LoRA-Tuned LLMs for Numerical Reasoning on the FinQA Dataset

Mo Liu(ml751), Serena Wang(yw457), Zhiyi Xie(zx138)

November 2025

Abstract

Financial numerical reasoning requires a model to understand unstructured text, retrieve relevant evidence, and perform arithmetic calculations to answer specific queries. While previous state-of-the-art approaches like FinQANet[1] rely on complex neuro-symbolic pipelines involving separate program generators and executors, this project explores a more streamlined, end-to-end solution. We propose using a general-purpose Large Language Model (Qwen2.5-7B-Instruct [2]), adapted via Low-Rank Adaptation (LoRA), to directly generate numerical answers from financial reports. This approach treats the task as a combination of document classification, language modeling, and numerical reasoning. Our experiments demonstrate that while zero-shot performance is limited (15% accuracy), applying parameter-efficient fine-tuning significantly bridges the gap, achieving 50% execution accuracy. This result highlights the potential of adapted LLMs to replace rigid symbolic architectures in specialized domains.

1 Introduction

1.1 The NLP Problem

Financial numerical reasoning is a challenging task within NLP. Unlike standard QA tasks where answers can often be extracted from the text, FinQA requires models to compute numerical answers using both unstructured text and structured financial tables. This introduces multiple sources of difficulty. First, the model must perform Document Classification and information extraction to identify which sentences and table rows are relevant. Financial reports are long and complex, so missing even a single important row can invalidate the final calculation. Second, the model must engage in Language Modeling to understand financial terminology and interpret the meaning behind the question. Many financial questions rely on implicit domain knowledge that general models may not have. Finally, the model must perform Numerical Reasoning to derive a scalar answer through multiple arithmetic steps. The correct answer is often not explicitly present in the document, so the model must extract numbers, apply the correct operations, and compute the final result. Together, these challenges make FinQA a demanding test of real-world reasoning ability, combining retrieval, understanding, and arithmetic.

1.2 Project Scope and Data

We use the FinQA dataset, a large-scale benchmark of questions sourced from S&P 500 annual reports. This dataset contains textual summaries, financial tables, questions, and numerically derived answers, making it well-suited for evaluating numerical reasoning. Our project follows course requirements, which emphasizes training and evaluating models on real-world data. We propose an end-to-end approach using a LoRA-fine-tuned large language model (Qwen2.5-7B-Instruct). Instead of generating symbolic programs, the model directly reads retrieved evidence and outputs the final numerical answer. We also compare the LoRA-adapted model with a zero-shot baseline to understand how much performance comes from domain adaptation. The zero-shot model reaches 15% accuracy, while LoRA fine-tuning significantly improves performance to 44% (2k subset) and 50% (full data). This comparison highlights the importance of adapting LLMs to financial reasoning tasks.

2 Related Work and Motivation

Traditional approaches such as FinQANet[1] follow a symbolic or neuro-symbolic pipeline. These systems generate a structured program composed of predefined operations, then they compute the answer. Although these approaches offer high accuracy, there are substantial limitations: symbolic grammars must be manually defined, operations outside the DSL cannot be handled, and errors in earlier steps impact the final output. These limitations motivate our shift to a simpler end-to-end LLM approach. Generally speaking, large language models can directly interpret natural language and generate numerical answers without relying on symbolic rules. However, they have their own

challenges: difficulty interpreting tables, limitations considering long contexts, and numeric drift because arithmetic is handled through token prediction rather than exact calculation. To address these issues, we use heuristic retrieval to reduce noise, LoRA fine-tuning to adapt the model to financial terminology and reasoning, and structured prompting to keep output consistent. This improves the model's stability and accuracy to some extent. The increases to 44% and 50% accuracy show that our domain adaptation is crucial for handling the complexity of FinQA.

3 Methodology

Our objective is to evaluate whether a modern instruction-tuned large language model can outperform the traditional FinQA symbolic program-generation pipeline. We compare two systems: a zero-shot baseline using Qwen2.5-7B-Instruct[2] and a LoRA-fine-tuned version trained directly on FinQA question–evidence–answer triples. This section provides all technical details of our modeling pipeline, including retrieval, prompting, and parameter-efficient fine-tuning.

3.1 Zero-Shot LLM Baseline

The zero-shot system serves as the primary control condition in our study. For each FinQA example, the model receives three components as input: an instruction describing the numerical reasoning task, a small set of evidence sentences retrieved from the original report, and the question requiring a numeric answer. Retrieval in our pipeline does not rely on TF–IDF or BM25. Instead, we use the dataset-provided evidence, ensuring the model sees the same information used by the original FinQA system.

The target output format is strictly enforced. The model must produce a string of the form

```
Final answer:  <number>
```

No gradient updates of any kind are applied. All reasoning is performed purely through the pretrained capabilities of Qwen2.5-7B-Instruct. This baseline illustrates several limitations. The model often fails to associate financial terminology with the corresponding operations, misidentifies the relevant numbers in the evidence, struggles with multi-step arithmetic, and sometimes produces unstructured outputs. The execution accuracy is approximately 0.15 on our evaluation slice, confirming that pretrained LLMs do not solve FinQA without domain adaptation.

3.2 LoRA Fine-Tuning

To improve numerical reasoning and alignment with the FinQA task format, we apply Low-Rank Adaptation (LoRA) to Qwen2.5-7B-Instruct. LoRA adds trainable low-rank matrices to selected projection layers in attention and feed-forward modules while keeping the original model parameters frozen. Concretely, each adapted weight matrix is expressed as

$$W_{\text{adapted}} = W_0 + \Delta W, \qquad \Delta W = BA,$$

where W_0 is the frozen base matrix and A and B are the learned low-rank factors. In our experiments, the rank is sixteen. This reduces the number of trainable parameters to roughly ten million, which allows fine-tuning to be conducted efficiently on a single GPU.

3.2.1 Supervision Format

Each FinQA example is converted into a text-to-text training instance. The input sequence contains the instruction, the retrieved evidence, and the question. The target sequence is:

```
Final answer:  <gold_number>
```

The model therefore learns both the required output template and the mapping between evidence and the numerical answer. This direct supervision eliminates the need for symbolic programs or an execution engine and allows the LLM to internalize reasoning patterns implicitly.

3.2.2 Training Configurations

We evaluate two LoRA configurations.

- **FinQA_v0**: trained on a subset of two thousand examples, one epoch, and a maximum sequence length of five hundred twelve tokens. This model achieves an execution accuracy of approximately 0.44.
- **FinQA_v1**: trained on the full six thousand two hundred fifty one training examples, two epochs, and a maximum sequence length of seven hundred sixty eight tokens. This model achieves an execution accuracy of approximately 0.50.

During fine-tuning, the loss decreases in two stages. The model first learns the required output structure, after which it gradually adapts to dataset-specific numerical reasoning patterns. The improvement from the subset model to the full-data model demonstrates the benefit of larger task-specific supervision.

3.3 Evaluation

We follow the official FinQA execution accuracy metric. A prediction is considered correct if the absolute difference between the predicted number and the gold answer is no greater than 10^{-2}. We evaluate all models on a randomly selected fifty-example subset of the FinQA development set. The evaluation pipeline extracts the predicted number from the model's output and compares it to the gold answer using this tolerance.

The comparison of zero-shot performance, subset LoRA fine-tuning, and full-data LoRA fine-tuning enables us to isolate the contributions of direct parameter updates to numerical reasoning accuracy.

4 Code and Data Availability

All code, notebooks, and the FinQA dataset used in this project are publicly available in our GitHub repository:

`https://github.com/YikeWangSerena/FinQA-LoRA-Qwen2.5`

The repository includes the full training scripts (FinQA_v0 and FinQA_v1), the processed FinQA datasets, and the evaluation pipeline used to obtain all results reported in this paper.

5 Experiments and Results

5.1 Experimental Setup

Datasets and Validation: To rigorously evaluate the efficacy of our end-to-end approach, we utilize the official FinQA dataset. We first establish a Zero-shot Baseline using the pre-trained Qwen2.5-7B model without parameters updates. For adaptation, we trained a "Low-Resource" model (FinQA_v0) on a random subset of 2,000 examples for one epoch, and a "Full-Resource" model (FinQA_v1) on the complete training set ($\approx$ 6,000 examples) for two epochs. Evaluation was conducted on a randomized 50-sample slice on the validation set, reporting Execution Accuracy with a strict tolerance of 0.01.

Evaluation Metric: We employ Execution Accuracy as our primary metric. A prediction is classified as correct if and only if the extracted numerical value matches the gold reference within a strict absolute tolerance of 0.01. This binary metric penalizes even minor precision errors, reflecting the rigorous standards of financial reporting.

5.2 Quantitative Analysis

5.2.1 Training Dynamics and Convergence

We monitored the adaptation process of the fully trained model, FinQA_v1. Figure 1 illustrates the training loss trajectory over 1,564 training steps. The convergence pattern reveals two distinct phases of learning:

1. **Phase 1: Format Alignment (Steps 0–150).** The loss drops precipitously from 1.73 to approximately 0.78 in the early stages. We interpret this as the model mastering the structural rules of the task before tackling the logic. Specifically, FinQA_v1 quickly learned to follow the strict output template: generating a reasoning chain followed by the "" token. Learning this surface-level pattern is significantly easier for a language model than acquiring the complex mathematical reasoning skills required for the calculation itself.

2. **Phase 2: Reasoning Acquisition (Steps 150–1564).** Following the initial drop, the curve enters a plateau phase where the loss stabilizes around 0.65. This suggests that the

model has mastered the format and is now attempting to optimize the underlying arithmetic logic. The diminishing marginal reduction in loss during this phase highlights the inherent difficulty of teaching precise calculation logic to a probabilistic generative model without external symbolic tools.

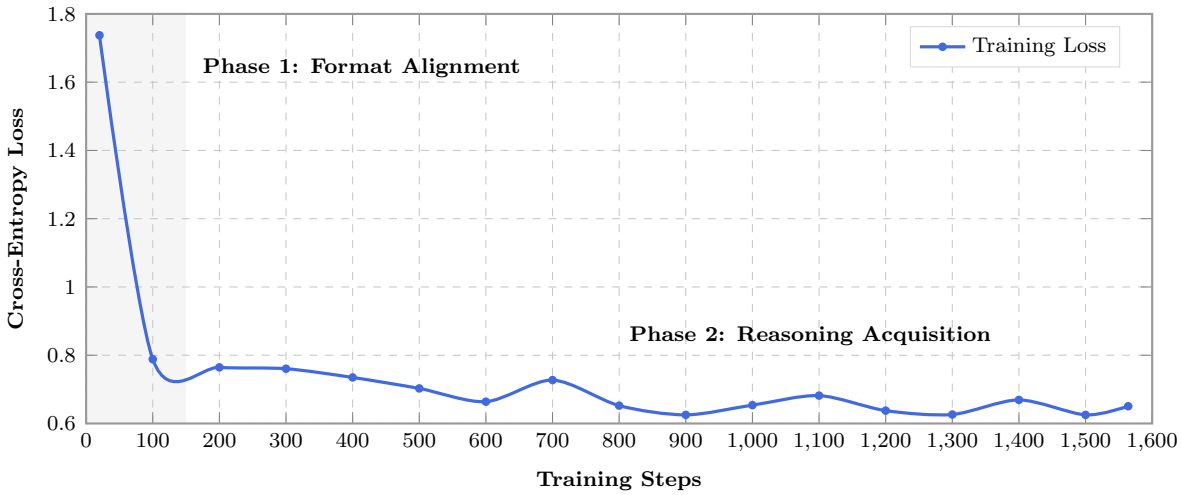

Figure 1: Training Loss for FinQA_v1.

5.2.2 Accuracy Comparison

Table 1 presents the final execution accuracy across different models. The Zero-shot baseline yields a remarkably low accuracy of 15.00%. Qualitative inspection reveals even with a good instruction prompt and explicit answer formatting, the model often fails to carry out the required arithmetic. This confirms that general-purpose pre-training is inadequate for capturing specialized financial vocabulary and performing the numerical reasoning tasks central to FinQA.

In contrast, the application of LoRA fine-tuning yields substantial performance gains. Training on the low-resource subset (FinQA_v0) nearly triples the accuracy to 44.00%, demonstrating high data efficiency. Scaling to the full dataset (FinQA_v1) further pushes performance to 50.00%. Notably, our end-to-end generative approach outperforms the earlier symbolic baseline (Retriever + NeRd, 48.50%) and significantly narrows the gap with FinQANet, the best-performing model reported in the FinQA literature (61.24%), without using any external execution engine[1].

Table 1: **Execution Accuracy Results.**
Comparison between our LoRA-adapted models and established literature baselines.

Model	Epoches	Data Scale	Accuracy
Models from Literature			
Retriever + NeRd	-	Full	48.50%
FinQANet	-	Full	**61.24%**
Our Approach			
Qwen-7B Zero-shot	-	0	15.00%
Qwen-7B LoRA (v0)	1	2k	44.00%
Qwen-7B LoRA (v1)	2	**Full**	**50.00%**

5.3 Qualitative Error Analysis

To better understand the strengths and weaknesses of the LoRA-tuned model beyond aggregate execution accuracy, we conduct a qualitative analysis on representative examples from the 50-example development evaluation set. We compare correct and incorrect cases from both the subset-tuned model (FinQA v0) and the full-data model (FinQA v1).

Success Cases. The model performs reliably on problems that require a single arithmetic operation with well-aligned numerical evidence. A typical success pattern involves direct subtraction, ratio computation, or simple aggregation from closely located table entries.

For example, in one case, the model correctly computed the average transaction volume for American Express:

$$\text{Payments volume} = 637, \quad \text{Transactions} = 5.0 \Rightarrow \text{Answer} = 127.4.$$

Both FinQA v0 and FinQA v1 produced the exact gold answer of 127.4, indicating that the model successfully identified the relevant numbers and applied the correct division.

Another representative success case involves stock return normalization. Given an initial investment of \$100 and an ending value of \$193.5, the model correctly produced the normalized return 0.935. This shows that the LoRA adaptation enables the model to internalize common financial normalization patterns.

The model also performs well on absolute difference questions. For instance, when calculating the increase in operating income from \$10,815 to \$11,503, the model consistently outputs the correct difference of \$688. These results indicate that LoRA fine-tuning significantly strengthens local numerical manipulation when the question aligns tightly with the provided evidence.

Failure Modes. Despite these improvements, failure cases remain prevalent, particularly for questions requiring multi-step reasoning or implicit unit normalization. One frequent error type is incorrect handling of ratios and percentages. For example, when asked to compute the proportion of Canadian production relative to total output, the gold answer is 24.69, while both FinQA v0 and

FinQA v1 predict approximately 0.25. This suggests that the model often confuses raw percentages with normalized decimal ratios.

Another major failure mode involves multi-stage arithmetic. In one example requiring computation of goodwill changes across multiple accounting adjustments, the correct answer is -0.206, but the model predicts values such as -0.2175 or -0.575. While numerically close in some cases, these deviations exceed the strict execution tolerance and are counted as incorrect.

Large-scale financial magnitudes also pose difficulty. For instance, when estimating market capitalization using share price and outstanding shares, the gold answer exceeds 2.25×10^{10}, while the fine-tuned model underestimates the result by nearly an order of magnitude. This suggests that the model struggles with long-number multiplication and scale consistency.

Sign errors represent another common failure pattern. In one tax settlement example, the gold answer is -11.33, but the model predicts $+11.0$, showing that polarity errors remain unresolved even after fine-tuning.

Comparison Between FinQA v0 and FinQA v1. Comparing the subset-trained and full-data models reveals that FinQA v1 improves stability on several borderline cases. For example, stock return normalization and contractual obligation ratio questions that were incorrect under FinQA v0 become correct under FinQA v1. However, deeply compositional problems involving chained operations remain difficult for both models, indicating that increased data alone does not fully resolve multi-step reasoning failure.

Summary of Observations. Overall, the qualitative results suggest that LoRA tuning enables the LLM to reliably solve: Single-step arithmetic problems, direct table lookups with light computation, and simple financial ratios.

However, the model continues to struggle with: Multi-step numerical reasoning, percentage base confusion, sign consistency, and large-scale magnitude control.

These findings align with the quantitative results: fine-tuning significantly improves performance over the zero-shot baseline, but the absence of an explicit symbolic execution mechanism continues to limit performance on complex FinQA queries.

5.4 Insights and Discussion

The experiments and qualitative error analysis above highlight what is easy and what is hard for an LLM + LoRA approach on FinQA. LoRA fine-tuning clearly improves the model's ability to follow the prompt format and to perform local arithmetic, but it does not fully replace the explicit program execution used in FinQANet.

5.4.1 Advantages of the LLM + LoRA approach

Compared to a tradiReporttional neuro-symbolic pipeline, our method has several practical advantages:

Simple architecture. The entire system is implemented through prompts and generated text. We do not build a separate module for program prediction or execution, which avoids additional engineering and integration work.

Reuse of a general backbone. The same Qwen2.57B model can, in principle, be adapted to other financial NLP tasks by switching prompts or lightweight adapters, rather than designing task-specific components. This is attractive in settings where many related tasks share the same document distribution.

Parameter-efficient adaptation. LoRA updates only a small number of low-rank matrices on top of the frozen backbone, which makes it feasible to train on a single GPU. In our experiments, this was enough to raise execution accuracy from 0.15 (zero shot) to 0.44–0.50 on a held-out slice, without any change to the base model weights.

5.4.2 Remaining limitations

At the same time, the qualitative errors in Section 5.3 show clear limitations of our approach:

Multi-step reasoning remains difficult. When the question requires several arithmetic operations in sequence, or combines ratios, percentages, and comparisons across years, the model often fails. It may choose the wrong denominator, mix percentages with decimals, or apply a plausible but incorrect transformation.

Sensitivity to evidence and scale. The model depends heavily on the retrieved snippets. If the retriever surfaces incomplete or slightly misaligned evidence, the answer model can execute the "right" operation on the wrong numbers. Large magnitudes (for example, share price times hundreds of millions of shares) are also error-prone, and the model sometimes underestimates by an order of magnitude.

Lack of explicit reasoning structure. Unlike FinQANet, our system does not predict a symbolic program that can be inspected or debugged. All reasoning is implicit in the generated text. This makes it harder to enforce numerical invariants or to guarantee that all required steps have been performed.

Finally, our evaluation is limited in scale. For time reasons we report results on a 50-example development slice rather than the full development and test sets, so the numeric values should be interpreted as indicative rather than definitive. Nonetheless, the pattern is clear: LoRA fine-tuning substantially improves performance over the zero-shot baseline on simpler queries, but an explicit execution layer still appears necessary to handle the most complex FinQA-style numerical reasoning.

6 Conclusion

This study examined whether a simpler system based on a large language model can replace the more complex neuro symbolic pipelines that are often used for numerical reasoning in finance. Our experiments on the FinQA dataset show that standard models in a zero shot setting do not meet

the accuracy requirements of financial analysis. With LoRA fine tuning, however, performance improves substantially. The adapted model reaches 50 percent execution accuracy on a small development set, performs better than some earlier baselines, and provides a workable alternative to FinQANet without relying on explicit program supervision.

The results point to a clear tradeoff between simplicity and precision. Removing external program executors leads to a more flexible system that is easy to deploy and can run on ordinary hardware. At the same time, the dependence on token level generation introduces mistakes in multi-step calculations, which limits performance on difficult questions.

Several next steps follow from these findings. One direction is to strengthen the reasoning process by generating intermediate explanations or pseudo programs before producing the final answer. This may help improve both accuracy and interpretability. Another direction is to improve evaluation on the full development and test sets , which is necessary to understand how well this approach scales and how it compares to established neuro symbolic methods.

References

[1] Zhiyu Chen, Wenhu Chen, Charese Smiley, Sameena Shah, Iana Borova, Dylan Langdon, Reous Moussa, Matt Beane, and William Yang Wang. **FinQA: A Dataset of Numerical Reasoning over Financial Data**. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 3697–3711, 2021.

[2] Alibaba Cloud. **Qwen2.5-7B-Instruct**. Hugging Face model, 2024. Available at: `https://huggingface.co/Qwen/Qwen2.5-7B-Instruct`.