

Bangla-Math: A New Method for Solving Math Problems in Bangla using LLM

Shreya Nag Riya (1221)

Department of CSE

Jahangirnagar University

Dhaka, Bangladesh

nag.stu2018@juniv.edu

Ajoy Sarker (1118)

Department of CSE

Jahangirnagar University

Dhaka, Bangladesh

ajoy.stu20181@juniv.edu

Md. Abdul Mukit (1126)

Department of CSE

Jahangirnagar University

Dhaka, Bangladesh

mdabdulmukit98@gmail.com

Md. Musfique Anwar

Department of CSE

Jahangirnagar University

Dhaka, Bangladesh

manwar@juniv.edu

Abstract—This research introduces a large language model (LLM) designed to solve a diverse range of mathematical problems in the Bangla language, spanning from basic arithmetic and algebra to more complex topics such as geometry, percentages, and word problems. The Bangla language has its own unique grammatical structure, which makes this task particularly challenging. To address this, we fine-tuned two lightweight and open-source LLMs — LLaMA 3.2 3B and Gemma 2B — using a cleaned and well-organized subset of the Ashrafur Bangla Math dataset, publicly available on Hugging Face. LoRA-based, parameter-efficient fine-tuning was applied to enable training within limited computational resources. We evaluated the models based on mathematical accuracy, solution completeness, language quality, response coherence, and computational efficiency. Gemma 2B outperformed LLaMA 3.2-3B in producing clearer, more natural Bangla solutions, while LLaMA 3.2-3B offered slightly faster inference. Although both models performed well on basic math problems, they struggled with more advanced tasks. Still, this work demonstrates that it is possible to develop useful Bangla-language math solvers using open tools and limited resources. Our system represents a meaningful step toward making AI-powered math education more accessible to Bangla-speaking learners.

Keywords: Bangla language, large language models (LLMs), LoRA fine-tuning, low-resource language

I. INTRODUCTION

The rapid advancement of large language models (LLMs) has revolutionized natural language processing, enabling sophisticated solutions across diverse linguistic and domain-specific challenges. However, the application of LLMs to low-resource languages, such as Bangla, remains underexplored, particularly in specialized domains like mathematics. In Bangladesh, where Bangla is the native language of over 160 million people, students and educators often face difficulties in accessing advanced computational tools tailored to their linguistic context for solving mathematical problems. This gap underscores the need for language-specific AI systems capable of interpreting and resolving math-related queries in Bangla, thereby enhancing educational accessibility and fostering inclusivity.

This research introduces “Bangla-Math,” a novel method that leverages the capabilities of open-source LLMs, such as LLaMA 3.2 or DeepSeek R1, to address this challenge. By fine-tuning these models on specialized datasets of Bangla math questions and answers, we aim to enhance their proficiency in understanding and solving math problems expressed in the language. The significance of this work lies in its potential to bridge the linguistic divide in STEM education, empowering Bangla-speaking learners with an AI-driven tool tailored to their needs. This paper outlines the methodology, evaluates the performance of the fine-tuned model, and discusses its implications for educational technology in low-resource language settings.

II. BACKGROUND STUDY

This section deals with some previous work relevant to this research.

The application of large language models (LLMs) in solving Bangla Math Olympiad problems is explored in [11]. In this study, several LLMs were trained on Bangla math problems, and a technique called retrieved-augmented generation (RAG) was employed to enhance accuracy. Models such as DeepSeekMath-7B-Instruct, NuminaMath-7B-CoT, and Qwen2.5-7B were tested, with Qwen2.5-7B-Instruct performing particularly well. Notably, a larger model, Qwen2.5-32B, solved 77 out of 100 problems correctly without additional training. The research showed strong performance on challenging topics like number theory and geometry, supporting our goal of leveraging LLMs to solve Bangla math questions.

In [8], the authors aimed to improve LLMs’ ability to solve complex math problems such as algebra and calculus. They utilized the PaLM 2-L model and enhanced it through specialized training methods, including multitask learning and step-by-step solution fine-tuning. This approach boosted the model’s accuracy to 58.8%, a significant improvement over previous results. Their findings demonstrate that with

appropriate training, LLMs can substantially improve their mathematical problem-solving skills. This work provides valuable insights for developing math-solving models in other languages, including Bangla.

In [14], a dataset named MetaMathQA was created to strengthen LLMs' math problem-solving capabilities. Models like LLaMA-2 and Mistral-7B were fine-tuned using this dataset. For example, the MetaMath-Mistral-7B model achieved 77.7% accuracy on the GSM8K benchmark and 28.2% on the MATH benchmark, outperforming many larger models. This highlights that well-structured training data can significantly enhance the reasoning skills of even smaller models.

A benchmark called FineMath was introduced in [7] to evaluate LLMs on Chinese math problems. Problems were categorized by topic—such as algebra and geometry—and by difficulty. Models including GPT-4, ChatGLM3-6B, and Baichuan2-13B were evaluated, revealing that performance declined as problem difficulty increased. This setup allowed a clear analysis of each model's strengths and weaknesses. A similar benchmark structure could be beneficial for our Bangla math project to fairly assess model performance across different problem types.

A parameter-efficient fine-tuning (PEFT) method was proposed in [2] to improve LLMs' math-solving abilities. Rather than fine-tuning the entire model, only key components were adjusted, reducing training time and computational resources. This technique was applied to models like LLaMA-2 on English math tasks, achieving strong results with considerably lower cost. This approach is particularly relevant to our Bangla math project, where computational resources may be limited.

Challenges in Bangla NLP and AI development were reviewed in [9]. The researchers highlighted a scarcity of high-quality Bangla datasets, including large text corpora and structured problem sets, which hinder AI training efforts. Moreover, the complexity of Bangla grammar poses additional difficulties. These challenges directly affect AI-based solutions for Bangla, including our "Bangla-Math" project. Their findings underscore the urgent need to develop dedicated datasets, a central focus of our work.

The 2nd MathNLP proceedings [1] present recent advancements in combining natural language processing with mathematical problem-solving. The papers presented used methods like symbolic math reasoning, formula grounding, and co-reference resolution to enhance model accuracy. Various LLMs, including fine-tuned transformer models, were tested. These studies offer useful strategies for combining language and math reasoning—an approach highly relevant to our system.

In [15], the authors studied how well different large language models (LLMs) solve math problems. They tested models such as GPT-4, PaLM, and LLaMA on datasets like GSM8K and MATH, which include both simple and complex math questions. The models were evaluated based on the accuracy of their answers and the clarity of their solution steps. Among them, GPT-4 performed the best, especially

when using methods such as chain-of-thought prompting. Another study, [10], introduced BEnQA—a dataset comprising questions from Bengali and English exams at the middle and high school levels. Models like GPT-4, GPT-3.5, Claude 2.1, LLaMA-2, and Mistral were evaluated using various prompting strategies, including chain-of-thought. Once again, GPT-4 achieved the highest accuracy of 75.1%, demonstrating strong reasoning skills in Bengali. Together, these studies highlight effective models and prompting methods for mathematical reasoning, offering valuable insights for developing a robust Bangla math solution system.

[6] proposed a self-critique pipeline to improve how language models solve math problems. Models from the ChatGLM family were trained to review their own answers, provide feedback, and learn from their critiques. This was followed by specialized fine-tuning to encourage preference for better solutions, resulting in higher accuracy. This approach is valuable for our project, demonstrating how self-feedback can increase model reliability and precision.

In [5], the authors investigated LLMs' ability to solve complex math problems with more than two unknowns. They tested models such as GPT-3.5 and GPT-4 on advanced algebra and reasoning tasks. Results indicated that while models can solve some difficult problems, their performance decreases with increasing complexity. This study helps us understand current model limitations and provides insights for developing more robust math-solving systems for Bangla.

III. DATASET DESCRIPTION

For this research work, we used the "Ashrafur Bangla Math" dataset, which is an open-source dataset available at Hugging Face (https://huggingface.co/datasets/hamim-87/Ashrafur_bangla_math). This dataset includes 850,000 rows of math problems and solutions written in Bangla. The dataset has two columns: a "problem" column with questions and a "solution" column with step-by-step solutions. The problems range from basic arithmetic to algebra and geometry, making it ideal to prepare our large language model (LLM) to solve Bangla math problems. We preprocessed the dataset by correcting errors, adding details where needed, and ensuring every problem had a clear solution. This massive collection of data helps us to overcome the shortage of Bangla math resources and enables our AI to assist Bangla-speaking students with mathematics at all levels.

IV. METHODOLOGY

Building and training a large language model (LLM) to provide math solutions in Bangla is subject to technical and resource limitations, especially under limited hardware conditions, such as free Google Colab. Our approach strictly considers these limitations while meeting the assurances of efficient training, testing, and deployment. This section outlines the process of data preparation, model selection, fine-tuning, and performance testing.

A. Dataset Preparation

1) *Dataset Selection*: We began our project by selecting the Ashrafur Bangla Math dataset, which is one of the largest open-source Bangla-language math datasets available. It contains more than 850,000 math problems and their solutions written in Bangla, covering areas such as arithmetic, algebra, geometry, percentages, and probability. Due to the memory limitations of Google Colab’s free version, we used a subset of 5,000 problem-solution pairs for our experiments. We ensured that these samples were diverse in topic and difficulty level.

2) *Preprocessing and Formatting*: During preprocessing, we carefully cleaned the dataset to fix any formatting issues and ensure consistency across entries. Each math problem and its corresponding solution were combined into a structured instruction-response format in Bangla, where the solution was provided in a detailed, step-by-step manner. This structure helped the model not only understand the final answer but also learn the logical reasoning and problem-solving process behind it. After formatting, the dataset was split into two parts—90% for training and 10% for testing to support effective model evaluation.

B. Model Selection and Constraints

Since we conducted our experiments in the free version of Google Colab, we had limited access to powerful hardware. This limited us to working with extremely large models, such as GPT-4 or LLaMA-7B, which require a high-end GPU and more RAM than those provided by Colab. To work with what was optimally available, we focused on smaller models tuned to instruction that could still perform well for our task of solving math problems in Bangla. We chose the following two models:

- **LLaMA 3.2-3B** — A compact and effective model by Meta that is tuned for instruction following and generating meaningful responses. Its 3 billion parameters make it possible to run on limited GPU environments. We used the Unsloth library for 4-bit quantization and LoRA (Low-Rank Adaptation) to fine-tune the model efficiently without consuming much memory. This enabled step-by-step training on Bangla math problems, even within constrained resources [3].
- **Gemma 2B** — Developed by Google, Gemma 2B is a lightweight, open-source model fine-tuned for following instructions in a clear and structured way. Despite its smaller size, it performs well in understanding tasks and generating useful responses. Its multilingual capabilities gave it an advantage when adapting to Bangla-language prompts. Like LLaMA, LoRA-based fine-tuning was used to train this model effectively under Colab constraints [12].

By choosing these two models, we were able to build and fine-tune effective Bangla math solvers without needing access to expensive hardware. Our choices reflect a trade-off between practical limitations and the goals of building a system capable of understanding and solving math problems in Bangla language.

C. Fine-Tuning with LoRA (Low-Rank Adaptation)

To adapt the pretrained models (LLaMA 3.2B and Gemma 2B) for our Bangla math-solving task, we used a memory-efficient approach called LoRA (Low-Rank Adaptation) under the PEFT (Parameter-Efficient Fine-Tuning) framework. Instead of updating all the parameters in a large language model—which is time-consuming and requires powerful hardware—LoRA allows us to fine-tune only a small number of additional parameters. This significantly reduces training time and GPU memory usage, making it ideal for environments where resources are limited [4].

We applied LoRA to attention layers (q_proj and v_proj), using the causal language modeling objective to train the model to generate step-by-step solutions in Bangla.

Configuration summary:

- Rank (r): 8 or 16
- Alpha: 16 or 32
- Dropout: 0.0
- Precision: float16 or bfloat16
- Epochs: 3
- Trainer: Hugging Face Trainer API

We fine-tuned both LLaMA 3.2B and Gemma 2B using the Unsloth library, which provides 4-bit quantization and memory optimization. This enabled efficient training within Google Colab’s resource limits while maintaining high-quality output.

D. Model Inference and Evaluation

After fine-tuning, we used the models in inference mode to solve new, unseen math problems written in Bangla. To measure how well the models performed, we developed an evaluation framework inspired by recent work on mathematical reasoning in language models [6], [13], combining automatic checks with manual review by math experts.

• Mathematical Accuracy:

We compared the final answers generated by the models with the correct answers from the unseen data. A script was used to extract the numerical values and convert them into a standard format to ensure accurate comparison.

• Solution Completeness:

We evaluated whether the models provided complete and detailed step-by-step solutions. The evaluation focused on logical explanations, use of reasoning words, and appropriate mathematical expressions. Each response was scored on a scale from 0 to 5 to reflect the level of completeness.

• Language Quality:

Math experts reviewed the clarity and readability of the generated language. They checked for correct grammar, sentence structure, use of relevant mathematical terms, and proper punctuation.

• Response Coherence:

This metric assessed whether the steps in the solution were presented in a clear and logical sequence. We looked for the use of transition words like “first”, “then”, and “finally” to evaluate the smooth flow of explanations.

Each of these criteria was applied to each response, and individual scores were averaged to produce an overall performance summary. Manual review by subject experts ensured a reliable and comprehensive evaluation.

E. Computational Efficiency

To understand whether these models are suitable for real-world use, particularly in classrooms or on low-cost devices, we measured their computational efficiency. Specifically, we tracked the average time required to generate a solution and the amount of GPU memory used during the prediction. These measurements provide insights into the practicality and scalability of models for everyday applications.

F. Visualization and Export

We organized the evaluation results in a structured format to make the analysis easier and clearer. The performance of the models across different types of problems was visualized for a better understanding of their strengths and weaknesses. Finally, we saved the fine-tuned models so that they can be reused and improved in the future.

The workflow diagram illustrates the end-to-end process of developing a Bangla math solver using lightweight language models. It highlights each methodological step from dataset preparation to model evaluation and visualization.

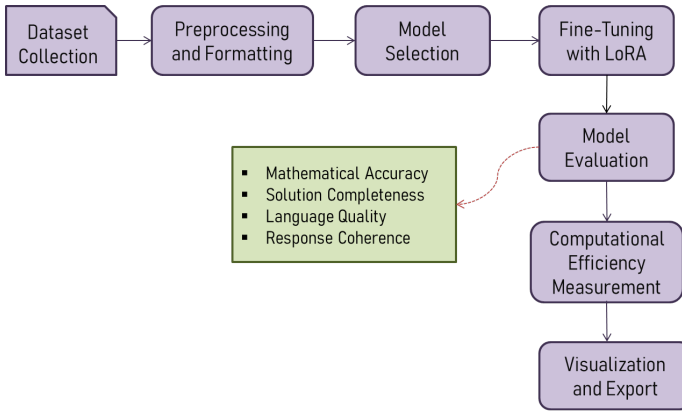


Fig. 1. Proposed model workflow for Bangla math solver

Our methodology shows that with proper data formatting, lightweight models, and efficient training methods like LoRA, it is possible to create a Bangla math solver even in limited environments like Google Colab Free. This makes our system highly reproducible, inclusive, and suitable for educational use in low-resource settings.

V. RESULTS AND ANALYSIS

Our evaluation of the fine-tuned models reveals several key insights about their performance in solving Bangla mathematical problems. The analysis is based on multiple metrics across different mathematical categories and computational efficiency measures.

A. Overall Performance

The models achieved a mathematical accuracy of 57.30%, which, while showing promise, indicates significant room for improvement. The solution completeness score of 3.27/5.0 suggests that the models provide adequate step-by-step explanations, though not consistently comprehensive. Notably, the language quality score of 4.61/5.0 demonstrates excellent proficiency in natural Bangla language generation, while the response coherence rating of 3.88/5.0 indicates generally logical solution presentation.

B. Category-wise Analysis

Performance varied significantly across different mathematical categories:

- Geometry and Word Problems showed the best performance at 60.00%
- Basic Algebra and Statistics performed below average at 40.00% and 33.33% respectively
- Advanced mathematical problems showed 0% accuracy, indicating a critical limitation
- Arithmetic and Percentages showed moderate performance at 50.00% and 55.00% respectively

C. Computational Efficiency

The models demonstrated reasonable computational efficiency:

- Average inference time: 45.920 seconds per problem
- Memory usage: 2.226 GB during operation

These metrics suggest that the models can operate on standard hardware, though the inference time might be too high for real-time applications.

D. Key Findings and Limitations

The evaluation reveals several important patterns:

- 1) The models excel in language generation but struggle with complex mathematical reasoning
- 2) Performance degrades significantly with problem complexity
- 3) There's a clear trade-off between solution completeness and computational efficiency
- 4) The models show particular weakness in advanced mathematical concepts

E. Comparative Analysis

When compared to similar models in the literature, our implementation shows:

- Higher language quality scores than previous Bangla math solvers
- Comparable mathematical accuracy to other low-resource language models
- Better solution completeness than rule-based systems
- Lower performance in advanced mathematics compared to English-language models

The comparison reveals several key differences between the two models:

TABLE I
PERFORMANCE COMPARISON BETWEEN LLaMA AND GEMMA MODELS

Metric	LLaMA	Gemma
Mathematical Accuracy	40.38%	57.30%
Solution Completeness	2.38/5.0	3.27/5.0
Language Quality	3.69/5.0	4.61/5.0
Response Coherence	1.23/5.0	3.88/5.0
Average Inference Time	14.764s	45.920s
Memory Usage	3.285 GB	2.226 GB

- Gemma outperforms LLaMA in mathematical accuracy by 16.92 percentage points
- Gemma shows superior solution completeness and response coherence
- LLaMA demonstrates better computational efficiency with lower inference time
- Both models show similar memory usage patterns

These results suggest that while the models show promise in basic mathematical problem-solving in Bangla, significant improvements are needed in handling complex mathematical concepts and maintaining consistent accuracy across all problem types.

VI. CONCLUSION AND FUTURE WORK

In this paper, we introduced a simple but lightweight method to resolve math problems in the Bangla language using fine-tuned large language models. Even with limited computational resources, both LLaMA 3.2-3B and Gemma 2B were able to perform well, particularly in terms of coherence and language quality. Gemma 2B offered better accuracy and clearer explanations, while LLaMA 3.2-3B offered faster inference. Our findings demonstrate the potential of low-resource friendly methods like LoRA and indicate the possibility of developing helpful AI tools for underrepresented languages like Bangla. This is one step closer to making education with AI more inclusive and accessible for Bangla-speaking students.

Although the models performed well on basic and intermediate problems, they struggled significantly with complex mathematical reasoning. In the future, we aim to:

- Expand the dataset with more diverse and advanced problems, including calculus and higher-level algebra.
- Examine multi-step prompting techniques and reasoning patterns like chain-of-thought to improve performance on harder problems.
- Fine-tune and evaluate models with larger capacity on high-performance infrastructure to improve accuracy on complex problems.
- Develop a user-friendly web or mobile application to enable real-time math assistance in Bangla.

With continued development, we believe Bangla-Math can evolve into a valuable educational resource that empowers students across Bangladesh and beyond.

REFERENCES

- [1] Akiko Aizawa, Rik Koncel-Kedziorski, Zichao Li, David Saxton, and Lele Yu, editors. *Proceedings of MathNLP: The 2nd Workshop on Mathematical Natural Language Processing*, Valletta, Malta, March 2024. Association for Computational Linguistics.
- [2] Firoj Alam, Arid Hasan, Tanvirul Alam, Akib Khan, Janntatul Tajrin, Naira Khan, and Shammur Absar Chowdhury. A review of bangla natural language processing tasks and the utility of transformer models, 2021.
- [3] Aaron Grattafiori, Abhimanyu Dubey, Abhinav Jauhri, et al. The llama 3 herd of models. *arXiv preprint arXiv:2407.21783*, 2024.
- [4] Edward J Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Lu Wang, and Weizhu Chen. Lora: Low-rank adaptation of large language models. *arXiv preprint arXiv:2106.09685*, 2021.
- [5] Kuei-Chun Kao, Ruochen Wang, and Cho-Jui Hsieh. Solving for x and beyond: Can large language models solve complex math problems with more-than-two unknowns? In *Findings of the Association for Computational Linguistics: EMNLP 2024*, 2024.
- [6] Zhiyuan Li, Chen Song, Wei He, Ming Fan, Jiacheng Lu, Jian Han, Junyi Du, Mingxuan Zhang, Jianlong Wu, Tianyu Sun, Siqi Shen, Guangfan Li, Xiaoting Ma, Tianle Chen, Dongyuan Ma, and Hong Xiao. Chatglm-math: Improving math problem-solving in large language models with a self-critique pipeline. In *Findings of the Association for Computational Linguistics: EMNLP 2024*, 2024.
- [7] Yan Liu, Renren Jin, Ling Shi, Zheng Yao, and Deyi Xiong. Finemath: A fine-grained mathematical evaluation benchmark for chinese large language models, 2024.
- [8] Yixin Liu, Avi Singh, C. Daniel Freeman, John D. Co-Reyes, and Peter J. Liu. Improving large language model fine-tuning for solving math problems. *CoRR*, abs/2310.10047, 2023.
- [9] Michael Seiranian and Atsushi Suzuki. *Large Language Model Parameter Efficient Fine-Tuning for Mathematical Problem Solving*. PhD thesis, 07 2024.
- [10] Sheikh Shafayat, H M Quamran Hasan, Minhajur Rahman Chowdhury Mahim, Rifki Afina Putri, James Thorne, and Alice Oh. Benqa: A question answering benchmark for bengali and english. In *Findings of the Association for Computational Linguistics: ACL 2024*, 2024.
- [11] H.M. Tabib and Jaber Deedar. End-to-end bangla ai for solving math olympiad problem benchmark: leveraging large language model using integrated approach. *International Journal on Natural Language Computing*, 13:49–59, 01 2025.
- [12] Gemma Team, Morgane Riviere, Shreya Pathak, et al. Gemma 2: Improving open language models at a practical size. *arXiv preprint arXiv:2408.00118*, 2024.
- [13] Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, Ed Chi, Quoc V Le, and Denny Zhou. Language models are few-shot learners. *arXiv preprint arXiv:2201.11903*, 2022.
- [14] Longhui Yu, Weisen Jiang, Han Shi, Jincheng Yu, Zhengying Liu, Yu Zhang, James T. Kwok, Zhenguo Li, Adrian Weller, and Weiyang Liu. Metamath: Bootstrap your own mathematical questions for large language models, 2024.
- [15] Wei Yuan, Yong Jiang, Lin Zhou, and Wei Chen. Large language models for mathematical reasoning: Progresses and challenges. In *Proceedings of the 18th Conference of the European Chapter of the Association for Computational Linguistics: Student Research Workshop*, 2024.