design is often treated as a purely technical implementation choice, even though it directly affects the quality, transparency, cost, and latency of learner–facing support [6, 18].

Prior work shows that prompting strategies such as Chain–of–Thought (CoT), self–consistency, and Program–of–Thought (PoT) can substantially affect mathematical reasoning performance [20, 19, 2, 4]. CoT elicits intermediate reasoning steps before the final answer, which can improve solution quality and make the response more interpretable. Self–consistency improves robustness by aggregating multiple sampled solutions through voting, although it also increases token usage and latency [19]. Program–based approaches such as PoT or PAL delegate numerical computation to executable code, which is often beneficial on harder mathematical problems that require sustained symbolic or numerical manipulation [2, 4]. These strategies therefore differ not only in correctness, but also in cost, response time, and pedagogical transparency.

Educational research has also begun to explore LLMs as components of tutoring, explanation generation, automated feedback, and assessment workflows [21, 1, 24]. In such settings, evaluation must go beyond raw correctness. A system that is accurate but slow may be unsuitable for interactive classroom use, while a system that produces correct but opaque answers may be less useful for learning support than one that provides interpretable intermediate reasoning. In multilingual educational deployment, these trade–offs become even more important because differences in response quality or speed across languages can translate into unequal learner support.

Multilingual mathematical reasoning remains uneven across languages. Benchmarks such as MGSM show that LLMs can solve math problems in multiple languages, but performance often degrades outside English and other high–resource settings [14]. More recent work similarly reports sensitivity to language resource level, translation quality, and prompting setup [12, 15]. Recent work also questions whether single–set math benchmark accuracy is sufficient for evaluating reasoning, showing that LLM performance can vary across controlled re–instantiations of the same GSM8K–style problems and can deteriorate when irrelevant but seemingly relevant clauses are added [10].

These issues are especially relevant for lower–resource languages, where weaker pretraining coverage may affect both answer quality and reasoning stability. However, prior work

*Corresponding author: nina.hosseinikivanani@ext.uni.lu

Louis Alfred Antoine Colbus, Christoph Schommer, and Nina Hosseini-Kivanani. Does Math Reason the Same Across Languages? Prompting Trade-offs in Multilingual Math Support. In Anthony Botelho, Maria Mercedes T. Rodrigo, Adish Singla, Hiroaki Ogata, Hyojeong So, and Young Hoan Cho (eds.) Proceedings of the 19th International Conference on Educational Data Mining, Seoul, Republic of Korea, June, 2026, pp. 647–652. International Educational Data Mining Society (2026).

https://doi.org/10.5281/zenodo.21040085

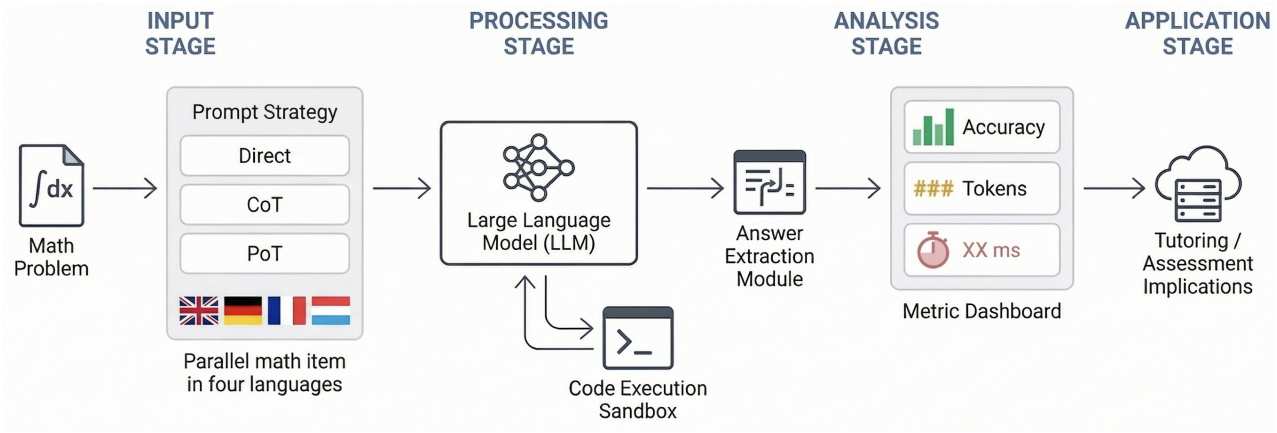

Figure 1: Pipeline of multilingual math support evaluation.

has focused mainly on per–language accuracy and has paid less attention to how prompt choice changes the trade–offs among correctness, token cost, latency, and interpretability in multilingual educational use.

These benchmarks are not only abstract tests of model reasoning. They also approximate common functions in learner–facing systems. Arithmetic problems are closely related to stepwise hinting, answer checking, and formative support in school mathematics, while harder problems reflect advanced feedback, verification, and teacher–facing solution analysis. Under this view, prompt choice is not merely an inference–time optimization detail. It changes the quality, speed, and interpretability of support that multilingual tutoring and assessment systems provide to different learner groups.

We address this gap through a compact empirical study across four languages, English (EN), German (DE), French (FR), and Luxembourgish (LB), using three prompting paradigms: Direct, CoT, and PoT. We evaluate performance on arithmetic and harder mathematics benchmarks and analyze both answer quality and computational cost, with particular attention to LB as a low–resource language relevant to multilingual educational deployment.

These trade–offs matter directly for educational use. In tutoring and hint generation, prompt choice affects whether learners receive interpretable step–by–step guidance or only a final answer. In automated checking, latency and computational cost affect classroom usability and scalability. In multilingual settings, unequal performance across languages raises fairness concerns because students using lower–resource languages may receive slower or less reliable support. Our study therefore treats prompting as a design choice for multilingual tutoring, feedback, and assessment systems, not only as an inference–time optimization problem. Our contributions are:

- We provide a focused evaluation of multilingual math support performance across EN, DE, FR, and LB, including a low–resource language.
- We show that the most suitable prompting strategy depends on task difficulty. CoT with self–consistency is strongest on arithmetic tasks, while PoT with self– consistency is more effective on harder mathematics.
- We show that lower–resource language support can be both less accurate and more computationally expensive, which makes prompt configuration a fairness and deployment issue for tutoring and assessment systems.

2. METHODOLOGY

2.1 Benchmarks and languages

We evaluate on three math benchmarks covering different difficulty levels: GSM8K [3], SVAMP [11], and CollegeMath [16]. GSM8K and SVAMP emphasize arithmetic word problems, while CollegeMath contains advanced textbook–style mathematics. We use EN test sets and translate them into DE, FR, and LB using DeepL, preserving mathematical expressions verbatim and applying light post–editing. Manual spot–checking was used to verify that translated items preserved problem structure, numerical references, and solution equivalence across languages. This step helps reduce the risk that observed cross–lingual differences are caused by obvious translation artifacts rather than by differences in model behavior. Table 1 shows a parallel SVAMP item with the original EN problem and its FR, DE, and LB translations.

These benchmarks also serve as controlled proxies for multilingual educational use cases, including hint generation, worked–solution support, answer validation, and teacher–facing checking or verification. To support reproducibility and facilitate further experimentation, the translated benchmark files as well as the full code used for the evaluation pipeline are publicly available on the accompanying repository[1].

2.2 Model and prompting

We use Qwen3–32B [22], a strong multilingual open–weight model with established mathematical reasoning ability, making it suitable for controlled comparison across prompt styles and languages. It is a 32B dense model trained on approximately 36T tokens across 119 languages and dialects, and

[1]Project repository: multilingual-math-bench.

Table 1: Parallel SVAMP example with the original English item and its French, German, and Luxembourgish translations.

Lang.	Problem text
EN	Debby bought 200 water bottles and 256 soda bottles when they were on sale. If she drank 312 water bottles and 4 soda bottles a day, how many days would the soda bottles last?
FR	Debby a acheté 200 bouteilles d'eau et 256 bouteilles de soda lorsqu'elles étaient en promotion. Si elle buvait 312 bouteilles d'eau et 4 bouteilles de soda par jour, combien de jours les bouteilles de soda dureraient–elles ?
DE	Debby hat 200 Wasserflaschen und 256 Limonadenflaschen gekauft, als diese im Angebot waren. Wenn sie täglich 312 Wasserflaschen und 4 Limonadenflaschen trinkt, wie viele Tage reichen die Limonadenflaschen dann?
LB	Debby huet 200 Waasserfläschen a 256 Soda–Fläschen kaaft, wéi se am Rabatt waren. Wann si all Dag 312 Waasserfläschen a 4 Soda–Fläschen drénkt, wéi vill Deeg géifen d'Soda–Fläschen duergoen?

includes a native thinking mode for internal reasoning [22]. We evaluate it across three benchmarks, four languages, and three prompting strategies in a multilingual math–support setting. Figure 1 summarizes the evaluation pipeline.

For each problem, we compare three prompting modes. Direct prompting requests only the final answer. CoT prompting elicits intermediate reasoning steps before the answer. PoT prompting generates executable Python code whose returned output is treated as the final answer. For each strategy, we evaluate both single–sample decoding and self–consistency with five samples (SC@5). In the SC@5 setting, we sample five independent outputs per item and aggregate their final answers by majority vote. This design allows us to compare not only prompting styles, but also the trade–off between single–pass efficiency and ensemble–style robustness.

2.3 Decoding and evaluation

We use the same prompting setup across all four languages and evaluate both single–sample decoding and self–consistency with five samples (SC@5). We use a fixed temperature of $T = 1.0$ across prompting strategies to encourage output diversity for self–consistency, while leaving other decoding parameters at their default settings.

For each dataset and language, we evaluate the full test split under both single–sample and SC@5 settings. For each item, final answers are extracted with a uniform post–processing pipeline that normalizes formatting differences across languages and prompting styles. Predictions are counted as correct when the extracted answer matches the reference after simple normalization, such as removing whitespace or formatting wrappers. For arithmetic expressions, we additionally allow equivalence under basic symbolic normalization. For PoT, only the printed final output is evaluated, and executions that fail to produce a parsable answer are counted as incorrect. We record problem–level accuracy, average generated tokens, and wall–clock latency. This setup allows us to compare not only answer quality, but also the deployment trade–off between efficiency and robustness in multilingual educational settings.

We report problem–level accuracy, average generated tokens, and wall–clock latency. These metrics are relevant for multilingual educational deployment, where systems must balance answer quality with response cost and speed. Accuracy captures answer quality, generated tokens approximate inference cost, and latency reflects whether feedback can be delivered quickly enough for interactive use.

3. RESULTS AND DISCUSSION

Table 2 summarizes the main accuracy patterns. Across prompting modes and datasets, self–consistency generally improves over single–sample decoding, with larger gains on harder tasks and for some prompting styles. Two findings stand out. First, the best prompting strategy depends on task difficulty: CoT with self–consistency is strongest on arithmetic tasks, while PoT with self–consistency becomes the best–performing option on harder mathematics. Second, performance is consistently uneven across languages: EN performs best, DE and FR are usually close behind, and LB shows the largest degradation.

Table 2: Accuracy (%) for Qwen3–32B across languages. Highlighted rows indicate the best–performing method.

Dataset	Method	EN	DE	FR	LB
GSM8K	Direct	78.1	64.8	66.9	33.5
	CoT SC@5	**87.5**	**71.2**	**69.1**	**51.5**
	PoT SC@5	69.3	70.3	68.5	44.7
SVAMP	Direct	81.6	72.2	74.3	44.2
	CoT SC@5	**87.0**	**76.9**	**74.9**	**68.6**
	PoT SC@5	66.2	73.9	71.9	64.6
CollegeMath	Direct	41.6	41.0	38.4	31.1
	CoT SC@5	38.2	40.0	39.9	35.6
	PoT SC@5	**56.6**	**48.7**	**51.9**	**42.1**

For school–level arithmetic, CoT with self–consistency offers the best balance of performance and interpretability. On GSM8K, for example, CoT SC@5 improves EN accuracy from 78.1% with Direct prompting to 87.5%, and similar gains appear on SVAMP. This pattern is consistent with prior work showing that self–consistent CoT can recover diverse but correct reasoning paths [19]. For learner–facing support, CoT offers an additional advantage: beyond improving answer quality, it produces explicit intermediate reasoning that can be repurposed as hints or worked examples.

PoT behaves differently on easier benchmarks. On GSM8K for Qwen3–32B, PoT can underperform because the model sometimes fails to follow the Python instructions or produces code with syntax errors, which the execution environment counts as incorrect. This suggests that the underlying solution strategy may be reasonable, but brittle at the implementation level, in line with prior observations for PAL and related code–based methods [2, 7].

A qualitative inspection of failed outputs suggests that er-

Table 3: Observed trade–offs of prompting strategies for multilingual math support.

Method	Observed strength	Observed constraint	Deployment relevance
Direct	Computationally efficient and structurally simple.	Does not provide explicit intermediate reasoning.	Appropriate when answer delivery and low overhead are prioritized.
CoT + SC@5	Strongest arithmetic performance, with interpretable reasoning traces.	Increased token usage and response latency.	Appropriate for explanatory support, including hints and worked examples.
PoT + SC@5	Strongest performance on harder tasks requiring sustained computation.	Depends on stable code generation and execution.	Appropriate for advanced verification and more formal solution support.

rors arise from several sources. For Direct and CoT prompting, failures often involve incorrect arithmetic, incomplete final–answer extraction, or reasoning paths that are plausible but numerically inconsistent. For PoT, errors are more implementation–sensitive: some outputs contain syntactically invalid Python code, missing print statements, or code that computes an intermediate quantity rather than the requested final answer. In the translated settings, some failures also appear to be linked to language–specific phrasing or answer–format variation. These observations are not intended as a full causal error taxonomy, but they indicate that multilingual math–support failures involve both reasoning errors and interface–level issues such as parsing, translation, and code execution.

On harder problems, the pattern changes. For CollegeMath, PoT SC@5 outperforms Direct and CoT in EN and remains strongest in the other languages. This suggests that delegating computation to executable code becomes useful when tasks require sustained symbolic or numerical manipulation, in line with prior work on program–aided mathematical reasoning [2, 4]. In educational settings, this makes PoT suited to advanced support scenarios such as automated checking, teacher–facing verification tools, or systems that generate formal worked solutions for difficult exercises.

Table 3 summarizes these observed trade–offs and links each prompting strategy to its most relevant educational deployment scenario.

No single prompting strategy is optimal across all forms of math support. CoT is strongest when interpretable step–by–step guidance is valuable, whereas PoT becomes preferable when computational precision dominates. This matters for educational applications, where the same system is expected to support both hint generation and answer validation.

For educational applications, this distinction is consequential. If a system is primarily used to generate hints or worked examples, CoT may be preferable because it exposes intermediate reasoning that can be shown to learners or adapted by teachers. If the system is used for automated checking or higher–stakes verification, PoT may be preferable on harder tasks because computational reliability becomes more important than explanatory detail. In multilingual settings, these design choices also interact with equity, since learners using lower–resource languages may receive support that differs not only in quality but also in responsiveness.

Across all settings we observe a consistent hierarchy

$$\text{EN} > \text{DE} \approx \text{FR} > \text{LB},$$

where German and French typically retain much of the English baseline, while Luxembourgish shows larger drops, especially on the harder benchmark. This pattern aligns with multilingual math evaluations that report stronger performance for high–resource languages and larger degradation for lower–resource ones [14, 15, 12].

Efficiency also differs by language. Figure 2 shows that LB requires more generated tokens than EN across prompting strategies, with especially large increases under self–consistency. The token gap is also consistent across methods, which suggests that the efficiency disadvantage is not tied to a single prompting style. In our runs, this also translated into higher latency. For example, under single–sample CoT on GSM8K, average latency increased from approximately 5.7 seconds in EN to 7.6 seconds in LB. Thus, lower–resource language support is disadvantaged twice: it is often less accurate and more expensive, which aligns with prior multilingual mathematical reasoning studies reporting stronger performance in high–resource languages and larger drops in lower–resource settings [14, 12, 15]. For multilingual tutoring or classroom–facing systems, this matters because learners using lower–resource languages may receive slower and less reliable support, even when the underlying task is the same.

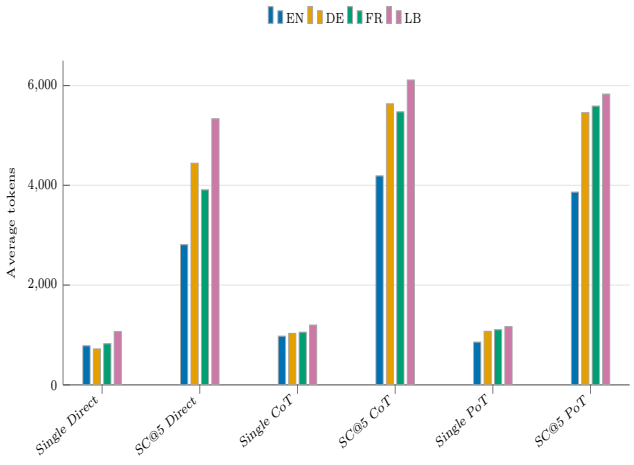

Figure 2: Average token usage across prompting strategies and languages. Self–consistency (SC@5) increases token cost, and LB is consistently among the most expensive settings.

Our findings have three implications for educational data mining and multilingual math support systems: **First**, prompt choice should be treated as a deployment decision rather than a neutral implementation detail. For arithmetic tutoring, CoT with self–consistency provides a strong default because it improves accuracy while preserving interpretable reasoning traces that can support hints, worked examples, or feedback generation [9]. By contrast, Direct prompting may be cheaper, but it offers less transparency and may therefore be less suitable when the instructional goal is to help learners understand the solution process rather than only obtain the correct answer. In education, the value of a response depends not only on correctness but also on whether the reasoning process is understandable and usable. This makes interpretability a functional requirement rather than an optional property when the system is intended to support learning rather than only scoring.

Second, the best strategy changes with task difficulty. For harder mathematics, PoT with self–consistency becomes more reliable than CoT, suggesting that external code execution can stabilize multi–step symbolic or numerical reasoning [23, 8]. This is especially relevant for advanced exercises, automated checking, or teacher–facing systems that must verify mathematically precise solution procedures. In other words, a single prompting strategy is unlikely to be optimal across all educational uses of LLM–based math support.

Third, multilingual support shows clear operational disparities across languages. The consistent hierarchy EN $>$ DE $\approx$ FR $>$ LB indicates that a single prompt configuration may offer uneven quality across learner languages, although this study does not isolate the causal source of the gap. The higher token and latency burden for LB suggests that multilingual systems should use language–specific calibration and fairness auditing rather than assuming uniform performance [5, 17].

This cross–lingual gap has consequences for multilingual educational deployment. A student asking for support in a lower–resource language may receive feedback that is not only less accurate, but also slower and more expensive to generate. This creates an equity issue for tutoring and assessment systems serving linguistically diverse learners. Multilingual math support should therefore be evaluated not only for average correctness, but also for consistency of support quality across learner languages, alongside broader trade–offs between reasoning quality and computational efficiency [13].

4. LIMITATIONS AND CONCLUSION

This study is benchmark–based and does not evaluate learners, teachers, classroom interaction, or learning outcomes directly. The multilingual setup also relies on translated benchmarks, so residual translation or answer–extraction artifacts may affect results, and EN performance may benefit from stronger pretraining coverage or benchmark familiarity. This caution is reinforced by recent work showing that performance on GSM8K–style tasks can vary across controlled re–instantiations and small problem modifications, suggesting that single benchmark scores may not fully capture reasoning robustness [10]. Our scope is also limited to one multilingual model, four languages, and a small set of prompting strategies, so the exact trade–offs may differ for other models or language settings.

Even with these limitations, the main patterns are consistent. CoT with self–consistency is the strongest default for arithmetic support, whereas PoT with self–consistency is more effective for harder problems that benefit from executable reasoning. Across languages, lower–resource settings such as LB show both accuracy degradation and higher efficiency costs. For multilingual math support systems, prompt configuration is therefore a question of reliability, cost, and fairness. More broadly, evaluating multilingual LLMs for mathematics is not only a question of benchmark performance, but also of how reliably and equitably support can be delivered across learner languages.

5. ETHICS STATEMENT

This study does not involve the collection of new human–subject data, learner interaction data, classroom data, or personal information. All experiments are conducted on existing mathematical benchmark datasets and their translated versions. No personally identifiable information is processed, and no participants are recruited. Therefore, the study does not raise human–subject privacy risks. The work is intended as a benchmark–based analysis of multilingual prompting behavior, not as an evaluation of individual learners or teachers. Because no new human–subject data were collected, local ethics–board approval was not required under the scope of this study.

6. ACKNOWLEDGMENTS

This research was conducted in the context of the LuxVoice project, funded by the Luxembourg National Research Fund (FNR) under grant agreement (project reference 19205922). LuxVoice aims to advance Luxembourgish language technologies and support the development of robust multilingual AI resources. The authors gratefully acknowledge the support of RTL Lëtzebuerg and the University of Luxembourg. This work contributes to broader efforts toward inclusive, reliable, and linguistically grounded AI systems for Luxembourgish and other low–resource language settings.

7. REFERENCES

[1] C. Borchers and T. Shou. Can large language models match tutoring system adaptivity? a benchmarking study. In *International Conference on Artificial Intelligence in Education*, pages 407–420. Springer, 2025.

[2] W. Chen, X. Ma, X. Wang, and W. W. Cohen. Program of thoughts prompting: Disentangling computation from reasoning for numerical reasoning tasks. *arXiv preprint arXiv:2211.12588*, 2022.

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

[4] L. Gao, A. Madaan, S. Zhou, U. Alon, P. Liu, Y. Yang, J. Callan, and G. Neubig. Pal: Program-aided language models. In *International Conference on Machine Learning*, pages 10764–10799. PMLR, 2023.

[5] A. Gupta, J. Reddig, T. Calo, D. Weitekamp, and C. J. MacLellan. Beyond final answers: Evaluating large language models for math tutoring. In *International Conference on Artificial Intelligence in Education*, pages 323–337. Springer, 2025.

[6] V. Gupta, S. P. Chowdhury, V. Zouhar, D. Rooein, and M. Sachan. Are large language models for education reliable across languages? In *Proceedings of the 20th Workshop on Innovative Use of NLP for Building Educational Applications (BEA 2025)*, pages 612–631. Association for Computational Linguistics, 2025.

[7] S. Imani, L. Du, and H. Shrivastava. Mathprompter: Mathematical reasoning using large language models. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 5: Industry Track)*, pages 37–42, 2023.

[8] Z. Jie, T. Q. Luong, X. Zhang, X. Jin, and H. Li. Design of chain-of-thought in math problem solving. *arXiv preprint arXiv:2309.11054*, 2023.

[9] Z. Liang, Y. Liu, T. Niu, X. Zhang, Y. Zhou, and S. Yavuz. Improving llm reasoning through scaling inference computation with collaborative verification. *arXiv preprint arXiv:2410.05318*, 2024.

[10] I. Mirzadeh, K. Alizadeh, H. Shahrokhi, O. Tuzel, S. Bengio, and M. Farajtabar. Gsm-symbolic: Understanding the limitations of mathematical reasoning in large language models. *arXiv preprint arXiv:2410.05229*, 2024.

[11] A. Patel, S. Bhattamishra, and N. Goyal. Are nlp models really able to solve simple math word problems? In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 2080–2094, 2021.

[12] B. Paul, J. J. Era, M. R. Zim, T. S. Aothoi, and F. M. Shah. Leveraging large language models for bengali math word problem solving with chain of thought reasoning. *arXiv preprint arXiv:2505.21354*, 2025.

[13] K. Seßler, Y. Rong, E. Gözlüklü, and E. Kasneci. Benchmarking large language models for math reasoning tasks. *arXiv preprint arXiv:2408.10839*, 2024.

[14] F. Shi, M. Suzgun, M. Freitag, X. Wang, S. Srivats, S. Vosoughi, H. W. Chung, Y. Tay, S. Ruder, D. Zhou, et al. Language models are multilingual chain-of-thought reasoners. *arXiv preprint arXiv:2210.03057*, 2022.

[15] M. E. Sobhani, M. F. A. Sayeedi, T. Mohiuddin, M. M. Islam, and S. Shatabda. Mathmist: A parallel multilingual benchmark dataset for mathematical problem solving and reasoning. *arXiv preprint arXiv:2510.14305*, 2025.

[16] Z. Tang, X. Zhang, B. Wang, and F. Wei. Mathscale: Scaling instruction tuning for mathematical reasoning. In *International Conference on Machine Learning*, pages 47885–47900. PMLR, 2024.

[17] J. C. Tonga, K. Srivatsa, K. K. Maurya, F. Koto, and E. Kochmar. Simulating llm-to-llm tutoring for multilingual math feedback. *arXiv preprint arXiv:2506.04920*, 2025.

[18] S. Wang, T. Xu, H. Li, C. Zhang, J. Liang, J. Tang, P. S. Yu, and Q. Wen. Large language models for education: A survey and outlook. *IEEE Signal Processing Magazine*, 42(6):51–63, 2026.

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

[20] J. Wei, X. Wang, D. Schuurmans, M. Bosma, F. Xia, E. Chi, Q. V. Le, D. Zhou, et al. Chain-of-thought prompting elicits reasoning in large language models. *Advances in neural information processing systems*, 35:24824–24837, 2022.

[21] D. Weitekamp, M. N. Siddiqui, and C. J. MacLellan. Tutorgym: A testbed for evaluating ai agents as tutors and students. In *International Conference on Artificial Intelligence in Education*, pages 361–376. Springer, 2025.

[22] A. Yang, A. Li, B. Yang, B. Zhang, B. Hui, B. Zheng, B. Yu, C. Gao, C. Huang, C. Lv, et al. Qwen3 technical report. *arXiv preprint arXiv:2505.09388*, 2025.

[23] H. Ying, S. Zhang, L. Li, Z. Zhou, Y. Shao, Z. Fei, Y. Ma, J. Hong, K. Liu, Z. Wang, et al. Internlm-math: Open math large language models toward verifiable reasoning. *arXiv preprint arXiv:2402.06332*, 2024.

[24] L. Zhang, J. Lin, Z. Kuang, S. Xu, and X. Hu. Spl: a socratic playground for learning powered by large language model. *arXiv preprint arXiv:2406.13919*, 2024.