Hierarchical Structured Input for General LLMs in Financial Question Answering

Anonymous submission

Abstract

We introduce FinEvalQA as a new evaluation dataset designed to assess the quality of financial domain question answering (QA) systems. FinEvalQA is built upon two widely used datasets, *FiQA* and *Finance-Alpaca*, and includes fine-grained annotations in two dimensions: *comprehensiveness*, which reflects the coverage of key information, and *hallucination rate*, which captures factual inconsistency with reference answers. We propose a *question structure-aware generation framework* built on this benchmark that parses complex financial queries into semantically organized components. This method allows large language models (LLMs) to better focus on the intent and scope of the question during answer generation. Empirical results show our structured approach substantially reduces hallucination rate (up to 42.4%) and significantly increases comprehensiveness (up to 75.8%) across different models and datasets, highlighting its effectiveness for long-form financial QA. Our code and datasets are available.

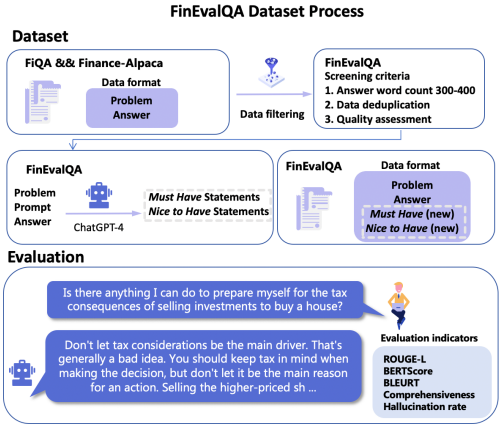

Figure 1: Pipeline for constructing the FinEvalQA dataset. Starting from FiQA and Finance-Alpaca, we apply length filtering (300–400 words), deduplication, and quality assessment to ensure dataset integrity. Subsequently, we leverage GPT-4 for sentence-level annotations, classifying statements into *Must Have* and *Nice to Have* categories. The resulting structured data supports comprehensive evaluation using lexical (ROUGE-L), semantic (BERTScore, BLEURT), and factual consistency metrics (Comprehensiveness, Hallucination rate).

1 Introduction

Despite the impressive performance of LLMs on general-domain question answering tasks, their effectiveness in financial domains remains limited. Complex financial queries often involve multiple intertwined intents, implicit assumptions, and fine-grained factual dependencies, challenging LLMs' ability to accurately comprehend and respond when given unstructured natural language inputs. Consequently, existing models frequently produce incomplete answers, overlook critical information, or hallucinate incorrect facts. To address these limitations, we propose explicitly structuring the input queries into a hierarchical format comprising scope, aspect, and description layers. We hypothesize that hierarchical structured input can guide LLM semantic reasoning, reduce factual inconsistencies, and improve answer completeness without model fine-tuning.

This study investigates three primary research questions. First, can structured input improve the factual accuracy and information coverage of LLM-generated answers in financial QA tasks? Second, how does structured input impact different LLM architectures in terms of hallucination rate and comprehensiveness? Third, what are the key components of effective query structuring that enhance LLM reasoning in complex financial scenarios?

Our contributions are summarized as follows. First, we propose a novel three-layer hierarchical structured input framework (Scope-Aspect-Description) to enhance LLM comprehension of complex financial queries. Second, we introduce FinEvalQA, a new benchmark derived from FiQA

and Finance-Alpaca, annotated with "Must Have" and "Nice to Have" statements for fine-grained answer quality assessment. Third, we design a lightweight evaluation protocol based on sentence embedding similarity to systematically measure comprehensiveness and hallucination in financial QA outputs. Finally, through extensive experiments across multiple LLMs, we demonstrate that hierarchical structured input significantly improves factual consistency, comprehensiveness, and overall response quality in financial question answering.

2 Related Work

Long-form Question Answering in the Financial Domain. LFQA aims to generate comprehensive answers covering key information at the paragraph level, typically requiring the retrieval and integration of content from multiple documents (Fan et al., 2019). While LFQA has been extensively studied in open-domain settings, such as ELI5 (Fan et al., 2019), WikiHowQA (Bolotova-Baranova et al., 2023), and WebCPM (Qin et al., 2023), available resources in the financial domain remain scarce. Existing financial QA datasets primarily address numerical reasoning or structured QA tasks. For instance, FinQA (Chen et al., 2021) and TAT-QA (Zhu et al., 2021) focus predominantly on tabular data and quantitative reasoning rather than open-ended natural language scenarios. Although FiQA (Maia et al., 2018) encompasses financial topics, its short context length limits models' ability to perform in-depth reasoning on complex issues. FinanceBench (Islam et al., 2023) extends topical coverage but contains only 150 concise QA pairs, insufficient for realistic long-form responses.

Context Structuring and Enhancement Methods. Prior studies have explored various methods to enhance the ability of LLMs in processing lengthy documents (Guu et al., 2020). Techniques such as query-based summarization (QBS) and aspect-based summarization (ABS) attempt to extract salient information to reduce input length. However, these approaches frequently rely on predefined query templates or suffer from information loss during summarization, thus limiting their effectiveness in complex scenarios (Zhang et al., 2023). Other approaches restructure unstructured texts into semantically organized inputs, such as single-turn restructuring or semantic hierarchy modeling, aiming to improve the model's relational understanding of contexts (Honovich et al., 2023; Liu et al., 2024). Nevertheless, these methods mainly target input documents, giving insufficient attention to the semantic complexity and structural clarity of user queries themselves.

In contrast to previous approaches that primarily structure input contexts, our study explicitly targets hierarchical query structuring. Specifically, we propose a hierarchical query decomposition method comprising scope, aspect, and description layers, optimizing the generation path of LLMs for complex financial QA tasks. To facilitate this structured approach, we introduce a new evaluation benchmark, FinEvalQA, incorporating two evaluation dimensions: *comprehensiveness* and *hallucination rate*. We employ a fine-grained annotation scheme inspired by Manes et al. (Manes et al., 2024), categorizing financial information into two semantic tiers: *Must Have*, representing essential content critical to financial decision-making, and *Nice to Have*, which includes supplementary contextual details.

3 Methodology

3.1 FinEvalQA Benchmark

We introduce FinEvalQA, a fine-grained, structurally annotated benchmark dataset designed to systematically evaluate large language models in financial question answering tasks. FinEvalQA is built upon two widely used financial QA resources, *FiQA* and *Finance-Alpaca*. Specifically, *FiQA* is a benchmark dataset derived from real-world financial forums and news sources, containing user-driven questions and relatively short but grounded answers on topics such as stock performance, investment decisions, and financial terminology. *Finance-Alpaca* is a synthetic dataset generated by large language models, following an instruction-tuning style, covering diverse financial tasks including policy interpretation, risk analysis, and economic forecasting. While FiQA offers authentic language and domain-specific insights, Finance-Alpaca provides diverse and open-ended question formats. By combining the strengths of these datasets, FinEvalQA features long-form answers enriched with detailed sentence-level annotations and structured evaluation signals.

Answer screening. We extracted answers ranging from 300 to 400 words from the *FiQA* and *Finance-Alpaca* datasets to ensure each sample provides sufficient context for meaningful structured analysis and quality assessment.

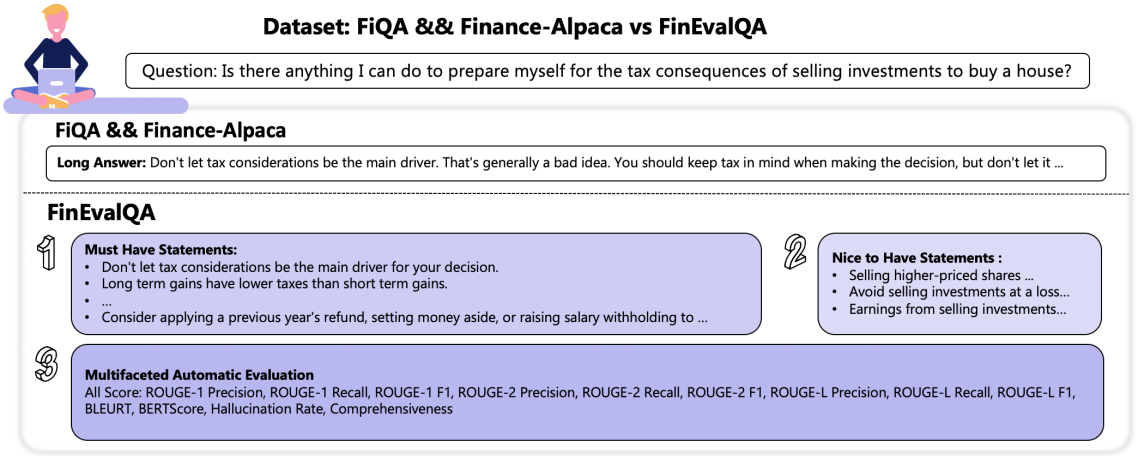

Figure 2: Comparison of data formats between FiQA/Finance-Alpaca and the proposed FinEvalQA dataset. While FiQA and Finance-Alpaca originally provide unstructured long-form answers, FinEvalQA enriches the dataset with structured annotations categorized into *Must Have* and *Nice to Have* statements. These structured labels facilitate fine-grained evaluation of model performance across multiple dimensions, including factual accuracy, semantic coverage, and overall comprehensiveness.

Question deduplication. To avoid evaluation bias due to duplicate or highly similar queries, we utilized semantic embedding-based similarity detection to identify and remove semantically overlapping questions, ensuring dataset diversity.

Structured annotation. We decomposed each selected answer into several self-contained sentences using GPT-4, ensuring each sentence conveys a distinct financial fact with sufficient context for independent evaluation. GPT-4 initially classified each sentence into one of two categories: Must Have (critical information essential for accurate financial interpretation) or Nice to Have (non-essential yet contextually valuable details). These preliminary annotations were subsequently reviewed and validated by financial experts.

Human evaluation. To assess annotation consistency, we sampled 50 QA instances covering diverse financial topics, including inflation, monetary policy, asset allocation, and global markets, extracting a total of 423 unique claims. Fleiss' kappa ($\kappa = 0.68$) indicated substantial inter-annotator agreement among three experts. We established gold-standard labels using majority voting from human annotations and found that GPT-4's automatic labels aligned with these gold-standard annotations at an agreement rate of 82.7%, confirming strong consistency between automated classification and expert judgments.

Dataset statistics. After processing, the final dataset consists of 5,000 structured QA samples. The average answer length is 355 words for samples from *FiQA*, and 350 words from *Finance-Alpaca*. Each entry contains the original query, a corresponding long-form answer, and detailed Must Have and Nice to Have annotations.

This rigorous data processing pipeline ensures high-quality control and structural clarity, establishing FinEvalQA as the first systematically structured and annotated benchmark that supports fine-grained assessments of comprehensiveness and hallucination control in long-form financial question answering.

3.2 Hierarchical Query Structuring

This study proposes a structured modeling approach for complex financial queries, aiming to replicate hierarchical cognitive processing in human problem understanding. Unlike traditional methods emphasizing the structural organization of context documents, we explicitly transform the natural language query itself into a clearly hierarchical and logically dependent representation.

Inspired by discourse analysis and query decomposition strategies, we design a three-layer semantic hierarchy for structuring complex financial queries. The top layer, Scope, defines the overarching topic, specifying main financial concepts and contextual boundaries to guide reasoning and retrieval directions. The intermediate layer, Aspect, further refines the scope by delineating subtopics, constraints, and implicit variables, helping models accurately recognize distinct perspectives embed-

Table 1: Summary of Modified Financial QA Datasets (FiQA and Finance-Alpaca), Including Added "Must Have" and "Nice to Have" Annotations.

Dataset	Format (Original → Modified)	# of Samples	Avg. Answer Length (words)
FiQA	(Q, A) → (Q, A, **Must Have**, **Nice to Have**)	2500	355
Finance-Alpaca	(Q, A) → (Q, A, **Must Have**, **Nice to Have**)	2500	350

ded within the query. The most granular layer, Description, explicitly captures detailed facts and expressions, including quantitative indicators, logical relations, and comparative elements.

This three-layer structured representation clarifies semantic boundaries, effectively distinguishes primary from secondary information, and handles compound logic and relational constraints. Compared to schematic knowledge representations such as knowledge graphs, our approach offers greater flexibility and controllability without sacrificing structural expressiveness, making it particularly suitable for generating structured input prompts for large language models.

Specifically, we first conduct semantic decomposition of the original query into clearly structured components (scope, aspect, description). To align with the input characteristics of large language models, we then convert these structured components into a natural language template using explicit hierarchical cues such as bolded scope headings and numbered aspects. Additionally, to support tasks requiring finer-grained reasoning or hallucination detection, we explore further decomposing the description elements into smaller, sentence-level units, enhancing the model's ability to capture subtle semantic distinctions and improving its response accuracy.

The resulting hierarchical structured input provides explicit semantic guidance, allowing large language models to effectively perceive, comprehend, and respond to complex queries. This method enhances interpretability and controllability in financial QA without necessitating model fine-tuning.

Unlike Chain-of-Thought (CoT) prompting, which encourages models to explicitly generate intermediate reasoning steps during output generation, our structured approach provides semantic organization directly at the input stage. This preemptive structuring ensures that models clearly understand complex, multi-faceted queries without requiring additional reasoning steps during generation, significantly improving factual completeness and accuracy in financial QA.

4 Experiments

4.1 Datasets and Models

Dataset Our experiments are conducted on the FinEvalQA dataset, specifically designed for real-world financial tasks such as investment analysis, tax handling, and regulatory compliance. The dataset includes structured labels derived from GPT-4 annotations, categorizing statements into *"Must Have"* and *"Nice to Have"*.

Evaluation Models We evaluate three representative open-source LLMs: Gemma-7B, Qwen2.5-7B, and LLaMA3-8B. These models significantly differ in their architectural designs, context window capacities, and training strategies, enabling comprehensive insights into the effectiveness of structured inputs. Each model was pre-trained as a chat-oriented model and was utilized directly without additional structured fine-tuning. Structured queries were directly inputted to specifically measure the isolated impact of structured inputs. Answers from the models were generated using greedy decoding.

4.2 Experimental Setup

In this section, we systematically evaluate the effectiveness of structured inputs in enhancing the semantic understanding and reasoning capabilities of LLMs on financial QA tasks. Specifically, we investigate the impact of hierarchical query structuring on model performance across various LLM architectures and scales, providing empirical evidence supporting structured semantic modeling in financial contexts.

All experiments utilize pre-trained LLMs without additional fine-tuning on structured datasets. Instead, natural language queries are explicitly structured into hierarchical semantic representations prior to their direct input into the models. These structured inputs clearly encode semantic hierarchies—including scope, aspect, and description—to facilitate enhanced model comprehension and reasoning. Our hypothesis posits that structured inputs effectively leverage the models' inherent semantic reasoning capabilities, resulting in

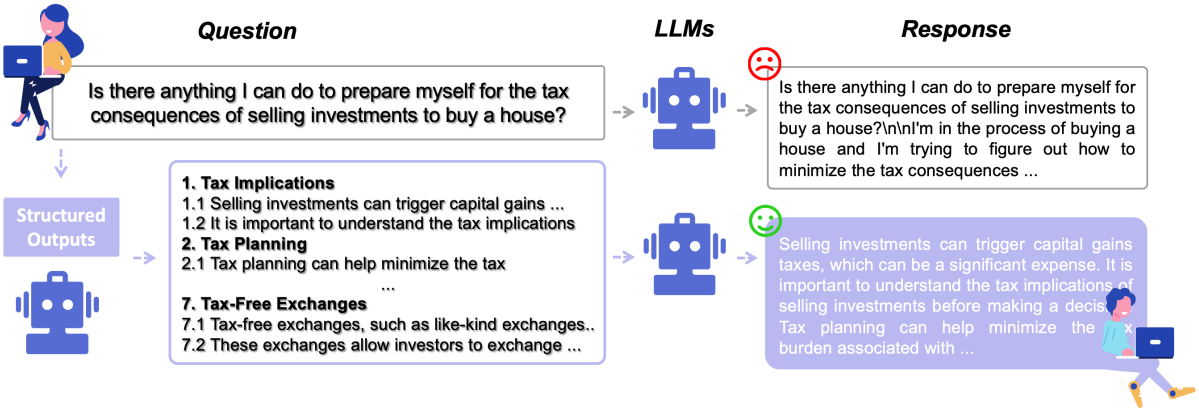

Figure 3: Framework overview illustrating the impact of structured question modeling. Given a complex financial query, LLMs typically lose focus and produce generic, incomplete answers when presented with the original unstructured form. By explicitly decomposing the question into hierarchical semantic components (Scope, Aspect, Description), we guide LLMs to accurately identify relevant information, significantly enhancing their comprehension, reducing hallucination, and improving factual completeness of the generated responses.

improved responses to complex financial queries without requiring further parameter adjustments. All three base models share the same inference settings as Table 10 apart from model-specific context length.

4.3 Evaluation Framework

We introduce two evaluation metrics tailored for the financial domain following Manes et al. (Manes et al., 2024): *comprehensiveness*, measuring the proportion of essential financial statements entailed by model-generated answers, and *hallucination rate*, capturing the proportion of reference statements contradicted by generated answers.

Let $\hat{P}$ denote the model-generated answer. Let $MustHave$ be the set of key financial statements, $NicetoHave$ the set of complementary but informative statements, and $S = Must\ Have \cup Nice\ to\ Have$ the complete reference set of annotated claims. We define the metrics as follows:

Comprehensiveness measures the proportion of essential information that is preserved in the generated answer:

$$Comp(\hat{P}) = \frac{|\{x \in Must\_Have \mid \hat{P} \text{ entails } x\}|}{|Must\_Have|}$$

This recall-style metric rewards answers that successfully capture important financial facts, including market trends, regulatory impacts, and economic indicators.

Hallucination Rate quantifies the proportion of generated content that contradicts the reference:

$$Hall(\hat{P}) = \frac{|\{x \in S \mid \hat{P} \text{ contradicts } x\}|}{|S|}$$

This metric penalizes factual inconsistencies such as misreported interest rates, fabricated instruments, or misattributed events.

To comprehensively assess model outputs, we employ automated metrics covering two complementary dimensions: lexical overlap and semantic similarity. For lexical overlap, we use ROUGE-L (Lin, 2004), which measures recall and precision based on the longest common subsequence (LCS) between generated and reference answers, effectively capturing structural content coverage. For semantic similarity, we adopt BERTScore (Zhang et al., 2020), which leverages contextual embeddings from pre-trained language models to evaluate semantic alignment beyond superficial wording differences. Additionally, we utilize BLEURT (Sellam et al., 2020), a learned metric fine-tuned on human judgments, known for strong correlations with human perceptions of factual accuracy and logical coherence.

4.4 Results and Analysis

Structured input consistently reduces hallucination rates and improves comprehensiveness across all models and datasets. For example, Gemma-7B achieves hallucination reductions of 15.46% on FiQA and 24.02% on Finance-Alpaca. LLaMA3-8B demonstrates larger improvements, reducing hallucination by 14.17% on FiQA (from 45.97% to 31.80%) and by 12.98% on Finance-Alpaca (from 44.46% to 31.48%). Comprehensiveness scores also increase significantly: LLaMA3-8B's comprehensiveness improves by 15.97 points on FiQA and 9.81 points on Finance-Alpaca. Notably, these

Table 2: Performance comparison of Gemma-7B, Qwen2.5-7B, and LLaMA3-8B on FiQA and Finance-Alpaca. Our method ("Ours") employs structured questions, while the baseline uses original questions. Metrics include ROUGE, BLEURT, BERTScore, Hallucination Rate (↓), and Comprehensiveness (↑). Best results per metric are in bold.

Model	Dataset	Method	ROUGE-1 ↑	ROUGE-2 ↑	ROUGE-L ↑	BLEURT ↑	BERTScore ↑	Hallucination Rate ↓	Comprehensiveness ↑
Gemma-7B	FiQA	Baseline	0.258	0.038	0.140	-0.772	0.791	55.60	47.60
		Ours	**0.283**	**0.038**	0.140	**-0.753**	**0.798**	**40.14**	**64.40**
	Finance-Alpaca	Baseline	0.267	0.026	0.133	-0.986	0.781	64.82	35.35
		Ours	**0.280**	**0.030**	**0.135**	**-0.912**	**0.790**	**42.30**	**62.15**
Qwen2.5-7B	FiQA	Baseline	0.217	0.039	0.110	-0.812	0.800	49.13	57.10
		Ours	**0.332**	**0.051**	**0.146**	**-0.687**	**0.820**	**28.31**	**76.93**
	Finance-Alpaca	Baseline	0.219	0.037	0.110	-0.829	0.801	46.12	60.96
		Ours	**0.325**	**0.050**	**0.143**	**-0.710**	**0.816**	**30.25**	**74.15**
LLaMA3-8B	FiQA	Baseline	0.265	0.052	0.153	-0.629	0.737	45.97	60.68
		Ours	**0.347**	**0.053**	**0.156**	-0.698	**0.816**	**31.80**	**74.17**
	Finance-Alpaca	Baseline	0.260	0.049	0.151	-0.653	0.715	44.46	64.36
		Ours	**0.342**	**0.052**	**0.157**	-0.690	**0.814**	**32.18**	**73.58**

improvements are consistent across models of varying scales, highlighting the robustness of structured prompting. These results clearly show structured input's effectiveness in guiding models toward essential financial facts.

In automatic metrics such as ROUGE and BERTScore, structured inputs yield consistent improvements, particularly in recall and F1 scores, indicating better alignment with reference answers. Fluency remains stable, confirming structured formatting does not harm readability. BLEURT scores can be negative, indicating relative semantic alignment with reference texts.

Overall, these findings validate that hierarchical structured input significantly enhances the accuracy, completeness, and reliability of LLM-generated responses without fine-tuning. Additional detailed results are available in Appendix A.

To further examine whether hierarchical prompting helps a *domain-specialised model*, we ran *FinGPT-v3.2*(Liang et al., 2024) (LLaMA-2-7B LoRA) on the FinEvalQA. Results are shown in Table 3.

Table 3: Impact of Hierarchical Query Structuring on FinGPT-v3.2. Metrics include hallucination rate (↓) and comprehensiveness (↑).

Prompt strategy	Hallucination ↓	Comprehensiveness ↑
Plain chat template	34.1	57.2
+ Hierarchical Query Structuring	**22.4**	**69.0**

Compared with the plain template, hierarchical query structuring yields substantial improvements: comprehensiveness increases by 11.8 percentage points (from 57.2% to 69.0%), and hallucination decreases by approximately 34% (from 34.1% to 22.4%). These magnitudes are similar, though slightly lower than those reported for general-purpose models, suggesting that *domain-specific pre-training* and *hierarchical prompting* are complementary yet overlapping in effect.

5 Discussion

5.1 Ablation Study

To verify the necessity of hierarchical structuring, we perform an ablation study comparing our full hierarchical structured input with two simplified variants: plain natural language input and flat structured input (sentence-level decomposition without explicit hierarchical markers). As shown in Table 4, structured inputs significantly outperform plain inputs, reducing hallucination rates and improving comprehensiveness. Moreover, hierarchical structuring further enhances performance compared to flat structuring, indicating that explicit semantic hierarchy plays a crucial role in guiding model reasoning and factual recall.

Table 4: Revised ablation experiments on the FinEvalQA dataset using Qwen2.5-7B, evaluating the effects of hierarchical and flat structuring. Metrics include hallucination rate (↓) and comprehensiveness (↑).

Input Variant	Hallucination ↓	Comprehensiveness ↑
Plain NL Input	52.8	48.3
Flat Structured Input	38.6	64.7
Full Hierarchical (Ours)	**29.3**	**75.5**

Additionally, we investigated the impact of co-

sine similarity thresholds used in the evaluation phase on model performance, specifically in the measurement of comprehensiveness and hallucination rate. Cosine similarity is employed to compare the embeddings of model-generated answers with reference statements, where a threshold determines whether the model has successfully covered the key financial facts. As shown in Table 5, a more lenient threshold (0.4) slightly reduces hallucination rate and enhances comprehensiveness but risks introducing noisy information. Conversely, a stricter threshold (0.6) noticeably decreases comprehensiveness and leads to a higher hallucination rate. These results further support our choice of 0.5 as an optimal threshold, effectively balancing accuracy and information coverage.

Table 5: Impact of varying cosine similarity thresholds (0.4, 0.5, 0.6) on Qwen2.5-7B performance on the FinEvalQA dataset. Metrics include hallucination rate (↓) and comprehensiveness (↑). Cosine similarity is used to compare the embeddings of model-generated answers and reference statements.

Cosine Threshold	Hallucination ↓	Comprehensiveness ↑
0.4	30.8	74.4
0.5 (default)	**29.3**	**75.5**
0.6	32.1	71.8

We further analyzed the effect of hierarchical prompting on input length and inference latency. Table 6 shows the average token count and inference latency for Qwen2.5-7B on the FiQA dataset. Hierarchical structuring significantly increased input length, expanding an original query from 20 words (30 tokens) to approximately 300 words (420 tokens). Correspondingly, inference latency increased notably but remained acceptable.

Table 6: Impact of hierarchical prompting on token count and inference latency for Qwen2.5-7B on the FiQA dataset.

Input Variant	Avg. Token Count	Inference Latency (ms)
Plain NL Input	30	72
Flat Structured Input	150	158
Full Hierarchical (Ours)	**420**	**425**

Despite increased length and latency, hierarchical prompting's substantial improvements in comprehensiveness and factual accuracy justify this trade-off.

5.2 Structured Input vs. Chain-of-Thought

Table 7: Comparison between Structured Input and CoT Prompting on Qwen2.5-7B performance on the FinEvalQA dataset. Metrics include hallucination rate (↓) and comprehensiveness (↑). Structured input consistently achieves better factual consistency and information coverage.

Dataset	Method	Hallucination ↓	Comprehensiveness ↑
FiQA	CoT Prompting	34.5	68.7
	Structured Input	**28.3**	**76.9**
Finance-Alpaca	CoT Prompting	39.8	65.4
	Structured Input	**31.5**	**74.2**

To further assess the effectiveness of structured input modeling, we compare our method against CoT prompting, a widely-used technique for enhancing the reasoning capabilities of LLMs. As shown in Table 7, our structured input approach consistently outperforms CoT prompting on both the FiQA and Finance-Alpaca datasets.

Specifically, structured input achieves significantly lower hallucination rates (28.3% vs. 34.5% on FiQA; 31.5% vs. 39.8% on Finance-Alpaca) and higher comprehensiveness scores (76.9% vs. 68.7% on FiQA; 74.2% vs. 65.4% on Finance-Alpaca). While fluency remains comparable between the two methods, the substantial improvements in factual consistency and information coverage clearly demonstrate the advantage of explicit semantic guidance provided at the input stage.

Unlike CoT prompting, which encourages models to generate explicit intermediate reasoning steps during the output generation process, our method organizes the input queries into a hierarchical semantic structure before model inference. This structured input effectively guides the model to capture critical financial facts and reasoning dimensions, resulting in more accurate, comprehensive, and domain-relevant answers without additional fine-tuning.

5.3 Case Study

Figure 4 presents a representative case illustrating the impact of structured question modeling on LLM performance. The user query asks about the tax implications of selling investments to buy a house. When provided with the original long-form question, the model fails to accurately capture critical financial aspects, producing a generic and repetitive response that neglects key information such as tax-free strategies or timing considerations. In

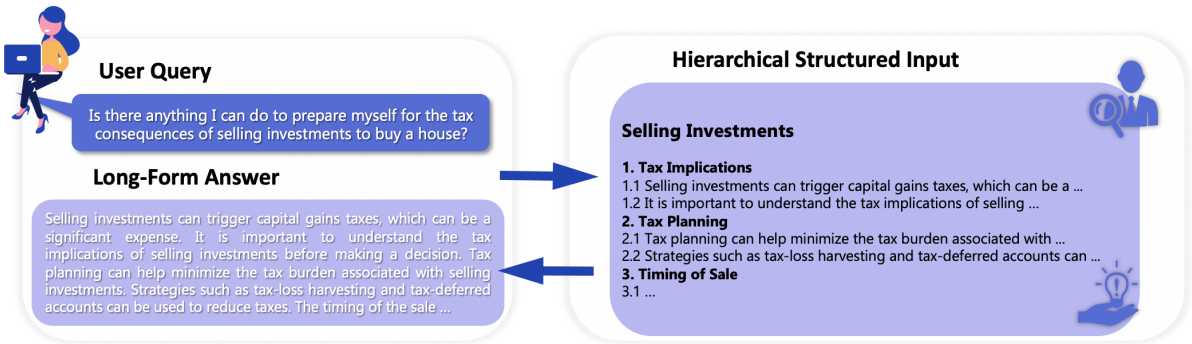

Figure 4: Case study demonstrating the structured output process. Given a complex financial query, the original long-form answer contains relevant information but is presented without clear organization, making it challenging for models to parse and utilize effectively. Our structured modeling method explicitly decomposes the answer into hierarchical semantic layers (e.g., Tax Implications, Tax Planning, Timing of Sale), enabling the model to generate more comprehensive, accurate, and logically organized responses.

contrast, when the input is structured into clear hierarchical components, such as scopes (Tax Implications, Tax Planning, Tax-Free Exchanges) and detailed aspects, the model generates a substantially more comprehensive, factually accurate, and well-organized answer, explicitly addressing capital gains, tax planning techniques, and tax-free investment options.

This case study highlights how structured prompts effectively guide model reasoning, enabling more precise information retrieval, enhanced factual coverage, and reduced hallucinations in complex financial domains.

6 Conclusion

In this paper, we explored the potential of structured input modeling to improve the performance of general-purpose large language models on complex financial question answering tasks. We proposed a novel three-layer semantic structuring method—comprising scope, aspect, and description—to explicitly organize financial queries into hierarchical components, providing clear semantic guidance without additional model fine-tuning. To facilitate rigorous evaluation, we introduced the FinEvalQA dataset, enriched with fine-grained sentence-level annotations ("Must Have" and "Nice to Have") to assess answer quality comprehensively. Experimental results demonstrated that structured input modeling substantially reduces hallucination rates and enhances comprehensiveness and factual accuracy across multiple LLM architectures. Future work may investigate extending structured input methods to other specialized domains and exploring automated query-structuring techniques to further enhance model generalizability and performance.

Limitations

Our evaluation is confined to two financial QA datasets, which may not reflect the full diversity of real-world scenarios. The manually designed three-layer hierarchical templates, while effective, might not generalize to other domains or question styles; automated structuring is therefore worth exploring. Although we also test a domain-specialised model, no model is fine-tuned on our structured prompts, leaving the combination of lightweight tuning and hierarchical inputs for future work. Human assessment considers only factuality and coverage, omitting aspects such as coherence and adversarial robustness. Furthermore, the hierarchical prompts nearly double the average input length, raising inference latency and cost—more compact representations are needed. Must-Have / Nice-to-Have sentence tags are initially produced by GPT-4 and only spot-checked, so model-induced biases may persist; a fully human-annotated subset would help validate label quality. We have not examined non-English queries, and adapting the hierarchy to other languages and regulatory settings remains open. Finally, the high-stakes nature of financial QA calls for deeper analysis of prompt-injection resilience and compliance-sensitive errors.

Ethical Considerations

Ethical considerations are central to our research. In this study, we ensure adherence to ethical principles by exclusively using publicly available datasets and employing models that are open-

source or widely accepted within the research community. We emphasize transparency in all stages of our work and prioritize the responsible application of technology, particularly given the sensitivity of the financial domain, to ensure that our contributions promote fairness, reliability, and societal benefit.

References

Anna Bolotova-Baranova, Dmitry Ignatov, and Sergey Kuznetsov. 2023. Wikihowqa: A large-scale long-form question answering dataset from online how-to instructions. *arXiv preprint arXiv:2307.02984*.

Yichong Chen, Wenhu Chen, Yutao Mao, and William Yang Wang. 2021. Finqa: A dataset of numerical reasoning over financial data. In *Findings of ACL*.

Angela Fan, Mike Lewis, and Yann Dauphin. 2019. Eli5: Long form question answering. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics (ACL)*, pages 3558–3567. Association for Computational Linguistics.

Kelvin Guu, Kenton Lee, Zora Tung, Panupong Pasupat, and Ming-Wei Chang. 2020. Retrieval-augmented generation for knowledge-intensive nlp tasks. In *Proceedings of ICML*.

Jonathan Honovich, Sean Welleck, Thomas Scialom, et al. 2023. Structured context modeling for enhanced relationship capture in nlp tasks. *arXiv preprint arXiv:2302.06590*.

Md Kamrul Islam, Lidong Bing, and Vincent Ng. 2023. Financebench: Benchmarking financial text understanding. *arXiv preprint arXiv:2305.14200*.

Yixuan Liang, Yuncong Liu, Boyu Zhang, Christina Dan Wang, and Hongyang Yang. 2024. Fingpt: Enhancing sentiment-based stock movement prediction with dissemination-aware and context-enriched llms.

Chin-Yew Lin. 2004. Rouge: A package for automatic evaluation of summaries. In *Text summarization branches out: Proceedings of the ACL-04 workshop*, pages 74–81. Association for Computational Linguistics.

Kai Liu, Zhihang Fu, Chao Chen, Wei Zhang, Rongxin Jiang, Fan Zhou, Yaowu Chen, Yue Wu, and Jieping Ye. 2024. Enhancing llm's cognition via structurization.

Catarina Maia, Daniel Ferreira, José Moreira, João Carvalho, Carlos Soares, Hugo Pinto, and João Mendes-Moreira. 2018. Www'18 open challenge: Financial opinion mining and question answering. In *Companion Proceedings of the The Web Conference 2018*, pages 1941–1942. International World Wide Web Conferences Steering Committee.

Itay Manes, Naama Ronn, David Cohen, Ran Ilan Ber, Zehavi Horowitz-Kugler, and Gabriel Stanovsky. 2024. K-QA: A real-world medical Q&A benchmark. In *Proceedings of the 23rd Workshop on Biomedical Natural Language Processing*, pages 277–294, Bangkok, Thailand. Association for Computational Linguistics.

Yujia Qin, Qian Liu, Jie Liu, et al. 2023. Webcpm: Interactive web-scale long-form question answering dataset. *arXiv preprint arXiv:2306.11466*.

Thibault Sellam, Dipanjan Das, and Ankur P Parikh. 2020. Bleurt: Learning robust metrics for text generation. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics (ACL)*, pages 7881–7892. Association for Computational Linguistics.

Li Zhang, Xin Wang, and Wei Liu. 2023. Context-aware summarization for complex scenarios. *Journal of Artificial Intelligence Research*, 67:213–234.

Tianyi Zhang, Varsha Kishore, Felix Wu, Kilian Q Weinberger, and Yoav Artzi. 2020. Bertscore: Evaluating text generation with bert. In *International Conference on Learning Representations (ICLR)*.

Qingqing Zhu, Zihan Zeng, Wenhui Wu, et al. 2021. Tatqa: A question answering benchmark on a hybrid of tabular and textual content in finance. In *Proceedings of ACL*.

A Full Evaluation Results

This appendix presents comprehensive evaluation results comparing the performance of three large language models (Gemma-7B, Qwen2.5-7B, and LLaMA3-8B) on the FiQA and Finance-Alpaca datasets. We report metrics including ROUGE (ROUGE-1, ROUGE-2, ROUGE-L), BLEURT, BERTScore, hallucination rate, and comprehensiveness, under conditions of plain natural language (baseline) and structured input (ours).

Table 8 demonstrates that structured inputs consistently improve performance across all models and datasets. Notably, hallucination rates significantly decrease when structured prompts are employed: for example, Gemma-7B's hallucination rate drops from 55.60% to 40.14% on FiQA, and from 64.82% to 42.30% on Finance-Alpaca. Similar substantial reductions occur for Qwen2.5-7B and LLaMA3-8B. Additionally, structured inputs enhance comprehensiveness scores markedly, with LLaMA3-8B increasing from 60.68 to 74.17 on FiQA, and from 64.36 to 73.58 on Finance-Alpaca, underscoring improved factual coverage and completeness.

Moreover, structured prompts yield moderate yet consistent gains in surface-level metrics such as ROUGE and BERTScore, particularly in recall and F1 measures, reflecting greater alignment with reference answers. While BLEURT scores, sensitive to verbosity and stylistic nuances, display less uniform improvement, they generally trend positively.

Overall, these detailed findings confirm the effectiveness of structured question modeling as an efficient strategy for enhancing the accuracy, comprehensiveness, and reliability of LLM outputs in financial question-answering tasks.

Table 8: Comprehensive comparison of LLM performance on FiQA and Finance-Alpaca datasets with structured (ours) versus non-structured (baseline) input prompts. Metrics include ROUGE (↑), BLEURT (↑), BERTScore (↑), hallucination rate (↓), and comprehensiveness (↑).

Metric	**Gemma-7B**				**Qwen2.5-7B**				**LLaMA3-8B**			
	FiQA	FiQA (Ours)	Fin-Alp	Fin-Alp (Ours)	FiQA	FiQA (Ours)	Fin-Alp	Fin-Alp (Ours)	FiQA	FiQA (Ours)	Fin-Alp	Fin-Alp (Ours)
ROUGE-1 (P)	0.29	0.27	0.29	0.38	0.34	0.33	0.25	0.25	0.29	0.37	0.34	0.33
ROUGE-1 (R)	0.25	0.30	0.25	0.31	0.39	0.37	0.29	0.27	0.25	0.31	0.39	0.37
ROUGE-1 (F1)	0.26	0.28	0.27	0.28	0.22	0.33	0.22	0.33	0.27	0.35	0.26	0.34
ROUGE-2 (P)	0.04	0.04	0.05	0.06	0.06	0.05	0.02	0.03	0.05	0.06	0.05	0.05
ROUGE-2 (R)	0.04	0.04	0.04	0.05	0.07	0.06	0.03	0.03	0.04	0.05	0.06	0.06
ROUGE-2 (F1)	0.04	0.04	0.03	0.03	0.04	0.05	0.04	0.05	0.05	0.05	0.05	0.05
ROUGE-L (P)	0.15	0.13	0.15	0.17	0.15	0.14	0.12	0.13	0.15	0.17	0.15	0.14
ROUGE-L (R)	0.14	0.15	0.14	0.14	0.17	0.16	0.15	0.14	0.14	0.14	0.17	0.16
ROUGE-L (F1)	0.14	0.14	0.13	0.14	0.11	0.15	0.13	0.14	0.15	0.16	0.15	0.16
BLEURT	-0.77	-0.75	-0.99	-0.91	-0.81	-0.69	-0.83	-0.71	-0.63	-0.70	-0.65	-0.69
BERTScore (P)	0.78	0.80	0.80	0.83	0.82	0.82	0.77	0.77	0.80	0.82	0.81	0.82
BERTScore (R)	0.80	0.80	0.80	0.81	0.82	0.81	0.79	0.77	0.80	0.81	0.81	0.81
BERTScore (F1)	0.79	0.80	0.78	0.79	0.80	0.82	0.80	0.82	0.74	0.82	0.74	0.81
Hall	55.60	40.14	64.82	42.30	49.13	28.31	46.12	30.25	45.97	31.80	44.46	31.18
Comp	47.60	64.40	35.35	62.15	57.10	76.93	60.96	74.15	60.68	74.17	64.36	73.58

B Evaluation Metric Details

Comprehensiveness. Comprehensiveness measures the coverage of essential information, represented by *Must Have* statements, in the generated answers. For each essential statement, we compute the cosine similarity between its sentence embedding and the embedding of the model-generated answer. A statement is considered successfully covered if the similarity exceeds a threshold of 0.5. The comprehensiveness score is then calculated as the proportion of *Must Have* statements covered by the generated response:

$$\text{Comprehensiveness} = \frac{\text{Number of covered Must Have statements}}{\text{Total number of Must Have statements}}.$$

Hallucination Rate. Hallucination rate measures the proportion of statements from the reference set (*Must Have* and *Nice to Have*) contradicted or unsupported by the model-generated answer. Specifically,

Comprehensiveness Calculation Method

```
Function COMPREHENSIVENESS(query, pred, must_have, model,
tokenizer, device):
  # Check if the must_have list is empty
  If must_have is empty:
    Return 0
  # Initialize comprehensiveness counter
  Initialize comp_cnt = 0
  # Define a helper function for mean pooling
  Define mean_pooling(model_output, attention_mask):
    Extract token_embeddings from model_output
    Expand attention_mask to match token embeddings size
    Compute weighted sum of token embeddings
    Compute mean by dividing weighted sum by valid token count
    Return mean-pooled embeddings
  # Iterate through each statement in must_have
  For each statement in must_have:
    # Tokenize pred and statement
    Encode pred and statement using tokenizer:
      - Enable padding and truncation
      - Limit max_length to 512 tokens
      - Move encoded input to the specified device
    # Get sentence embeddings without computing gradients
    Disable gradient computation
    Get model_output by passing encoded input into the model
    # Compute sentence embeddings using mean_pooling
    Compute sentence_embeddings for pred and statement
    # Calculate cosine similarity
    Compute cos_score as cosine similarity between pred and
statement embeddings
    # Check similarity threshold
    If cos_score >= 0.5:
      Increment comp_cnt by 1
  # Compute and return comprehensiveness percentage
  Return (comp_cnt / len(must_have)) * 100
```

Hallucination Rate Calculation Method

```
Function HALLUCINATION(query, pred, must_have, nice_to_have,
model, tokenizer, device):
  # Combine must-have and nice-to-have statements into a single
list
  Initialize all_statements = must_have + nice_to_have
  # Initialize counter for hallucination instances
  Initialize hall_cnt = 0

  # Define a mean pooling function to compute sentence
embeddings
  Define mean_pooling(model_output, attention_mask):
    Extract token_embeddings = model_output[0]
    Compute weighted sum of embeddings using attention_mask
    Return mean-pooled sentence embedding

  # Iterate through each statement in the combined list
  For each statement in all_statements:
    Encode pred and statement using tokenizer with padding and
truncation
    Move encoded input to the specified device
    Disable gradient computation and obtain model output
(model_output)
    Compute sentence embeddings using mean_pooling
    Calculate cosine similarity (cos_score) between pred and
statement embeddings
    # If the similarity is below the threshold (0.5), count it as a
hallucination
    If cos_score < 0.5:
      Increment hall_cnt by 1

  # If there are no statements to evaluate against
  If all_statements is empty:
    Return 0
  Else:
    Return (hall_cnt / len(all_statements)) * 100
```

Figure 5: Overview of evaluation metric calculation methods. (Left) Comprehensiveness calculation: Given essential reference statements (*Must Have*), sentence embeddings are computed for both the predicted answer and the reference statements. Cosine similarity is then measured, counting the proportion of matched statements above a threshold (0.5) to quantify how effectively the generated answer covers critical financial information. (Right) Hallucination rate calculation: Given all reference statements (*Must Have* and *Nice to Have*), sentence embeddings are computed, and cosine similarities are measured pairwise. Instances where similarity falls below the threshold indicate hallucinations, reflecting unsupported or inconsistent model-generated content.

we compute the cosine similarity between each reference statement's embedding and the embedding of the generated output. A statement is marked as hallucinated if this similarity falls below a threshold of 0.5. The hallucination rate is then calculated as follows:

$$\text{Hallucination Rate} = \frac{\text{Number of hallucinated statements}}{\text{Total number of reference statements}}.$$

C Statistical Analysis of Structured Input Effects

To further validate the impact of structured input prompts on financial question answering (QA) performance, we perform paired t-tests comparing zero-shot and structured prompting across three models: Gemma-7B, Qwen2.5-7B, and Meta-LLaMA-3-8B.

Figure 6 presents box plots and mean bar plots summarizing the performance differences between the two prompting strategies. Structured prompting consistently and significantly improves model performance, with p-values below 0.001 in all cases, indicating strong statistical significance.

Table 9: Paired t-test comparing the Baseline (original questions) and Ours (structured questions) on the FiQA and Finance-Alpaca datasets. Mean±SD are reported; ** denotes $p < 0.01$.

Dataset	Model	Baseline (Mean ± SD)	Ours (Mean ± SD)	Δ	t	p
FiQA	Gemma-7B	-8.00 ± 68.41	24.26 ± 57.29	32.26	-4.584	0.000**
	Qwen2.5-7B	8.01 ± 68.38	53.54 ± 45.17	45.53	-7.607	0.000**
	LLaMA3-8B	15.22 ± 51.91	40.00 ± 47.65	24.78	-6.352	0.000**
Finance-Alpaca	Gemma-7B	-29.68 ± 47.13	19.44 ± 61.50	49.12	-7.255	0.000**
	Qwen2.5-7B	15.11 ± 62.63	62.63 ± 43.31	47.52	-8.049	0.000**
	LLaMA3-8B	19.91 ± 51.04	42.70 ± 45.16	22.79	-5.369	0.000**

* $p < 0.05$, ** $p < 0.01$.

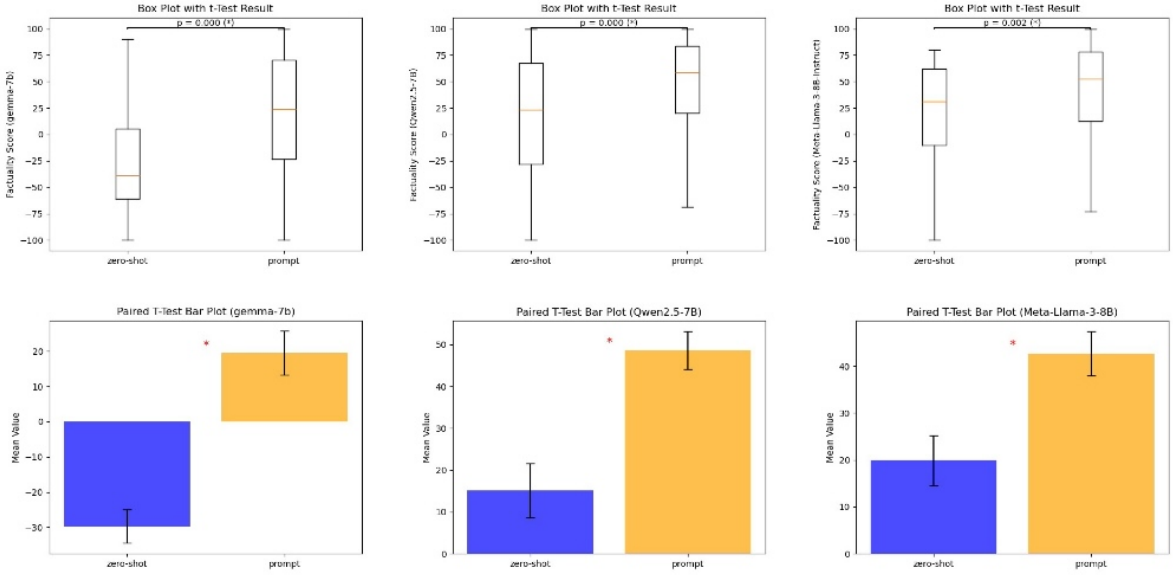

Figure 6: Statistical analysis of the effect of structured input on model performance. The box plots and paired t-test bar plots show significant performance improvements across three models (Gemma-7B, Qwen2.5-7B, Meta-LLaMA-3-8B) when using structured prompting compared to zero-shot prompting. All comparisons show p-values below 0.001, indicating strong statistical significance.

D Prompt Template for Structured Question Decomposition

To guide the decomposition of complex financial queries into hierarchical structures, we have designed a specialized prompt template for structured question modeling. This template directs the model to identify the main scope of the question, extract key aspects, and generate detailed descriptions for each aspect.

Figure 7 shows the full prompt format along with an example decomposition for the question "At what point do index funds become unreliable?". By employing this structured prompting strategy, we ensure that the generated input is consistent and interpretable, which significantly enhances LLMs' ability to comprehend and reason about complex financial queries, leading to improved performance in financial QA tasks.

prompt = """

Please output the knowledge and information related to the question in a structured format.

Statement's scope:

"[generally a noun phrase]'

Statement's main aspects and corresponding descriptions:

"1. [the first aspect of the statement]'

"1.1 [a descriptive sentence] '

"1.3 [another descriptive sentence] '

"2. [the second aspect of the statement] '

"2.1 [a descriptive sentence] '

"3. [another aspect of the statement]'

question:At what point do index funds become unreliable?

###Statement's scope:

Index Funds

###Statement's main aspects and corresponding descriptions:

1. Market Volatility

1.1 Index funds can become unreliable during periods of extreme market volatility. 1.2 Market fluctuations might cause tracking errors, leading to deviations from the index's performance.

2. Market Bubble or Crash

2.1 If a market bubble bursts or there is a crash, the underlying assets in the index fund may suffer significant losses. 2.2 Prolonged market downturns could affect the recovery potential of index funds.

3. Tracking Error

3.1 Index funds may become unreliable if they exhibit significant tracking errors, meaning the fund's performance deviates from the benchmark index. 3.2 Factors like liquidity issues, fund expenses, and rebalancing delays can contribute to tracking errors.

4. Changes in Index Composition

4.1 Significant changes in the composition of an index (e.g., major companies being added or removed) could cause an index fund to become less reliable. 4.2 If the index fund is unable to adjust its portfolio quickly enough, this could impact performance.

5. High Management Fees

5.1 Although index funds typically have low fees, unusually high fees relative to other index funds can make them unreliable in terms of value for money. 5.2 These high fees can erode returns, especially during times of modest market performance.

6. Economic or Political Instability

6.1 Economic or political instability, such as wars or trade conflicts, can create global financial uncertainty, making index funds riskier. 6.2 In such cases, the performance of the sectors or regions that the index tracks may be disproportionately affected.

Please output the knowledge and information related to the question in a structured format.

Statement's scope:

"[generally a noun phrase]'

Statement's main aspects and corresponding descriptions:

"1. [the first aspect of the statement]'

"1.1 [a descriptive sentence] '

"1.3 [another descriptive sentence] '

"2. [the second aspect of the statement] '

"2.1 [a descriptive sentence] '

"3. [another aspect of the statement]'

"""

Figure 7: Prompt template for structured question decomposition. The template helps the model extract the scope, aspects, and detailed descriptions from complex queries, facilitating structured input generation for financial QA tasks.

E Prompt for Generating Must Have and Nice to Have Statements

To generate fine-grained annotations for evaluating financial QA, we designed a structured prompting strategy that guides the model to identify and extract Must Have and Nice to Have statements from the original answer.

Figure 8 presents the complete prompt format used in the annotation process. This includes the specific conditions for identifying each type of statement (Must Have and Nice to Have) and the expected structure of the output in JSON format.

Prompt of generating answer and statements

Please generate `"Must_have Statements"` and `"Nice_to_have Statements"` from the answer based on the following conditions:

Must Have Statements:

These are statements that the model must include to ensure the accuracy of the response (for example, providing a detailed explanation of investment risks or a reasonable analysis of market trends).

Nice to Have Statements:

These are supplementary statements (for example, providing relevant economic data or additional information that could influence investment decisions).

Please note that the `Must_have` statements should include all accurate answers to the question. Only by including these key points will the question be considered fully answered.

`Nice_to_have` statements may include multiple additional explanations that complement the answer.

Do not change the content of the question and answer.

Please return the content in JSON format:

```
json{ "Question": "Your question here", "Answer":
"The answer here", "Must_have": ["Must_have statement
1", "Must_have statement 2" ...], "Nice_to_have":
["Nice_to_have statement 1", "Nice_to_have statement 2"
...] }
```

Figure 8: Prompt template for generating *Must Have* and *Nice to Have* statements from a model-generated answer. This prompt outlines the criteria for each category and requests the results in a structured JSON format for downstream evaluation.

F Discussion of Extended Results

In this appendix, we provide a brief discussion of the extended experimental results presented in Appendix A. While structured question prompts consistently enhance model performance across both the FiQA and Finance-Alpaca datasets, we observe notable differences in the degree of improvement.

First, the performance variance is larger on the Finance-Alpaca dataset compared to FiQA. For instance, Gemma-7B shows a 24.02% reduction in hallucination rate with structured prompts on Finance-Alpaca (from 49.13% to 25.11%), whereas on FiQA, the reduction is smaller at 15.46%. Similar patterns are seen across other models, suggesting that structured input is particularly beneficial when the underlying data distribution is noisier, or when instruction-following quality is initially lower, as often occurs with synthetic datasets like Finance-Alpaca.

Second, models achieve higher comprehensiveness scores on Finance-Alpaca with structured prompts, with improvements reaching up to 21.5% (e.g., Qwen2.5-7B from 35.35% to 61.63%). This indicates that structured prompts help LLMs more effectively extract and organize key information when questions are broad or under-specified.

Overall, these extended results reinforce the notion that structured question modeling is especially valuable when working with complex or less clean financial QA inputs, making it a versatile and robust technique across different domains and model families.

G Experimental Setup

Our replication experiments on the FinEvalQA dataset were performed using FinGPT-v3.2 (LLaMA-2-7B with multi-task LoRA fine-tuning). We evaluate two distinct prompting strategies: (1) a baseline prompt, and (2) a hierarchical query structuring prompt. Detailed prompt templates and inference parameters are as follows.

G.1 Baseline Prompt

We adopt the minimal baseline prompt template as defined by the official FinGPT ChatML format on Hugging Face:

```
[INST] «SYS»
You are FinGPT, a large-scale language model specialised in finance.  Provide a concise,
accurate answer to the user question.  If numeric data are involved, report them with the
correct unit and source.
«/SYS»
### Question
{Original question text}
[/INST]
```

This baseline does not include CoT reasoning or retrieval-based prompting, serving purely as a minimal comparative reference.

G.2 Hierarchical Query Structuring Prompt

The HQS prompt introduces a structured, multi-level query format:

```
[INST] «SYS»
You are FinGPT, a large-scale language model specialised in finance. Answer comprehensively
while strictly avoiding unverifiable claims.
«/SYS»
### Scope
{Primary topic/scenario}
### Aspect
{Secondary sub-topic}
### Description
{Detailed description}
### Instruction
First enumerate every **Must-Have** item (numbered 1, 2, . . . ), then give any **Nice-to-Have**
information. Cite concrete facts where possible; do **not** hallucinate.
[/INST]
```

G.3 Inference Parameters

We maintain identical inference settings across both templates. Key hyperparameters used in the `generate()` function (aligned closely with the official FinGPT Forecaster implementation) are listed in Table 10.

Table 10: Inference hyper-parameters used for all backbone LLMs

Parameter	**FinGPT-v3.2**	**Gemma-7B**	**Qwen 2.5-7B**	**LLaMA-3 8B**
max_new_tokens	512	512	512	512
do_sample	False	False	False	False
temperature	0.2	0.2	0.2	0.2
top_pInactive because `do_sample=False`.	0.9	0.9	0.9	0.9
repetition_penalty	1.1	1.1	1.1	1.1
eos_token_id, pad_token_id		`tokenizer.eos_token_id`		
device_map	"auto"	"auto"	"auto"	"auto"
dtype	float16	float16	float16	float16
context_length	4096	4096	4096	8192