One More Question is Enough, Expert Question Decomposition (EQD) Model for Domain Quantitative Reasoning

Mengyu Wang[1] **Sotirios Sabanis**[1,2,3] **Miguel de Carvalho**[1,4]
Shay B. Cohen[1] **Tiejun Ma**[1]
[1]The University of Edinburgh, United Kingdom
[2]National Technical University of Athens, Greece
[3]Archimedes/Athena Research Centre, Greece
[4]University of Aveiro, Portugal
{mengyu.wang, s.sabanis, miguel.decarvalho, scohen, tiejun.ma}@ed.ac.uk

Abstract

Domain-specific quantitative reasoning remains a major challenge for large language models (LLMs), especially in fields requiring expert knowledge and complex question answering (QA). In this work, we propose Expert Question Decomposition (EQD), an approach designed to balance the use of domain knowledge with computational efficiency. EQD is built on a two-step fine-tuning framework and guided by a reward function that measures the effectiveness of generated sub-questions in improving QA outcomes. It requires only a few thousand training examples and a single A100 GPU for fine-tuning, with inference time comparable to zero-shot prompting. Beyond its efficiency, EQD outperforms state-of-the-art domain-tuned models and advanced prompting strategies. We evaluate EQD in the financial domain, characterized by specialized knowledge and complex quantitative reasoning, across four benchmark datasets. Our method consistently improves QA performance by 0.6% to 10.5% across different LLMs. Our analysis reveals an important insight: in domain-specific QA, a single supporting question often provides greater benefit than detailed guidance steps.

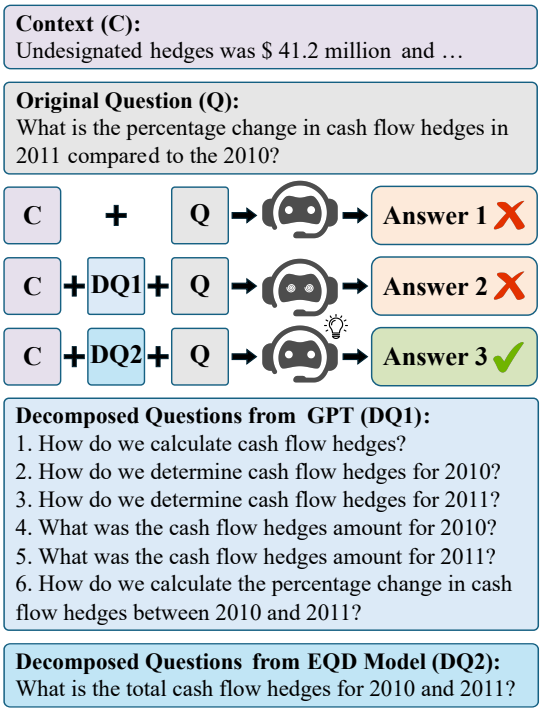

Figure 1: A practical example comparing different QA processes. General LLMs struggle to give correct answers directly. The CoT method attempts to simplify the question, but often decompose the query into overly detailed steps, introducing confusion. In contrast, our EQD model adds a single sub-question that effectively guides the LLM toward the correct answer.

1 Introduction

The performance of LLMs may significantly degrade in specialized domains (Shen et al.). Even advanced LLMs, such as GPT-4o (Hurst et al., 2024) and Llama3 (Dubey et al., 2024), exhibit substantial gaps compared to human experts in domain-specific question answering (QA), particularly in tasks involving quantitative reasoning, like financial analysis (Chen et al., 2021). This performance gap stems from the complex terminology and specialized knowledge inherent in these domains, which are often underrepresented in the pretraining corpora used for general-purpose LLMs.

Recent research addresses domain quantitative reasoning challenges through two main approaches: domain-adapted fine-tuning (Wang et al., 2023; Yang et al., 2023) and prompting techniques (Wei et al., 2022; Chen et al.). However, both approaches face significant limitations. Domain fine-tuning is resource-intensive, requiring large, high-quality domain datasets and significant computational power (Wu et al., 2023). Moreover, many state-of-the-art models like GPT are closed-source, making it difficult to tailor the model to the domain. Prompt-based methods, while model-agnostic and training-free, often reduce inference efficiency due to long augmented inputs. Moreover, they are constrained by the limited extra knowledge contained only in the prompt itself (Srivastava et al., 2024).

Findings of the Association for Computational Linguistics: EMNLP 2025, pages 20355–20369
November 4-9, 2025

Two often-overlooked aspects are potential to mitigate these limitations. First, complex domain knowledge can often be decomposed into simpler, more general components. For instance, a financial question like "*what is the ROI of the investment*" can be transformed into basic arithmetic questions about "*the initial investment, returns, and percentage change*". This suggests that much of the domain-specific knowledge encoded in fine-tuned models may be redundant. Second, given LLMs' inherent strong reasoning abilities, detailed, step-by-step guidance may be unnecessary, or even detrimental. Overly verbose reasoning chains can introduce noise or distract from the core problem, suggesting that targeted questions focusing on key challenges tend to be more effective.

Based on these insights, we develop an Expert Question Decomposition (EQD) model that generates concise and effective supporting questions to guide LLMs in domain-specific reasoning tasks. We select the financial domain, which is characterized by specialized knowledge and quantitative requirements, as the testbed for domain quantitative reasoning. As illustrated in Figure 1, a challenging financial question that remains unsolvable using standard decompositions becomes solvable with a single, supporting question from our EQD model. We consistently observe that such questions significantly improve QA performance.

EQD is developed through a two-step process: *domain fine-tuning* and *QA expert alignment*. In the first step, we fine-tune Llama 3.1-8B-Instruct model using step-by-step question data from financial dialogues. Unlike prior approaches that aim to inject broad domain knowledge into LLMs, our method focuses specifically on fine-tuning the model to decompose domain questions into simpler sub-questions. In the second step, we use a reward-based alignment process. We design a novel reward function that measures the impact of supporting questions by comparing QA performance with and without decomposition. This reward guides the model through reinforcement learning to optimize the quality of generated sub-questions.

Our approach balances computational efficiency and domain knowledge integration. It requires only a small decomposition dataset and a representative domain QA dataset, substantially less than domain-specific LLM fine-tuning. EQD is also compatible with both open- and closed-source LLMs, unlike domain-specific models such as FinMA (Xie et al., 2023) and InvestLM (Yang et al., 2023), which are tied to outdated base models. During inference, EQD typically add only a single supporting question, incurring minimal additional processing overhead and preserving response time comparable to zero-shot prompting.

Beyond its efficiency, EQD demonstrates strong performance in improving domain QA. Across four financial QA benchmarks, it achieves performance gains of 0.6% to 10.5% across multiple LLMs, outperforming advanced domain-adapted models and prompting techniques. These results challenge the conventional emphasis on comprehensive and step-by-step CoT prompting, revealing that concise and supporting questions can lead to better LLM reasoning in specialized domains.

In summary, our key contributions include:

1 We propose a two-stage training framework for expert question decomposition, which integrates domain knowledge efficiently while maintaining inference time comparable to zero-shot prompting. It requires only a few thousand examples and one GPU for training.
We have made our code publicly available at: EQD GitHub repository.

2 We introduce a novel answer comparison reward that guides EQD to generate concise and effective supporting questions. Experiments on four financial QA datasets show that our method consistently improves the performance of various LLMs by 0.6% to 10.5%, and achieve at least a 5% improvement over existing QA approaches.

3 Our results and analysis reveal that concise and supporting questions are more effective than extensive reasoning steps, providing new insights into LLM reasoning mechanisms.

2 Related Work

LLMs have shown progress in quantitative reasoning (Achiam et al., 2023; Zelikman et al., 2022), but continue to face challenges in incorporating specialized domain knowledge (Shen et al.). The financial domain, with its combination of technical terminology and numerical reasoning, has emerged as a key testbed for evaluating domain-specific reasoning capabilities. NLP techniques play an important role in advancing financial applications (Wang and Ma, 2024; Wang et al., 2024), and several financial benchmark datasets have been introduced to evaluate different aspects of domain quantitative reasoning. FinQA (Chen et al., 2021) and ConvFinQA (Chen et al., 2022) focus on multi-

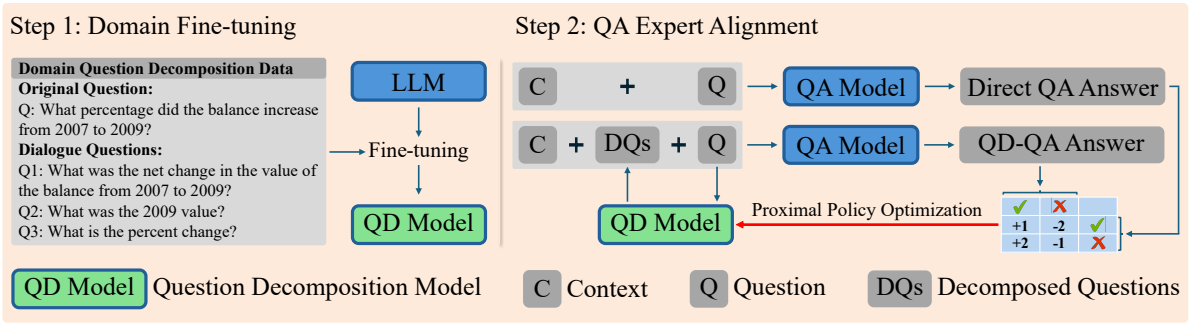

Figure 2: Two-step training framework of the Expert Question Decomposition model.

step mathematical reasoning, while TATQA (Zhu et al., 2021) addresses the challenge of processing diverse input formats, including structured tables and unstructured text in question answering. LLMs have prompted the development of various methods to enhance domain reasoning, including specialized QA models (Zhao et al., 2022; Herzig et al., 2020), prompting techniques (Singh et al., 2024; Leang et al., 2024), and reasoning planning frameworks (Srivastava et al., 2024).

However, most existing methods primarily depend on manually designed prompts to optimize LLMs' domain performance (Li et al., 2024; Cao et al., 2023; Huang et al., 2023; Li et al., 2025), without fully considering the differences between human reasoning and model behavior. Some recent studies have explored using LLMs' own errors as optimization signals (Wu et al., 2025), but their reward functions remain relatively simple. In this work, we propose a four-level reward function that guides LLMs to improve the generation of supporting questions by leveraging QA model outputs. This design aligns with QA models' inherent challenges, thereby enhancing LLM performance.

Despite these advances, a persistent challenge across these approaches lies in balancing domain knowledge integration with computational efficiency. Our EQD method addresses this trade-off by offering a lightweight yet effective question decomposition model. Rather than relying on extensive domain fine-tuning or verbose prompting, EQD generates concise, supporting questions that enhance LLM reasoning with minimal computational overhead.

3 Method

We propose a two-step training approach for developing the Expert Question Decomposition (EQD) model, as illustrated in Figure 2. The first step involves instruction fine-tuning to integrate domain knowledge into the model. Unlike existing methods that require extensive domain corpora for LLM adaptation, we propose to use a small financial conversation dataset to develop a domain-specific Question Decomposition (QD) model. The second step uses reinforcement learning with Proximal Policy Optimization (PPO; Schulman et al. 2017) to align the QD model with the QA process, optimizing the effectiveness of generated Decomposed Questions (DQs) in enhancing QA performance. We introduce a novel answer comparison reward function for this end.

3.1 Domain Fine-tuning

Domain knowledge plays an important role in improving LLM performance on specialized tasks. Traditional methods incorporate domain knowledge by fine-tuning on extensive domain-specific corpora or linking external databases. However, the gap between general and domain knowledge is often narrower than expected. In many cases, a simple term explanation can transform domain-specific questions into general queries. For instance, a specialized financial question like "*What is the ROI of the investment*" can be converted to a general arithmetic question involving "*initial investment, returns, and percentage change*".

Motivated by this insight, we propose to fine-tune a lightweight LLM (Llama3.1-8B-Instruct in our experiments) exclusively for domain-specific question decomposition. This approach balances domain adaptation and training efficiency.

We use ConvFinQA (Chen et al., 2022), an expert-annotated financial conversation dataset containing around only 3,000 entries, to fine-tune our QD model. As shown on the left side of Figure 2, we extract only the original questions and the dialogue questions as input information, discarding answers and explanations. We design a system prompt focused solely on question decomposition (detailed in Appendix C), and fine-tune the base LLM using next-token prediction on this input.

The resulting QD model embeds domain-specific knowledge essential for understanding and decomposing financial questions. This mitigates the limitations of prompt-based methods, which are constrained by context length and limited domain-specific input. The QD model serves as an automatic reasoning chain generator, generating effective supporting questions to assist QA process.

Since the model is trained only for question decomposition, its training cost is significantly lower than full-domain LLM fine-tuning. Additionally, although developed based on an open-source LLM, this QD model can transfer its domain knowledge to any other LLMs, regardless of their open- or closed-source nature. By inserting its generated decomposed questions into QA inputs, we observe consistent improvements in reasoning accuracy across diverse LLMs.

3.2 QA Expert Alignment

While domain fine-tuning equips the QD model with financial question decomposition expertise, it does not guarantee optimal support for the QA process. An ideal QD model should generate supporting questions that increase the QA model's likelihood of answering correctly while minimizing the risk of introducing misleading information. This requirement parallels the broader LLM alignment problem, where the goal is to align a model's behavior with human intent. Similarly, we aim to align the QD model's outputs with the latent preferences of the QA model, rather than solely mimicking human-annotated decomposition patterns.

To achieve this QA alignment, we introduce an answer comparison reward to quantify the impact of the QD model's outputs on QA performance. Specifically, we compare two outputs from the QA model: one answer obtained through direct QA (i.e., the LLM answers the question without assistance), and another answer from QD-assisted QA (i.e., the same LLM answers with supporting questions generated by the QD model). This controlled setup, shown on the right side of Figure 2, ensures that the only difference lies in the presence of decomposed questions (DQs), isolating their influence.

Let a_{di} denote the direct QA answer and a_{qd} represent the QD-assisted answer. The reward score r is calculated as:

$$c(a) = \begin{cases} 1, & \text{if answer } a \text{ is correct} \\ -1, & \text{otherwise} \end{cases}, \quad (1)$$

$$r = c(a_{qd}) \cdot (1 + 0.5 \cdot |c(a_{di}) - c(a_{qd})|) \quad (2)$$

where $c(\cdot)$ is the function to evaluate the correctness of the given answer.

This reward yields four possible values: +2, +1, -2, and -1, corresponding to high positive, low positive, high negative, and low negative rewards, respectively. DQs receive a positive score when they lead to correct answers, and a negative score otherwise. The magnitude of the score (high or low) is determined by whether the DQs alter the correctness of the answer compared to direct QA. Specifically, the four values represent: +2: DQs correct an originally incorrect answer, +1: DQs preserve a correct answer, -2: DQs turn a correct answer into an incorrect one, -1: DQs preserve an incorrect answer. This scoring mechanism captures both the correctness and influence of the DQs, encouraging outputs that improve QA performance and penalizing those that degrade it.

Using this reward, we fine-tune the QD model via PPO algorithm. PPO adjusts model parameters to maximize the expected reward while maintaining a bounded KL divergence from a reference model, ensuring updates remain within a trust region. The reference model is initialized as a copy of the QD model from step 1, allowing us to preserve its financial knowledge and decomposition style.

In summary, this reinforcement learning stage aligns the QD model with the QA model's requirements, evolving it into the EQD model to generate decomposed questions that effectively support the QA model in producing correct answers.

3.3 Resource Requirement

We detail the resource requirements of our method to highlight its efficiency and practicality.

Training Data. Our approach requires only a question decomposition dataset and a representative QA dataset for the focused domain. After completing the two-step training, the resulting EQD model can generalize to various QA datasets within the same domain. In our experiments, we use just two datasets, ConvFinQA and FinQA, among the many datasets used for financial LLM fine-tuning (Xie et al., 2024b), yet demonstrate strong performance across four different financial QA benchmarks.

Computational Resources. Our training requires only a single GPU capable of fine-tuning the base model (an A100 in our setup). Although the two-step training involves multiple roles-the QA model, QD model, and reference model-we efficiently organize model parameters to keep the resource demands equivalent to running a single LLM.

We use Low-Rank Adaptation (LoRA; Hu et al. 2022) for parameter-efficient fine-tuning in both steps. The added adapter contains only 22 million parameters, about 0.27% of the 8B base model. We use a continuous fine-tuning strategy, training the same adapter throughout both steps. In step 1, the model comprises the base LLM with a trainable adapter. In step 2, although three logical models are involved, only one full LLM and two adapters are required in practice. Specifically, the base LLM serves as the QA model, the trainable adapter forms the QD model, and a frozen copy of the adapter serves as the reference model. These roles are managed by activating or deactivating the corresponding adapters, minimizing memory overhead.

In summary, our method integrates domain knowledge in a more resource-efficient manner than existing domain model fine-tuning methods.

3.4 Expert Question Decomposition Model

After the two-step fine-tuning, we develop an Expert Question Decomposition (EQD) model combining domain expertise with optimized QA alignment. This model offers two key advantages over existing prompting-based methods.

First, the EQD model generates prompts automatically, in contrast to conventional methods that rely on rule-based or manually crafted prompts. This automation enables broad applicability across diverse datasets within the same domain, eliminating the need for dataset-specific prompt design.

Second, while existing prompting methods often construct detailed and comprehensive guidance, our EQD model produces concise yet effective supporting questions. As illustrated in Figure 1, a single well-chosen supporting question outperforms multiple detailed guidance steps. This observation suggests that LLMs already possess strong reasoning capabilities, and excessive guidance can be redundant, or even harmful. Our training objectives do not explicitly penalize or limit generation length. Instead, the reinforcement learning reward function solely optimizes the effectiveness of the generated decomposed questions in improving QA outcomes. The natural emergence of conciseness in the model's outputs indicates that brevity is an inherent requirement of effective QA support.

In conclusion, our EQD model exhibits two key advantages over existing prompting methods: domain-specific versatility and concise yet effective question decomposition.

4 Experiment Settings

4.1 Training and Testing Datasets

Our EQD model is trained using two financial datasets. For step 1, we use the training split of ConvFinQA (Chen et al., 2022), comprising 3,073 entries of financial reasoning conversations. For step 2, we use the training split of FinQA (Chen et al., 2021), containing 6,250 financial QA pairs.

To evaluate the generalized QA improvement capability of our EQD model, we conduct testing on four distinct financial datasets: FinQA, TAT-QA (Zhu et al., 2021), ECTQA (Mukherjee et al., 2022), and EDTQA (Xie et al., 2024a). All four testing datasets require both domain-specific knowledge and numerical reasoning capabilities. A detailed description of each dataset, along with relevant statistics, is provided in Appendix D.

4.2 Implementation Details

We use the Llama3.1-8B-Instruct model (Dubey et al., 2024) as both the base model and QA model. For reinforcement learning, we use the PPO model's value head as the critic model. For performance evaluation, we use the exact match accuracy (EmAcc) metric, following established practices in previous works (Xie et al., 2023; Zhao et al., 2024). Since the key points of the answers are numerical values, we implement a systematic evaluation process (Singh et al., 2024) that extracts values and compares them to the ground truth. Detailed information on parameter settings, API costs, evaluation setup, and computing devices are presented in Appendix A. The strategy for managing trainable parameters across the two fine-tuning stages is discussed in Appendix E.

In step 1, we perform next-token prediction by concatenating a question decomposition instruction, the original question, and the conversation sub-questions as inputs. In step 2, we conduct QA by concatenating a financial QA instruction, the financial article, the decomposed questions, and the final questions. The model's final response serves as the answer to the original question. Detailed prompt examples are provided in Appendix C.

In Step 2, we assign specific scores of +2, +1, –1, and –2 to the four reward levels. We also experimented with alternative discrete reward configurations for comparison-based learning, such as (+2, +1, –1, –4) and (+4, +1, –1, –2). The results show that the balanced configuration (+2, +1, –1, –2) achieves the best performance. Other arrangements, such as merging the lower levels or using unbalanced scores, led to performance degradation. Consequently, we use the balanced reward configuration for our final method.

To evaluate the generalization ability of our approach for obtaining an EQD model, we also experiment with three additional LLMs: Llama3.2-1B-Instruct, Llama3.2-3B-Instruct, and DeepSeek-R1-Distill-Qwen-7B (DeepSeek-AI, 2025), spanning different model sizes and architectures. The corresponding results and analysis are presented in Appendix F.

4.3 Baseline Methods

We benchmark our method across various QA models and reasoning support techniques.

Our experiments use a diverse range of LLMs, including Llama3.1-8B-Instruct, GPT-3.5-turbo, GPT-4o, o3-mini, Claude3.5-sonnet[1], and FinMA (Xie et al., 2023). This selection encompasses advanced open-source, closed-source, and domain-specific fine-tuned models. We compare each model's QA performance with and without support from our EQD method to demonstrate the generalized effectiveness of our method.

For reasoning support baselines, we compare EQD with several established prompting strategies, including zero-shot Chain-of-Thought (0-CoT; Wei et al. 2022), decomposed prompting (DP; Khot et al.), question decomposition CoT (QD-CoT; Zhou et al.), retrieval CoT (R-CoT; Trivedi et al. 2023), and few-shot in-context learning (N-shot; Li et al. 2023). These approaches are widely adopted for general QA tasks, and both 0-CoT and few-N-shot methods have proved effective in financial domains (Srivastava et al., 2024). Implementation details for all baseline methods are provided in Appendix B.

5 Results and Discussions

We evaluate our EQD method from four perspectives, presenting comprehensive experimental results to support our findings. In Section 5.1, we compare the performance of various LLMs on four datasets, both in direct QA and with EQD support, demonstrating that our approach consistently enhances LLM performance on domain-specific quantitative reasoning tasks. Section 5.2 focuses on the two most challenging datasets, FinQA and TAT-QA, to compare different reasoning support methods, further validating the effectiveness of EQD. Section 5.3 presents ablation studies, comparing our EQD model with other LLMs for question decomposition, and analyzing the impact of each fine-tuning step. Finally, Section 5.4 evaluates the computational efficiency of our EQD methods through inference time and generation length analysis.

5.1 Generalized QA Improvement on LLMs

Table 1 presents a comparative analysis of various LLMs on different financial QA datasets, evaluated both with and without EQD support.

Our EQD model yields consistent performance improvements across all general LLMs on different datasets, with two exceptions: FinMA and o3-mini. These models operate independently and cannot take advantage of the supporting techniques due to inherent limitations.

For general LLMs, EQD-supported QA consistently outperforms direct QA in average performance across both models and datasets. And the best results on three datasets, FinQA, TAT-QA, and ECTQA, are achieved by LLMs supported by our EQD model. Importantly, despite being trained solely on the FinQA training set and fine-tuned only using Llama3.1-8B-Instruct as the QA model, the EQD model exhibits strong generalization ability, improving QA across a range of datasets and models. This underscores the robustness of the expert question decomposition strategy.

The performance of the two standalone models, FinMA and o3-mini, reflects the limitations of systems that function in isolation rather than flaws of our approach. FinMA, an open-source LLM fine-tuned specifically for financial tasks, relies on the first-generation Llama model and is constrained by a limited input window. These factors hinder its ability to benefit from any additional contextual information. The o3-mini model, OpenAI's latest reasoning model, uses a simulated reasoning mechanism. According to OpenAI's documentation, this special model performs optimally with straightforward prompts, and prompt engineering techniques may actually impede its perfor-

[1] https://www.anthropic.com/news/claude-3-5-sonnet

Model	FinQA		TAT-QA		ECTQA		EDTQA		*Average*	
	Direct	EQD	Direct	EQD	Direct	EQD	Direct	EQD	Direct	EQD
Llama3.1-8B	47.2	54.0	51.2	54.9	61.8	64.0	52.2	55.1	53.1	57.0
GPT-3.5-turbo	28.4	55.1	47.2	52.7	64.7	65.4	56.0	57.3	47.1	57.6
GPT-4o	58.2	62.4	59.1	63.2	68.1	72.5	**64.9**	63.4	62.5	65.4
Claude3.5-sonnet	72.9	**73.7**	63.3	**64.4**	74.8	**75.2**	60.8	61.2	67.9	68.5
Average	51.7	61.2	55.2	58.8	67.4	69.3	58.5	59.3	58.2	62.1
FinMA	11.3	10.5	19.1	18.2	1.9	1.8	37.4	35.1	17.4	16.4
o3-mini	70.0	67.6	62.5	57.3	74.4	70.2	64.7	41.3	67.9	59.1

Table 1: Comparison of LLM performance on financial QA tasks: Direct QA vs. QA supported by our EQD model. Underlined values indicate the higher score between Direct QA and EQD-supported QA. Bold values denote the best performance for each dataset.

mance[2]. These cases, representing domain-specific and reasoning-optimized models, highlight the limitations of methods that can only work independently. In contrast, our EQD method demonstrates flexibility and compatibility with a wide range of advanced LLMs to achieve optimal results.

Additionally, two trends are observed in domain quantitative reasoning. (1) Greater EQD improvements for weaker QA models. The benefit of EQD is more pronounced in weaker LLMs. When averaged across datasets, Claude 3.5 Sonnet, the strongest model, shows the smallest performance gain (+0.6%), whereas GPT-3.5-turbo, the weakest, shows the largest (+10.5%). This suggests that weaker LLMs struggle more with complex reasoning and thus benefit more from EQD's decomposition. (2) Greater EQD improvements for more complex reasoning tasks. Averaged across LLMs, the smallest improvement is observed on EDTQA (+0.8%), while the largest is on FinQA (+9.5%). Since ECTQA and EDTQA are derived from summarization datasets, they contain questions that typically require simpler reasoning or direct value extraction. In contrast, FinQA questions demand multi-step calculations (typically 3–4 operations), which LLMs often fail to handle reliably without support. This indicates that EQD provides more value in tasks involving quantitative reasoning.

These findings reinforce our claim: EQD's question decomposition enhances LLMs' QA performance by degrading reasoning challenges. In contrast, when LLMs can already manage complex tasks, additional guidance yields diminishing returns. This also explains why concise supporting questions generated by EQD often outperform more detailed or verbose prompt instructions.

[2] https://platform.openai.com/docs/guides/reasoning-best-practices

5.2 Comparison with Different Methods

Table 2 presents the comparative results of various LLM-based methods for financial QA. Due to resource constraints, we evaluated three representative QA models: Llama3.1-8B-Instruct, GPT-3.5-turbo, and GPT-4o, using two of the most challenging datasets, FinQA and TAT-QA.

Our EQD-QA method demonstrates robust performance, outperforming all other methods by an average margin of at least 5%. It achieves the best results in four out of six specific scenarios and ranks second in the remaining case.

The first three baseline methods, 0-CoT, DP1, and DP2, aim to elicit reasoning capabilities from LLMs without incorporating external knowledge. While effective in general QA settings, they struggle with financial tasks due to insufficient domain-specific understanding. The next set of baselines, QD-CoT, R-CoT and N-shot, introduce domain knowledge through either example-based decompositions or domain-specific reasoning chains. Although these approaches improve performance in some scenarios, they fail to deliver consistent benefits across models and datasets. This inconsistency underscores a key insight: simply embedding domain knowledge into prompts is insufficient for reliable performance gains. In contrast, our EQD model is explicitly trained to optimize the effectiveness of the additional information in supporting QA process. This targeted optimization explains its superior and consistent performance over other prompt-based strategies.

5.3 Ablation Study

We conduct two sets of ablation studies to assess the components and training strategy of EQD.

First, we compare EQD with alternative methods

QA LLM	Dataset	Methods								
		Direct	0-CoT	DP1	DP2	QD-CoT	R-CoT	N-shot	EQD	Manual
Llama 3.1-8B	FinQA	47.2	52.0	47.6	50.0	50.0	48.0	45.1	**54.0**	51.5
	TAT-QA	51.2	**57.4**	49.7	51.4	51.8	49.9	53.9	54.9	51.6
GPT-3.5 -turbo	FinQA	28.4	16.8	31.2	51.4	49.6	50.9	39.1	**55.1**	52.6
	TAT-QA	47.2	46.8	46.5	52.6	46.4	**52.9**	50.0	52.7	52.1
GPT-4o	FinQA	58.2	53.1	55.8	49.8	60.3	49.7	42.6	**62.4**	52.5
	TAT-QA	59.1	58.6	54.4	50.9	54.2	51.1	61.0	**63.2**	51.8
	Average	48.6	47.5	47.5	51.0	52.1	50.4	48.6	**57.1**	52.0

Table 2: Comparison of different methods for conducting QA tasks across different LLMs. The names of the baseline methods are abbreviated as described in Section 4.3. Since DP is a basic question decomposition baseline, we test two versions using GPT-3.5-turbo and GPT-4o as the question decomposition models, denoted as DP1 and DP2, respectively. The final column, "Manual", refers to the method in which we manually design question decomposition examples based on our findings for prompts, serving as an ablation study. Bold and underline values represent the best and second-best results for each row.

for question decomposition. As shown in Table 2, the columns "DP1", "DP2", and "Manual" represent QA results using supporting questions from different methods. DP1 and DP2 serve as both baselines and ablations, since they use general-purpose LLMs to generate decompositions. Results show that our EQD model surpasses even the recent GPT models in generating effective sub-questions.

The "Manual" approach assumes foreknowledge of EQD's key conclusion that concise and supporting questions are more effective. We manually write five concise examples and apply them in a 5-shot prompting setup to guide LLMs' decomposition. This method outperforms many baselines, validating our finding. However, it still underperforms our EQD model. This is because our model is specifically trained to generate the most essential sub-questions for the QA process, whereas human annotators may not always be able to identify the most important reasoning steps for LLMs.

Second, using Llama3.1-8B-Instruct as the QA model, we examine the contribution of each training step. We compare four configurations: no fine-tuning, step 1 only, both steps, and step 2 only. Figure 3 presents the QA performance and average word count of the generated sub-questions. The trend in average word count also reflects changes in the number of decomposed questions generated, with average question counts of 15.0, 6.23, 1.2, and 15.6 for the four settings, respectively.

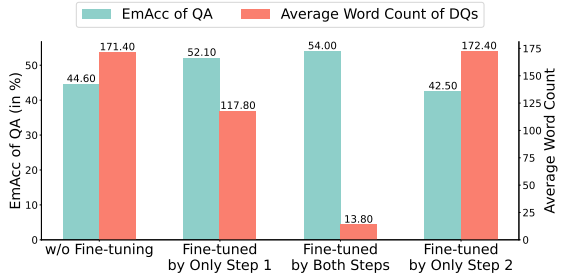

Figure 3: Comparison of QD models fine-tuned differently, using Llama3.1 as QA model. Blue bars reflect QA accuracy (left y-axis), while red bars indicate the average word count of generated questions (right y-axis).

Results indicate that combining both steps gives the best performance and most concise questions. Both steps enhance the effectiveness and conciseness of the generated questions. Step 1, which focuses on incorporating domain knowledge, contributes more to generation effectiveness. Step 2 improves brevity by guiding the model to generate focused questions. Together, they enable the generation of sub-questions that are both informative and efficient for downstream QA tasks. A case study illustrating the model's generation evolution is presented in Appendix G.

Importantly, removing Step 1 results in performance similar to no fine-tuning, emphasizing that domain knowledge is essential for EQD's effectiveness. Step 2 alone cannot optimize decomposition without the foundation built in step 1.

These findings highlight the necessity of our two-step strategy. It enables the EQD model to generate sub-questions that are both informative and efficient for downstream QA tasks.

5.4 Efficiency Analysis

Figure 4 presents a comparison of inference time consumption and additional input length across different methods on the test split of the FinQA dataset. We compare our EQD method with three

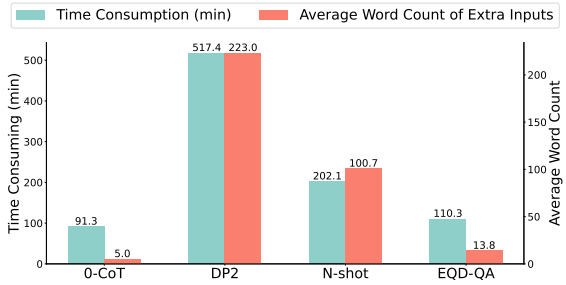

Figure 4: Comparison of inference time consumption and input length across methods. Blue bars represent time consuming (left y-axis), while red bars indicate the average word count of extra inputs (right y-axis).

baseline approaches, 0-CoT, DP2, and N-shot, which are the fastest among similar methods.

Our EQD-QA shows significantly lower inference time and shorter extra inputs compared to N-shot and DP2 (using GPT-4o as the QD model). Its inference time and extra prompt length are only marginally higher than 0-CoT, which simply adds reasoning prompts without domain knowledge.

Further analysis shows GPT-4o generates an average of 7.3 supporting questions per case, while EQD generates just 1.2. As illustrated in Figure 1, GPT-4o tends to produce overly detailed decomposed questions that may hinder reasoning. When considered with the previous performance comparisons, these results indicate that our EQD model generates more concise yet effective sub-questions for domain quantitative reasoning. The findings suggest that a single critical supporting question can be more beneficial for domain quantitative reasoning than multiple detailed reasoning steps.

6 Conclusions

This paper introduces a novel two-step fine-tuning method, including domain fine-tuning and QA expert alignment, to develop an Expert Question Decomposition (EQD) model. Our EQD model demonstrates significant effectiveness and efficiency in supporting domain quantitative reasoning. Experimental results across four financial datasets show consistent improvements in QA performance across various LLMs, maintaining computational efficiency comparable to zero-shot prompting methods. Furthermore, our analysis reveals that a single critical supporting question is more beneficial to the domain QA process than detailed step-by-step guidance, providing novel insights into LLMs' reasoning capabilities in specialized domains.

Limitations

While our EQD model presents a novel approach to enhance LLMs' domain quantitative reasoning capabilities, two primary limitations should be acknowledged, due to resource constraints and data availability:

First, our baseline comparison includes only FinMA as a representative domain fine-tuned LLM, due to limited accessibility of similar models. For instance, FinLLaMA (Xie et al., 2024b), a recent financial domain fine-tuned LLM, requires author authorization for access, but inactive response to access requests has restricted its availability. However, this limitation does not impact our comparative analysis, as the reported performance of major financial LLMs (e.g., BloomBergGPT; Wu et al. 2023, and InvestLM; Yang et al. 2023) on FinQA and TAT-QA datasets falls considerably below our results using Claude3.5-sonnet. As demonstrated through FinMA, these domain fine-tuned models are inherently constrained by their base model capabilities and often lag behind state-of-the-art LLMs. In contrast, our EQD model's flexible integration with advanced LLMs offers advantages without such constraints.

Second, our evaluation of the EQD-QA method is evaluated only on the financial domain, a representative field for quantitative reasoning. The implementation of our method requires domain-specific question decomposition datasets for fine-tuning. At present, ConvFinQA (Chen et al., 2022) is the only publicly available dataset that meets our requirements. Although this limited dataset availability constrains broader evaluation across domains, our method's design and underlying principles are domain-agnostic, not specifically tied to financial knowledge. Moreover, since ConvFinQA contains only 3,037 training samples, datasets of this scale are feasible for companies to annotate based on their practical requirements. Therefore, our method has potential for extension to other domains.

Acknowledgments

This work was supported by the UKRI Centre for Doctoral Training (CDT) in Natural Language Processing through UKRI grant EP/S022481/1. We also acknowledge the support of the Centre for Investing Innovation at the University of Edinburgh.

References

Josh Achiam, Steven Adler, Sandhini Agarwal, Lama Ahmad, Ilge Akkaya, Florencia Leoni Aleman, Diogo Almeida, Janko Altenschmidt, Sam Altman, Shyamal Anadkat, et al. 2023. GPT-4 technical report. *arXiv preprint arXiv:2303.08774*.

Hejing Cao, Zhenwei An, Jiazhan Feng, Kun Xu, Liwei Chen, and Dongyan Zhao. 2023. A step closer to comprehensive answers: Constrained multi-stage question decomposition with large language models. *CoRR*.

Wenhu Chen, Xueguang Ma, Xinyi Wang, and William W Cohen. Program of thoughts prompting: Disentangling computation from reasoning for numerical reasoning tasks. *Transactions on Machine Learning Research*.

Zhiyu Chen, Wenhu Chen, Charese Smiley, Sameena Shah, Iana Borova, Dylan Langdon, Reema Moussa, Matt Beane, Ting-Hao Huang, Bryan R Routledge, et al. 2021. FinQA: A dataset of numerical reasoning over financial data. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 3697–3711.

Zhiyu Chen, Shiyang Li, Charese Smiley, Zhiqiang Ma, Sameena Shah, and William Yang Wang. 2022. Convfinqa: Exploring the chain of numerical reasoning in conversational finance question answering. In *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing*, pages 6279–6292.

DeepSeek-AI. 2025. Deepseek-r1: Incentivizing reasoning capability in llms via reinforcement learning. *Preprint*, arXiv:2501.12948.

Abhimanyu Dubey, Abhinav Jauhri, Abhinav Pandey, Abhishek Kadian, Ahmad Al-Dahle, Aiesha Letman, Akhil Mathur, Alan Schelten, Amy Yang, Angela Fan, et al. 2024. The llama 3 herd of models. *CoRR*.

Jonathan Herzig, Pawel Krzysztof Nowak, Thomas Mueller, Francesco Piccinno, and Julian Eisenschlos. 2020. Tapas: Weakly supervised table parsing via pre-training. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 4320–4333.

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

Xiang Huang, Sitao Cheng, Yiheng Shu, Yuheng Bao, and Yuzhong Qu. 2023. Question decomposition tree for answering complex questions over knowledge bases. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 37, pages 12924–12932.

Aaron Hurst, Adam Lerer, Adam P Goucher, Adam Perelman, Aditya Ramesh, Aidan Clark, AJ Ostrow, Akila Welihinda, Alan Hayes, Alec Radford, et al. 2024. GPT-4o system card. *arXiv preprint arXiv:2410.21276*.

Tushar Khot, Harsh Trivedi, Matthew Finlayson, Yao Fu, Kyle Richardson, Peter Clark, and Ashish Sabharwal. Decomposed prompting: A modular approach for solving complex tasks. In *The Eleventh International Conference on Learning Representations*.

Joshua Ong Jun Leang, Aryo Pradipta Gema, and Shay B Cohen. 2024. CoMAT: Chain of mathematically annotated thought improves mathematical reasoning. *arXiv preprint arXiv:2410.10336*.

Changcheng Li, Xiangyu Wang, Qiuju Chen, Xiren Zhou, and Huanhuan Chen. 2024. Mtmt: Consolidating multiple thinking modes to form a thought tree for strengthening llm. *arXiv preprint arXiv:2412.03987*.

Tianle Li, Xueguang Ma, Alex Zhuang, Yu Gu, Yu Su, and Wenhu Chen. 2023. Few-shot in-context learning on knowledge base question answering. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 6966–6980.

Ying Li, Mengyu Wang, Miguel de Carvalho, Sotirios Sabanis, and Tiejun Ma. 2025. Fingear: Financial mapping-guided enhanced answer retrieval. *Preprint*, arXiv:2509.12042.

Rajdeep Mukherjee, Abhinav Bohra, Akash Banerjee, Soumya Sharma, Manjunath Hegde, Afreen Shaikh, Shivani Shrivastava, Koustuv Dasgupta, Niloy Ganguly, Saptarshi Ghosh, et al. 2022. Ectsum: A new benchmark dataset for bullet point summarization of long earnings call transcripts. In *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing*, pages 10893–10906.

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

Junhong Shen, Neil Tenenholtz, James Brian Hall, David Alvarez-Melis, and Nicolo Fusi. Tag-llm: Repurposing general-purpose llms for specialized domains. In *Forty-first International Conference on Machine Learning*.

Kuldeep Singh, Simerjot Kaur, and Charese Smiley. 2024. Finqapt: Empowering financial decisions with end-to-end llm-driven question answering pipeline. In *Proceedings of the 5th ACM International Conference on AI in Finance*, pages 266–273.

Pragya Srivastava, Manuj Malik, Vivek Gupta, Tanuja Ganu, and Dan Roth. 2024. Evaluating llms' mathematical reasoning in financial document question answering. In *Findings of the Association for Computational Linguistics ACL 2024*, pages 3853–3878.

Harsh Trivedi, Niranjan Balasubramanian, Tushar Khot, and Ashish Sabharwal. 2023. Interleaving retrieval

with chain-of-thought reasoning for knowledge-intensive multi-step questions. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 10014–10037.

Mengyu Wang, Shay B Cohen, and Tiejun Ma. 2024. Modeling news interactions and influence for financial market prediction. In *Findings of the Association for Computational Linguistics: EMNLP 2024*, pages 3302–3314.

Mengyu Wang and Tiejun Ma. 2024. Mana-net: Mitigating aggregated sentiment homogenization with news weighting for enhanced market prediction. In *Proceedings of the 33rd ACM International Conference on Information and Knowledge Management*, pages 2379–2389.

Neng Wang, Hongyang Yang, and Christina Dan Wang. 2023. Fingpt: Instruction tuning benchmark for open-source large language models in financial datasets. *CoRR*.

Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, Fei Xia, Ed Chi, Quoc V Le, Denny Zhou, et al. 2022. Chain-of-thought prompting elicits reasoning in large language models. *Advances in neural information processing systems*, 35:24824–24837.

Ruofan Wu, Youngwon Lee, Fan Shu, Danmei Xu, Seung-won Hwang, Zhewei Yao, Yuxiong He, and Feng Yan. 2025. Composerag: A modular and composable rag for corpus-grounded multi-hop question answering. *arXiv preprint arXiv:2506.00232*.

Shijie Wu, Ozan Irsoy, Steven Lu, Vadim Dabravolski, Mark Dredze, Sebastian Gehrmann, Prabhanjan Kambadur, David Rosenberg, and Gideon Mann. 2023. Bloomberggpt: A large language model for finance. *arXiv preprint arXiv:2303.17564*.

Qianqian Xie, Weiguang Han, Xiao Zhang, Yanzhao Lai, Min Peng, Alejandro Lopez-Lira, and Jimin Huang. 2023. Pixiu: a large language model, instruction data and evaluation benchmark for finance. In *Proceedings of the 37th International Conference on Neural Information Processing Systems*, pages 33469–33484.

Qianqian Xie, Jimin Huang, Dong Li, Zhengyu Chen, Ruoyu Xiang, Mengxi Xiao, Yangyang Yu, Vijayasai Somasundaram, Kailai Yang, Chenhan Yuan, et al. 2024a. Finnlp-agentscen-2024 shared task: Financial challenges in large language models-finllms. In *Proceedings of the Eighth Financial Technology and Natural Language Processing and the 1st Agent AI for Scenario Planning*, pages 119–126.

Qianqian Xie, Dong Li, Mengxi Xiao, Zihao Jiang, Ruoyu Xiang, Xiao Zhang, Zhengyu Chen, Yueru He, Weiguang Han, Yuzhe Yang, et al. 2024b. Open-finllms: Open multimodal large language models for financial applications. *CoRR*.

Yi Yang, Yixuan Tang, and Kar Yan Tam. 2023. Investlm: A large language model for investment using financial domain instruction tuning. *arXiv preprint arXiv:2309.13064*.

Eric Zelikman, Yuhuai Wu, Jesse Mu, and Noah Goodman. 2022. Star: Bootstrapping reasoning with reasoning. *Advances in Neural Information Processing Systems*, 35:15476–15488.

Huaqin Zhao, Zhengliang Liu, Zihao Wu, Yiwei Li, Tianze Yang, Peng Shu, Shaochen Xu, Haixing Dai, Lin Zhao, Gengchen Mai, et al. 2024. Revolutionizing finance with llms: An overview of applications and insights. *CoRR*.

Yilun Zhao, Yunxiang Li, Chenying Li, and Rui Zhang. 2022. Multihiertt: Numerical reasoning over multi hierarchical tabular and textual data. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 6588–6600.

Denny Zhou, Nathanael Schärli, Le Hou, Jason Wei, Nathan Scales, Xuezhi Wang, Dale Schuurmans, Claire Cui, Olivier Bousquet, Quoc V Le, et al. Least-to-most prompting enables complex reasoning in large language models. In *The Eleventh International Conference on Learning Representations*.

Fengbin Zhu, Wenqiang Lei, Youcheng Huang, Chao Wang, Shuo Zhang, Jiancheng Lv, Fuli Feng, and Tat-Seng Chua. 2021. Tat-qa: A question answering benchmark on a hybrid of tabular and textual content in finance. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 3277–3287.

A More implementation details

A.1 Fine-tuning Settings

We added a LoRA adapter with a rank of 8 and a LoRA alpha of 16. The fine-tuning process targeted eight parameter matrices: “q_proj”, “k_proj”, “v_proj”, “o_proj”, “gate_proj”, “up_proj”, “down_proj”, and “lm_head”. The adapter includes 22 million parameters, representing 0.27% of the original model’s parameters.

We used a batch size of 32 for step 1 and 8 for step 2. The learning rate was set to 1e-5, with a warm-up of 5 steps. The maximum number of training iterations was set to 1000 for step 1 and 500 for step 2. Model checkpoints were selected based on performance on the FinQA validation set: iteration 400 for step 1 and iteration 200 for step 2.

During step 2, the average length of generated responses dropped from about 200 tokens at initialization to approximately 20 tokens within the

first 50 iterations. It remained stable at around 20 tokens until iteration 500. The selected checkpoint in step 2 did not complete a full epoch over the FinQA training set. This is reasonable as the goal of step 2 is not to exhaustively learn all QA pairs, but to capture the reasoning challenges faced by the QA model. Although FinQA contains numerous QA pairs, many share similar reasoning structures. Step 2 prioritizes learning he difficulty of the reasoning process rather than memorizing knowledge from specific QA instances.

A.2 Device and Training Time

We used an A100 GPU for both training and testing. Step 1 fine-tuning required approximately 2.5 hours, and the step 2 required around 4 hours. Both the device and time requirements are significantly less than those for fine-tuning a domain-specific LLM.

A.3 API Cost

For all closed-source LLMs, including GPT-3.5-turbo, GPT-4o, o3-mini, and Claude3.5-sonnet, we used API to run experiments. Converting the two summary datasets, ECTSum and EDTSum, to QA datasets cost approximately $2. Using these LLMs as QA models cost around $200, primarily covered by GPT-4o and Claude3.5-sonnet. Using them as QD models cost approximately $10.

A.4 Evaluation Details

Following prior work on FinQA dataset (Chen et al., 2021; Singh et al., 2024), we evaluate QA performance using the exact match accuracy (EmAcc) metric. Since most annotated answers are numerical, this evaluation process includes extracting answer strings using regular expression patterns, converting value representations to float numbers, matching digits between answers and ground truth, and performing comparisons.

Our reported results may differ from those in other papers due to differences in value extraction and evaluation implementations. Such inconsistencies are evident across multiple prior works, including InvestLM (Yang et al., 2023), PIXIU (Xie et al., 2023), and FinQAPT (Singh et al., 2024), which report divergent FinQA scores for the same models, such as GPT-3.5. These discrepancies arise from the absence of publicly released evaluation code, leading to differences in implementation details. To ensure fairness and consistency, we evaluated all models using our own implementation. Our evaluation code has been released to promote standardized benchmarking practices.

B Baseline Methods

We implement several prompting methods as baselines for comparison: zero-shot Chain-of-Thought (0-CoT) (Wei et al., 2022), decomposed prompting (DP) (Khot et al.), question decomposition CoT (QD-CoT) (Zhou et al.), retrieval CoT (R-CoT) (Trivedi et al., 2023), and few-shot in-context learning (N-shot) (Li et al., 2023). The details of each method and their implementation are as follows:

0-CoT. Zero-shot CoT is a widely used prompting strategy to enhance LLMs' reasoning capabilities. It simply appends the phrase "*Let's think step by step.*" to the end of a question, encouraging the model to provide intermediate reasoning steps before answering.

DP. Decomposed Prompting improves QA by breaking a complex question into sub-questions, which are then answered sequentially. This is a basic way of decomposing questions to improve QA performance, without domain adaptation and QA-specific optimization. It often produces overly detailed decompositions that may negatively impact answer accuracy. We implement this method using both GPT-3.5-turbo (DP1) and GPT-4o (DP2) to ensure a fair and comprehensive comparison.

QD-CoT. This method combines few-shot learning with question decomposition. We implement it by selecting decomposition examples from the ConvFinQA training set and including them in the prompt to guide the model's decomposition process.

R-CoT. Retrieval-based Chain-of-Thought enhances the QA process by incorporating retrieved external knowledge. We use the ConvFinQA dataset as the retrieval corpus and include relevant knowledge to support the model's reasoning.

N-shot. Few-shot in-context learning enhances reasoning through adding examples to the prompt. Following recent adaptations for financial data (Singh et al., 2024), we implement this method using examples from the training split of FinQA dataset. Our implementation uses sentence embeddings generated by OpenAI-Ada-002[3] to identify similar questions from the training set. The annotated reasoning program steps from these similar questions

[3] https://openai.com/blog/new-and-improved-embedding-model

are then incorporated as examples, guiding LLMs to generate analogous reasoning chains for new queries. We report the best results across 1-shot, 3-shot, and 5-shot settings.

C Prompt Design

We use two main types of prompts in our experiments: (1) prompts for QA and (2) prompts for question decomposition. The QA prompts guide the LLMs during answer generation. The decomposition prompts are used both for instruction tuning in the first fine-tuning step and for generating supporting questions during EQD inference.

Question Answering Prompts. The following base prompt is used to guide LLMs in the QA task:

```
You are a financial expert capable of analyzing and answering financial questions based on the given context. Focus on extracting relevant numerical data, simplifying information, and providing concise answers.
```

We slightly adapt this prompt depending on the dataset to ensure consistent output formatting. For example, since the FinQA dataset only contains numerical answers or binary responses (yes/no), we add the following constraint:

```
The final answer must include only a number (rounded to 5 decimal places), the word 'yes', or the word 'no', without any additional explanation or commentary.
```

Question Decomposition Prompt. The following prompt is used for generating decomposed questions, both during model training and inference:

```
You are a financial expert capable of analyzing financial questions. Break down this financial question into simpler sub-questions.
```

D Dataset Details

We conduct testing on four distinct financial datasets: FinQA, TAT-QA (Zhu et al., 2021), ECTQA, and EDTQA. These datasets cover both unstructured and tabular financial content, with varying lengths and source formats. Table 3 summarizes the statistics of these test sets.

ECTQA and EDTQA are derived from ECTSum (Mukherjee et al., 2022) and EDTSum (Xie et al., 2024a), which were originally summarization datasets. We convert them into QA datasets using GPT-4o, which generates questions based on the annotated summarization points. The goal of this conversion is to create QA pairs that focus on numerical reasoning. We used the following prompt to generate questions from individual summary sentences:

```
Focus on the numerical value in the following sentence and convert the sentence into a question. The question should be answerable by a single number only. Provide your output in the following format: \nQuestion: []\nAnswer: []
```

To verify the quality of the generated QA pairs, we manually reviewed 400 examples from ECTQA and 200 from EDTQA. Six QA pairs were judged ambiguous or unclear for ECTQA and only two for EDTQA. The error rates of these two generated datasets are both under 2%. Since each question is generated from a complete sentence, manual verification was straightforward and confirmed the effectiveness and correctness of the generated QA pairs.

Dataset	Resource	Size	Avg. Words
FinQA	Earning Reports	1147	700
TAT-QA	Financial Reports	1663	220
ECTQA	Earning Transcripts	1816	2715
EDTQA	Financial News	1662	714

Table 3: Statistics of QA test sets.

E Fine-tuning Strategy

For the QA expert alignment (second fine-tuning step), we implement three distinct strategies:

1 "Direct Fine-tuning": This strategy involves fine-tuning a new LoRA adapter in step 2, independent of the LoRA adapter from step 1. During the second adapter's fine-tuning, the first adapter is removed. The final EQD model combines the base model with both LoRA adapters simultaneously.

2 "LoRA Continue": This method continuously fine-tunes the LoRA adapter from step 1.

3 "LoRA Extra": This method fine-tunes a new

Strategy	EmAcc
Direct Fine-tuning	42.5
LoRA Continue	54.0
LoRA Extra	44.8

Table 4: Comparison of EQD fine-tuning strategies using Llama3.1-8B-Instruct as the QA model on the FinQA dataset.

Original Question
What is the growth rate in the balance of standby letters of credit from 2006 to 2007?
Decomposed Questions from the Base Llama3.1-8B-Instruct Model
To find the growth rate in the balance of standby letters of credit from 2006 to 2007, we need to break down the problem into simpler sub-questions: 1. What is the balance of standby letters of credit in 2006? 2. What is the balance of standby letters of credit in 2007? 3. How do we calculate the growth rate? Let's assume we have the required data. We'll use the following formula to calculate the growth rate: Growth Rate = ((New Balance - Old Balance) / Old Balance) * 100 We'll need to know the balances for 2006 and 2007 to calculate the growth rate. Sub-question 1: What is the balance of standby letters of credit in 2006? - We need to know the value of the standby letters of credit at the end of 2006. Sub-question 2: What is the balance of standby letters of credit in 2007? - We need to know the value of the standby letters of credit at the end of 2007.
Decomposed Questions from the QD Model after Step 1 Fine-tuning
What was the change in the balance of standby letters of credit from 2006 to 2007? And what was that balance in 2006? How much, then, does that change represent in relation to this 2006 balance?
Decomposed Questions from the EQD Model after Two-step Fine-tuning
What is the balance of standby letters of credit in 2006 and 2007?

Table 5: The evolution of decomposed questions generated by the models after different fine-tuning steps.

LoRA adapter in step 2 while keeping the LoRA adapter from step 1 locked and active. The final EQD model integrates the base model with both LoRA adapters.

We evaluate these fine-tuning strategies on the FinQA dataset using Llama3.1 as the QA model, with results presented in Table 4. The "LoRA Continue" strategy demonstrates superior performance compared to other methods. Based on these findings, we adopt "LoRA Continue" as our primary fine-tuning strategy for the EQD model.

F Results of EQD Models Fine-tuned from Different Base LLMs

We trained our EQD model using three additional LLMs of varying sizes and architectures: Llama3.2-1B-Instruct, Llama3.2-3B-Instruct, and DeepSeek-R1-Distill-Qwen-7B. All models were evaluated with Llama3.1-8B-Instruct as the QA model on the FinQA dataset. The results are summarized in Table 6.

Compared with the results in Table 1, these findings show that although model size affects EQD performance, even smaller models remain highly effective. All EQD-enhanced results on FinQA substantially outperform direct QA without EQD. The 1B model performs slightly worse than the larger models, while the 3B model achieves performance comparable to the 8B model, demonstrating good scalability. Model architecture also has minimal impact on EQD effectiveness, as the DeepSeek-based EQD model performs similarly to the Llama-based model of comparable size. Overall, these results confirm that our method is robust and effective even when applied to smaller LLMs.

G Case Study

G.1 Evolution of Generated Decomposed Questions

We fine-tune a base LLM, Llama3.1-8B-Instruct, by our two-step fine-tuning to an expert question decomposition model. The case study in Table 5 illustrates the model's progressive transformation into an expert decomposition system.

The base Llama3.1-8B-Instruct model initially generates overly detailed reasoning guidance, sim-

EQD base model	EmAcc
Llama3.2-1B-Instruct	52.3
Llama3.2-3B-Instruct	53.9
DeepSeek-R1-Distill-Qwen-7B	54.3

Table 6: Comparison of EQD models fine-tuned from different base LLMs.

Original Question
What is the growth rate in the balance of standby letters of credit from 2006 to 2007?
Decomposed Questions from the EQD Model after Two-step Fine-tuning
What is the balance of standby letters of credit in 2006 and 2007?
Answers to the Decomposed Questions
According to the given context, the balance of standby letters of credit in 2006 and 2007 are as follows: 2006: $4,926 million 2007, $4,711 million
Answers to the Original Question
To find the growth rate, we need to calculate the difference in the balance of standby letters of credit between 2007 and 2006, and then divide it by the balance in 2006. Balance in 2006: $4,926 million Balance in 2007: $4,711 million* Difference: $4,926 million - $4,711 million = -$215 million (negative, indicating a decrease) Growth rate: ($215 million / $4,926 million) 100% $\approx$ -4.36% The balance of standby letters of credit decreased by approximately 4.36% from 2006 to 2007.

Table 7: Answers generated during the QA process using our EQD model, including both the answers to the decomposed sub-questions and the final answer to the original question.

ilar to other LLMs like GPT models shown in Figure 1. After Step 1 (domain fine-tuning), the decomposed questions become more concise and professional, focusing on key steps and domain-specific terms of the original question. Upon completing the full two-step fine-tuning process, the generated questions become highly targeted, with a single question addressing the core challenge in answering the original query.

G.2 Answer Example

Building on the practical case of demonstrating the evolution of decomposed questions, we present the answering process for the final question in Table 7.