Two-Stage Training with Reinforcement Learning for Vietnamese Financial Numerical Reasoning

Nhat-Truong Dinh[1,2], Thanh-Trung Ngo[1,2], Quoc-Bao Trinh[1,2], Duc-Vu Nguyen[1,2]
[1]University of Information Technology, Ho Chi Minh City, Vietnam
[2]Vietnam National University, Ho Chi Minh City, Vietnam
{22521575, 22521560, 22520125}@gm.uit.edu.vn **Correspondence:** vund@uit.edu.vn

Abstract

Numerical reasoning plays a crucial role in the financial sector, requiring both accurate answers and clear explanations. FinQA is a key benchmark for this task, yet Vietnamese financial reasoning remains underexplored. In this study, we present a two-phase training pipeline for Vietnamese financial QA in the VLSP 2025 Challenge on Numerical Reasoning QA, combining supervised fine-tuning and Group Relative Policy Optimization. Our model achieves 84% execution accuracy and 85% program accuracy on the public test set, and 74% and 69% on the private test set, ranking fourth in the constrained track and third in the unconstrained track. Quantization further improves efficiency in model size and inference speed, demonstrating the effectiveness of our approach for Vietnamese financial numerical reasoning.

1 Introduction

Financial assessment is of paramount importance in contemporary finance, particularly within domains such as Corporate Finance, Investment and Financial Markets, and Banking and Financial Institutions. Nonetheless, the process is inherently susceptible to significant risks, including potential losses and inaccurate valuation of assets, largely due to limitations in numerical computational precision. As workloads increase, human accuracy in mathematical reasoning may decline, especially in tasks that require sustained attention, such as enterprise analysis, probabilistic assessment, and the calculation of long-term profits or cash flows from financial reports, which are typically extensive and information-rich. These problems are not only prevalent globally but also in Vietnam, creating a demand for a system that can automate this work.

A notable solution to address these challenges is the APPOLO (Sun et al., 2022) training approach, which integrates a retrieval module, as introduced by Chen et al. (2021), with a generator module. The retrieval module employs a sequence-pair classification model to identify and extract salient or pertinent facts for subsequent processing, thereby avoiding the need to analyze entire long-form documents. Upon completion of the retrieval phase, the selected facts and corresponding numerical tables are provided to the generator module, which then synthesizes the relevant information to generate a reasoning-based mathematical program.

Nevertheless, there is a paucity of research investigating the application of this technique to Financial Question Answering datasets in the Vietnamese language. Consequently, the performance of such systems and the presence of particularly challenging cases within the Vietnamese context remain uncertain. These considerations motivated us to undertake an experimental study addressing this complex task in the context of Vietnamese financial data.

This task, analogous to the FinQA dataset, provides the model with the context D and table T. The model processes these inputs and generates a programmatic solution $G = \{w_0, w_1, w_2, \ldots\}$, where each w_i represents a token segment of the program. The generated program is then executed to yield the final numerical answer (Chen et al., 2021), and the probability of the answer is given by $P(A \mid T, E, Q) = \sum_i P(G_i \mid T, E, Q)$, where $\{G_i\}$ denotes all correct programs that evaluate to the answer.

In this study, we adopt a two-stage training methodology for our language model. The first stage consists of Supervised Fine-Tuning (SFT) (Dong et al., 2023), during which the model learns from annotated examples to build a foundational understanding of the task. The second stage employs Reinforcement Learning to further improve predictive accuracy and reduce execution program errors. To enhance computational efficiency during both training and inference, we use a quantized

Listing 1 Each sample in the dataset is represented as a dictionary with six keys: "pre_context" (the text before the numerical table), "table" (the numerical table itself), "post_context" (the text after the table), "question" (the numerical question related to the table), "solution_program" (the ground-truth calculation program to answer the question), and "exec_answer" (the result of executing the program).

```
sample = {
    "pre_context": (
        "mục lục tập đoàn cdw và các công ty con thuyết minh báo cáo tài chính "
        "hợp nhất chi phí tài chính trả trước chi phí tài chính trả trước, "
        "chẳng hạn như phí bảo lãnh phát hành, tư vấn tài chính, phí chuyên gia "
        "và các khoản phí tư . . . "
    ),
    "table": [
        ["                ( đơn vị: triệu )                ", "chênh lệch tỷ giá hối đoái", "Lỗ toàn diện khác lũy kế"],
        ["năm kết thúc ngày 31 tháng 12 năm 2015", "      $ -61.1 ( 61.1 )     ", "      $ -61.1 ( 61.1 )     "],
        ["năm kết thúc ngày 31 tháng 12 năm 2014", "      $ -16.6 ( 16.6 )     ", "      $ -16.6 ( 16.6 )     "],
        ["năm kết thúc ngày 31 tháng 12 năm 2013", "      $  -6.3  (  6.3 )     ", "      $  -6.3  (  6.3 )     "]
    ],
    "post_context": (
        "ghi nhận doanh thu công ty là một kênh phân phối chính cho một nhóm lớn "
        "các nhà cung cấp và nhà cung ứng, bao gồm các nhà sản xuất thiết bị gốc (“oems”), "
        "các nhà phát hành phần mềm và các nhà phân phối bán buôn. công ty ghi nhận "
        "doanh thu từ các . . . "
    ),
    "question": "Mức lỗ chuyển đổi ngoại tệ tối thiểu là bao nhiêu, tính bằng triệu?",
    "solution_program": "table_max(năm kết thúc ngày 31 tháng 12 năm 2013, none)",
    "exec_answer": "-6.3"
}
```

model instead of a full-precision counterpart. The experimental data consist of the FinQA dataset, translated into Vietnamese, along with financial data extracted from publicly available Vietnamese financial reports from 2020 to 2025. Listing 1 shows a representative sample. This integrated dataset provides a solid foundation for evaluating the efficacy of our methodology in Vietnamese financial question answering.

Our system achieved fourth place in the Numerical Constrained Track and third place in the Numerical Unconstrained Track at the VLSP 2025 Challenge on Numerical Reasoning QA. Nevertheless, certain challenges remain. Specifically, some reasoning cases continue to yield incorrect answers, and the system has not yet effectively removed irrelevant facts from the context in sample questions.

2 Related Work

Recently, Chen et al. (2021) introduced FinQA, a numerical reasoning dataset focused on the financial domain, presenting a considerable challenge for contemporary language models. This expert-annotated dataset comprises 8,281 question-answer pairs, each accompanied by a lengthy document, a numerical table, a precise numerical answer, and a gold-standard reasoning program that explains the solution process. The FinQA task provides the natural language processing (NLP) community with a valuable opportunity to evaluate the effectiveness of current pre-trained models in addressing complex, domain-specific reasoning problems. Furthermore, extensive experiments across various baseline models reveal a significant gap between model performance and that of human experts, highlighting the need for ongoing research and development in this area.

The work by Sun et al. (2022) advances the retriever-generator paradigm for long-form numerical reasoning by introducing several novel techniques. Specifically, a number-aware negative sampling strategy is proposed for retriever training, enabling the model to better prioritize and distinguish numerical facts. At the generator stage, the authors employ target program augmentation and consistency-based reinforcement learning, which facilitate the exploration of consistent program spaces and improve execution accuracy. Their contributions include: (1) the introduction of a number-aware negative sampling method, which proves effective for retriever training in long-form numerical reasoning tasks; (2) the application of consistency-based reinforcement learning and target program augmentation in generator training, resulting in increased program accuracy and overall

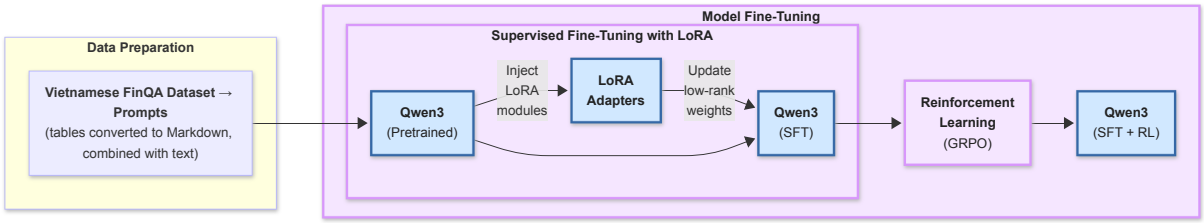

Figure 1: Overview of our two-phase training approach: A pretrained Qwen3 model is first fine-tuned using Supervised Fine-Tuning (SFT) with LoRA on prompts from the Vietnamese FinQA dataset. The resulting model is then further trained using a Reinforcement Learning (RL) phase with the GRPO approach.

performance improvements; and (3) achieving new state-of-the-art results, with execution and program accuracy scores of 0.7247 and 0.6801, respectively, on the FinQA leaderboard.

Group Relative Policy Optimization (GRPO) (Shao et al., 2024) has been introduced as an alternative to Proximal Policy Optimization (PPO) (Schulman et al., 2017), primarily to reduce dependence on value function approximation, which can introduce instability in reinforcement learning tasks. Instead of estimating a separate value function for each input, GRPO computes the average reward from multiple outputs generated from the same input and uses this as a baseline for comparison. This intra-group reward comparison provides a more stable and reliable estimation of the advantage function, improving the robustness of policy updates. GRPO also incorporates direct policy regularization through Kullback-Leibler (KL) (Van Erven and Harremos, 2014) divergence with respect to a reference model, which constrains policy updates and contributes to a more streamlined and computationally efficient optimization process compared to PPO. These design choices make GRPO particularly effective for tasks requiring stable learning from multiple sampled outputs and precise policy updates.

The introduction of Qwen3 (Yang et al., 2025) marks a significant development in the evolution of large language models. This family of open-weight models spans a broad range of parameter sizes, from 0.6B to 235B, and incorporates both dense and Mixture-of-Experts (MoE) (Zhou et al., 2022) architectures. Notably, the flagship MoE variant activates only a subset of its parameters during inference, effectively balancing computational efficiency with model performance. Qwen3 (Yang et al., 2025) models achieve state-of-the-art results across domains, showcasing scalable, efficient architectures for diverse tasks.

3 Method

Our method for Vietnamese financial question answering employs a two-phase training framework combining Supervised Fine-Tuning (SFT) and GRPO (Shao et al., 2024), with Markdown-preprocessed tables and long-form documents as input, and the overall pipeline is illustrated in Figure 1.

3.1 Data Preprocessing

The preprocessing of the translated FinQA dataset, which includes both long-form documents and tables, presented significant challenges. This is primarily due to the inability of language models to natively interpret table structures within input data. To address this, we explored several preprocessing techniques, such as converting list-based tables into string representations while maintaining the original table format (see Listing 2) and encoding tables in JSON format. However, the JSON approach proved inadequate due to the heterogeneous nature of table formats present in the dataset. Consequently, we adopted Markdown conversion for tables (see Listing 3), which effectively preserved structural fidelity and minimized format-related issues. In combination with pre-context and post-context, this process typically results in a long-form document that serves as the model input.

Not all information in these long-form documents is relevant for answering the question, and some content may introduce noise. To mitigate this, we applied BM25 (Robertson et al., 2009) to assign relevance scores to each sentence with respect to the corresponding question. We experimented with thresholds ranging from 20% to 60%, observing that lower thresholds led to reduced model performance. As a result, we retained the top 60% of sentences for further analysis. Despite this targeted filtering, models utilizing the full document context consistently outperformed those relying solely

Listing 2 The financial data is represented in Python, with each list representing a row, starting with the header row.

```
[' ngày ', ' pmi ', 'nhóm công ty cùng ngành của pmi (1)', 'chỉ số s&p 500'],
['ngày 31 tháng 12 năm 2012', '$100.00', ' $100.00 ', ' $100.00 '],
['ngày 31 tháng 12 năm 2013', '$108.50', ' $122.80 ', ' $132.40 '],
['ngày 31 tháng 12 năm 2014', '$106.20', ' $132.50 ', ' $150.50 '],
['ngày 31 tháng 12 năm 2015', '$120.40', ' $143.50 ', ' $152.60 '],
['ngày 31 tháng 12 năm 2016', '$130.80', ' $145.60 ', ' $170.80 '],
['ngày 31 tháng 12 năm 2017', '$156.80', ' $172.70 ', ' $208.10 ']
```

Listing 3 The financial data, formatted as a Markdown-style table for readability, corresponds to the Python list in Listing 2.

```
| ngày | pmi | nhóm công ty cùng ngành của pmi (1) | chỉ số s&p 500
|---|---|---|---
| ngày 31 tháng 12 năm 2012 | $100.00 | $100.00 | $100.00
| ngày 31 tháng 12 năm 2013 | $108.50 | $122.80 | $132.40
| ngày 31 tháng 12 năm 2014 | $106.20 | $132.50 | $150.50
| ngày 31 tháng 12 năm 2015 | $120.40 | $143.50 | $152.60
| ngày 31 tháng 12 năm 2016 | $130.80 | $145.60 | $170.80
| ngày 31 tháng 12 năm 2017 | $156.80 | $172.70 | $208.10
```

on BM25-based selection. This outcome is largely due to the inability of BM25 to capture semantic relationships between sentences, which may result in the removal of crucial information. Therefore, our final methodology preserves the complete document context throughout the pipeline, ensuring that all potentially relevant information is available for reasoning.

3.2 Prompt Design

Our prompting methodology employs a highly structured approach in which the context, tabular data, and question are explicitly defined to guide the model through a step-by-step reasoning process. The prompt template (Listing 4) is carefully designed to ensure that the model has access to all necessary information to reason through complex financial questions accurately. By presenting each component clearly, including context before the table, the table itself, context after the table, the question, the solution program, and the execution result, this template ensures that the model can perform systematic reasoning over the financial data and generate precise, correct solution programs.

3.3 Supervised Fine-Tunning

In the initial phase of training, we employ supervised fine-tuning (Shengyu et al., 2023; Parthasarathy et al., 2024) to adapt our foundational language model, Qwen (Bai et al., 2023), to the specific requirements of the Vietnamese Financial QA dataset. At this stage, the model lacks prior knowledge of the Financial QA task or its associated output format. Supervised fine-tuning enables the model to accurately interpret prompts, effectively process long-form documents and tabular data, and generate precise responses. Given

Listing 4 The function build_financial_prompt constructs a financial reasoning prompt by combining six components: "pre_context" (text before the table), "table_content" (the numerical table), "post_context" (text after the table), "question" (numerical question), "solution_program" (ground-truth solution), and "exec_answer" (execution result). These inputs correspond to the data sample shown in Listing 1.

```
def build_financial_prompt(pre_context,
    table_content, post_context, question,
    solution_program, exec_answer):
    """
    Build a financial reasoning QA prompt.
    """
    prompt = (
        f"You are a financial reasoning assistant. "
        f"Given the context text, the table "
        f"with financial data, "
        f"and the question below, your task is "
        f"to reason through the problem "
        f"step by step and generate "
        f"a solution program. "
        f"The program should detail "
        f"all intermediate steps "
        f"and provide the final"
        f"numerical result.\n\n"
        f"Pre Context:\n{pre_context}\n\n"
        f"Table:\n{table_content}\n\n"
        f"Post Context:\n{post_context}\n\n"
        f"Question:\n{question}\n\n"
        f"Solution Program:\n{solution_program}\n\n"
        f"Exec Answer:\n{exec_answer}"
    )
    return prompt
```

the substantial computational resources and time required to fine-tune a full-parameter model, we instead utilize LoRA-based (Hu et al., 2022) fine-tuning with hyperparameters set to rank $R = 128$ and LoRA alpha $= 128$. Additionally, we select a quantized version of the Qwen3 model, reducing the parameter count from the original 14 billion to 8.7 billion parameters, further optimizing efficiency without compromising performance.

3.4 Group Relative Policy Optimization Training Approach

The second phase of training utilizes Group Relative Policy Optimization (GRPO). Prior supervised fine-tuning (SFT) is essential, as GRPO training is both time- and resource-intensive. Supervised pretraining provides the model with task-specific knowledge, reducing the risk of random action selection and accelerating convergence during policy optimization. Within this framework, the reward (or verifier) function plays a critical role in shaping system performance by determining when and to what extent the model is penalized or rewarded. We define our reward function, inspired by (Chen et al., 2021):

$$R(E(p), y^*) = \begin{cases} -2, & \text{if } E(p) = \varnothing \\ 1, & \text{if } E(p) = y^*, \\ 0, & \text{otherwise} \end{cases} \quad (1)$$

where p denotes the program being evaluated, $E(p)$ is the execution function that returns the output of p or $\varnothing$ if execution fails, y^* represents the expected (correct) output, and $R(E(p), y^*)$ is the reward assigned based on the output of p. The reward function evaluates the model's output using the verifier: if the verifier encounters an error or fails to produce a result, indicating that the model's prediction is in an incorrect format, an immediate penalty of -2 is applied. If the output successfully passes the verifier and yields a numerical value, it is compared to the ground-truth label, with a reward of +1 assigned for an exact match and 0 otherwise.

4 Experiment

4.1 Data Statistics

The dataset utilized in this study comprises a combination of the translated FinQA dataset and Vietnamese financial reports, resulting in a total of 4,074 samples. Each sample includes a context document, with the maximum document length exceeding 7,000 tokens. This considerable length presents challenges for effective context comprehension and introduces potential noise, which may hinder the model's ability to identify relevant facts. The dataset is partitioned into training, validation, and test sets, following a split ratio of 0.73/0.14/0.12, corresponding to 2,993, 584, and 497 samples, respectively[1].

4.2 Experimental Setup

The initial preprocessing stage involved converting tables from their original format to Markdown to ensure structural consistency. Following preprocessing, the dataset was used in the first training phase with supervised fine-tuning, during which the Qwen3 model was trained to develop task-specific knowledge and reduce computational resource requirements, thereby accelerating convergence for the subsequent GRPO phase. The hyperparameters for supervised fine-tuning were set to 5 epochs and a total batch size of 64 per device, with a learning rate of 2e-4 using the Adam optimizer, weight decay of 0.01, and cross-entropy loss.

In the GRPO training phase, the hyperparameters were set as min_p=0.1, top_p=1.0, top_k=-1, learning_rate=5e-6, weight_decay=0.01, optim="adamw_8bit", warmup_ratio=0.1, with a per-device training batch size of 32, num_train_epochs=1, num_generations=4, and temperature=1.0. During inference, the hyperparameters were temperature=0.7, top_p=0.95, top_k=20, with a maximum of 4098 new tokens. Both training and inference were conducted on a single A100 SMX4 40 GB GPU using PyTorch and the UnSloth[2] framework. The first training phase took only 3-4 hours, whereas the second phase could take up to 10 hours per epoch.

4.3 Evaluation Metric

To evaluate program accuracy over a dataset of N samples, accuracy is calculated as

$$\text{Accuracy}(T, P) = \frac{1}{N} \sum_{i=1}^{N} \mathbf{1}\big(Rp(P_i) = Rp(T_i)\big), \quad (2)$$

where T_i is the ground-truth program at index i, P_i is the predicted program at index i, Rp is a function

[1] VLSP 2025 Challenge on Numerical Reasoning QA: https://vlsp.org.vn/vlsp2025/eval/numqa

[2] Fine-tuning and Reinforcement Learning for LLMs: https://github.com/unslothai/unsloth

ID	Strategy	Execution Accuracy	Program Accuracy
(1)	Markdown + SFT (5 epochs)	65%	61%
(2)	Markdown + SFT (5 epochs) + GRPO	85%	84%
(3)	Markdown + BM25 Filtering + SFT	59%	55%
(4)	Markdown + BM25 Filtering + SFT + GRPO	64%	60%
(5)	Original Table Form + SFT	60%	57%
(6)	Original Table Form + BM25 Filtering + SFT	58%	54%

Table 1: Performance comparison between methods on the public test set. *Markdown* refers to tables presented in a preprocessed Markdown format, while *Original Table Form* denotes the unprocessed tables as shown in the sample. *SFT* stands for the supervised fine-tuning approach, and *GRPO* represents the Group Relative Policy Optimization training approach.

that replaces all program parameters with symbolic variables, $\mathbf{1}(\cdot)$ is the indicator function returning 1 if the condition is true and 0 otherwise, and N is the total number of programs in the dataset.

Similarly, execution accuracy is computed as

$$\text{Acc2} = \frac{1}{N}\sum_{i=1}^{N}\mathbf{1}\Big[\,\text{Exec}(P_i) = y_i^*\Big], \qquad (3)$$

where P_i is the program generated by the model for example i, $\text{Exec}(P_i)$ is the numeric result obtained after executing the predicted program, y_i^* is the ground-truth numeric answer, and N is the total number of examples in the dataset. The indicator function $\mathbf{1}[\cdot]$ equals 1 if the condition inside is true, and 0 otherwise.

4.4 Main Result

Table 1 presents a comparative analysis of the training strategies evaluated in this study. The pipeline that employed markdown-formatted tables, retained only the top 60% of facts, and was trained with supervised fine-tuning (SFT) (3) achieved the lowest performance, with 59% execution accuracy and 55% program accuracy. This decline is primarily due to BM25 filtering, which removed essential information required to answer numerical questions. When BM25 filtering was removed (1), performance improved to 65% execution accuracy and 61% program accuracy. Comparing the original table format (5) with the markdown format (1), using the original table with SFT achieved 60% execution accuracy and 57% program accuracy. Applying markdown conversion resulted in a relative improvement of +5% in execution accuracy and +4% in program accuracy. With a pipeline that relied solely on markdown tables and SFT (1), the model reached 65% execution accuracy and 61% program accuracy.

Introducing a second training phase with Group Relative Policy Optimization (GRPO) (2) further boosted performance to 85% execution accuracy and 84% program accuracy, corresponding to improvements of +20% and +23%, respectively. This substantial gain stems from the fact that GRPO does not optimize for surface-level similarity between the predicted and gold programs; instead, it emphasizes both syntactic correctness and the accuracy of numerical results obtained when executing the generated programs. The BM25 filtering strategy consistently resulted in lower performance across all settings. Comparing (3) and (4) with (1) and (2), as well as (5) and (6), shows a clear drop in both execution and program accuracy when BM25 filtering is applied. These results suggest that simply truncating input via BM25 is not an effective preprocessing method. Therefore, a more advanced filtering strategy, such as using a cross-encoder for evidence selection, is necessary to better preserve reasoning-relevant information.

4.5 Error Analysis

A total of 78 error cases out of 497 predictions were identified in the test set. Of these, 57% were categorized as logical errors, referring to instances in which the predicted program executed but produced results inconsistent with the reference answers. Errors involving incorrect unit usage accounted for 43% of cases, typically manifesting as outputs such as 60% instead of 0.6; these errors are generally straightforward to address. Errors resulting from incorrect parameter order represented 10% of cases, most commonly occurring in subtraction and division operations. Finally, 6% of cases produced the correct numerical result, but the predicted program's reasoning differed from that of the reference implementation. Errors related to units, parameter order, or alternative program logic

may be mitigated by refining the reward function during the GRPO training phase.

5 Discussion

Our pipeline indicates that a considerable performance gap persists in this task, highlighting significant opportunities for further improvement. Moreover, implementing a robust mechanism to filter irrelevant factual information from the input is essential. Given the lengthy convergence period of the GRPO approach, greater investment of time and computational resources is warranted for training. It is also advisable to explore various augmentation techniques, such as introducing greater variability in numerical program transformations. Error analysis reveals that some calculation functions are missing from the training data, which may further hinder model accuracy. Finally, alternative evaluation strategies should be considered, as many mathematically valid reasoning programs may employ different calculation functions while still producing correct results, a nuance not fully captured by the current evaluation system.

6 Conclusion

In this study, we introduced a novel training approach for financial numerical reasoning question answering on a Vietnamese dataset, employing a two-phase process consisting of supervised fine-tuning and Group Relative Policy Optimization. Our pipeline achieved an execution accuracy of 84% and a program accuracy of 85% on the public test set, and 74% and 69% on the private test set, respectively. This performance ranked fourth in the constrained numerical reasoning track and third in the unconstrained track. Future research should further explore and experiment with alternative approaches to improve performance in this domain.

Limitations

Despite these findings, the current reward mechanism remains suboptimal, as it does not address all scenarios present in the dataset, particularly cases involving unit mismatches, which consequently constrains model performance. Our current filtering approach based on BM25 is insufficient for capturing semantic relevance and can result in the removal of important information. A more effective filtering mechanism, such as a cross-encoder model, is needed to replace BM25. Additionally, the absence of certain calculation functions in the training data further limits the model's ability to generalize effectively. It is also necessary to compare with other pipelines trained on Vietnamese, such as Apollo. However, we did not experiment with pipelines related to reinforcement learning, since the training process is costly in terms of both time and computational resources.

Acknowledgement

This research was supported by The VNUHCM-University of Information Technology's Scientific Research Support Fund. We thank the anonymous reviewers for their time and helpful suggestions that improved the quality of the paper.

References

Jinze Bai, Shuai Bai, Yunfei Chu, Zeyu Cui, Kai Dang, Xiaodong Deng, Yang Fan, Wenbin Ge, Yu Han, Fei Huang, and 1 others. 2023. Qwen technical report. *arXiv preprint arXiv:2309.16609*.

Zhiyu Chen, Wenhu Chen, Charese Smiley, Sameena Shah, Iana Borova, Dylan Langdon, Reema Moussa, Matt Beane, Ting-Hao Huang, Bryan Routledge, and 1 others. 2021. Finqa: A dataset of numerical reasoning over financial data. *arXiv preprint arXiv:2109.00122*.

Guanting Dong, Hongyi Yuan, Keming Lu, Chengpeng Li, Mingfeng Xue, Dayiheng Liu, Wei Wang, Zheng Yuan, Chang Zhou, and Jingren Zhou. 2023. How abilities in large language models are affected by supervised fine-tuning data composition. *arXiv preprint arXiv:2310.05492*.

Edward J Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, Weizhu Chen, and 1 others. 2022. Lora: Low-rank adaptation of large language models. *ICLR*, 1(2):3.

Venkatesh Balavadhani Parthasarathy, Ahtsham Zafar, Aafaq Khan, and Arsalan Shahid. 2024. The ultimate guide to fine-tuning llms from basics to breakthroughs: An exhaustive review of technologies, research, best practices, applied research challenges and opportunities. *arXiv preprint arXiv:2408.13296*.

Stephen Robertson, Hugo Zaragoza, and 1 others. 2009. The probabilistic relevance framework: Bm25 and beyond. *Foundations and Trends® in Information Retrieval*, 3(4):333–389.

John Schulman, Filip Wolski, Prafulla Dhariwal, Alec Radford, and Oleg Klimov. 2017. Proximal policy optimization algorithms. *arXiv preprint arXiv:1707.06347*.

Zhihong Shao, Peiyi Wang, Qihao Zhu, Runxin Xu, Junxiao Song, Xiao Bi, Haowei Zhang, Mingchuan Zhang, YK Li, Yang Wu, and 1 others. 2024. Deepseekmath: Pushing the limits of mathematical reasoning in open language models. *arXiv preprint arXiv:2402.03300*.

Zhang Shengyu, Dong Linfeng, Li Xiaoya, Zhang Sen, Sun Xiaofei, Wang Shuhe, Li Jiwei, Runyi Hu, Zhang Tianwei, Fei Wu, and 1 others. 2023. Instruction tuning for large language models: A survey. *arXiv preprint arXiv:2308.10792*.

Jiashuo Sun, Hang Zhang, Chen Lin, Xiangdong Su, Yeyun Gong, and Jian Guo. 2022. Apollo: An optimized training approach for long-form numerical reasoning. *arXiv preprint arXiv:2212.07249*.

Tim Van Erven and Peter Harremos. 2014. Rényi divergence and kullback-leibler divergence. *IEEE Transactions on Information Theory*, 60(7):3797–3820.

An Yang, Anfeng Li, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chang Gao, Chengen Huang, Chenxu Lv, and 1 others. 2025. Qwen3 technical report. *arXiv preprint arXiv:2505.09388*.

Yanqi Zhou, Tao Lei, Hanxiao Liu, Nan Du, Yanping Huang, Vincent Zhao, Andrew M Dai, Quoc V Le, James Laudon, and 1 others. 2022. Mixture-of-experts with expert choice routing. *Advances in Neural Information Processing Systems*, 35:7103–7114.

A Appendix

A.1 Example of Training Input Prompt

Listing 5 illustrates the final input format used to train the financial reasoning model. Each input contains the textual context, a structured financial table, and a numerical reasoning question. During training, both the ground-truth program (solution_program) and its execution result (exec_answer) are included; these fields are not available during inference.

Listing 5 Example of a model input prompt for financial reasoning. This input directly corresponds to the arguments of the build_financial_prompt function (Listing 4) and contains pre_context, table_content, post_context, question, solution, and exec_answer. The solution_program and exec_answer fields are included during training but omitted during inference.

```
System:
You are a financial reasoning assistant. Given the context text, the table with financial data, and the question below, your task is to reason through the problem step by step and generate a solution program. The program should detail all intermediate steps and provide the final numerical result.

Pre Context:
Biểu đồ hiệu suất: biểu đồ dưới đây so sánh tổng lợi nhuận tích lũy của cổ đông trên cổ phiếu PMI và các công ty cùng ngành.

Table:
            ngày            |   pmi    | nhóm công ty cùng ngành của pmi (1) | chỉ số s&p 500
----------------------------|----------|-------------------------------------|---------------
ngày 31 tháng 12 năm 2012 | $100.00 |                $100.00              |     $100.00
ngày 31 tháng 12 năm 2013 | $108.50 |                $122.80              |     $132.40
ngày 31 tháng 12 năm 2014 | $106.20 |                $132.50              |     $150.50
ngày 31 tháng 12 năm 2015 | $120.40 |                $143.50              |     $152.60
ngày 31 tháng 12 năm 2016 | $130.80 |                $145.60              |     $170.80
ngày 31 tháng 12 năm 2017 | $156.80 |                $172.70              |     $208.10

Post Context:
(1) Nhóm công ty cùng ngành của PMI được trình bày để so sánh hiệu suất.

Question:
Tỷ lệ tăng trưởng giá cổ phiếu của PMI từ năm 2012 đến 2013 là bao nhiêu?

Solution Program:
subtract(108.50, 100), divide(#0, 100)

Exec Answer:
0.085
```