Unlocking the Power of LLMs for Efficiently Automatic Extract Information from Hybrid Long Documents

Anonymous ACL submission

Abstract

Information extraction is a vital task in natural language processing. It involves extracting user-interesting information from natural language and serves many downstream tasks, including knowledge graphs, information retrieval, and question-answering systems. Given LLMs' robust comprehension and reasoning across diverse tasks, their potential for this task is substantial. However, applying LLMs directly for complex documents faces challenges, including handling lengthy documents, understanding tables, adapting to representation ambiguity, and ensuring numerical precision. Given the absence of comprehensive datasets encompassing these challenges, we introduce the **Fi**nancial Reports **N**umerical **E**xtraction (**FINE**) dataset to facilitate further investigation. We present the **Spl**i**t**-**Re**combination **F**ramework (**SiReF**) that effectively counters these challenges with table serialization, embedding retrieval, and precision prompts. Extensive experiment results demonstrate its adaptability across various domains and LLMs with different capabilities. The dataset and code are provided in the attachments.

1 Introduction

Information extraction (IE), which involves extracting and restructuring specific information from natural language texts, is a significant task in natural language processing (Zheng et al., 2023). For example, extracting time and location from news articles (Sedik and Romadhony, 2023); or extracting product names and performance metrics from technical documents (Meuschke et al., 2023). It has extensive applications in various fields such as knowledge graphs (Jaradeh et al., 2023), question-answering systems (Khot et al., 2017), and sentiment analysis (Cheng et al., 2016).

Recently, LLMs have displayed remarkable capabilities in a wide array of tasks, showcasing their potential to process complex textual data (Wei et al., 2023a; Wang et al., 2023b; Zhou et al., 2022; Kojima et al., 2023). Hence, it is important to investigate how to harness the powerful capabilities of LLMs for IE. Currently, only a few tools, such as PDF-GPT (Tripathi, 2023) and ChatPaper (Luo et al., 2023), directly leverage LLMs for IE. However, when applying these methods, they encounter four challenges in handling complex scenarios: 1) The document's length far exceeds the token limit of LLMs, preventing them from processing the entire content. 2) Documents contain tables, and LLMs struggle to directly handle such structured data. 3) The presence of multiple representations for the same concept leads to ambiguity. LLMs fail to extract relevant information when faced with inconsistent keywords. 4) In documents rich in numerical data, the same keyword corresponds to values with varying precision. LLMs can't return the most precise result.

We refer to documents exhibiting these characteristics as Hybrid Long Documents (HLDs). Given the lack of an appropriate dataset encompassing these challenges, we propose the **Fi**nancial Reports **N**umerical **E**xtraction (**FINE**) dataset, derived from real-world and publicly accessible financial reports. This dataset features several characteristics: each document is lengthy with a blend of textual and tabular contents; a high degree of keyword ambiguity; an abundance of numerical information; and stringent quality control measures are employed.

Through comprehensive experimentation, we introduce a **spl**i**t**-**re**combination-based **f**ramework (**SiReF**). By employing a splitting and recombination process, the framework allows LLMs to gradually process the entire document. To address the above challenges: 1) We propose two implementation strategies: Refine and Map-Reduce. The Refine strategy maintains a continuously evolving summary. The Map-Reduce strategy extracts information in parallel and combines it to form a com-

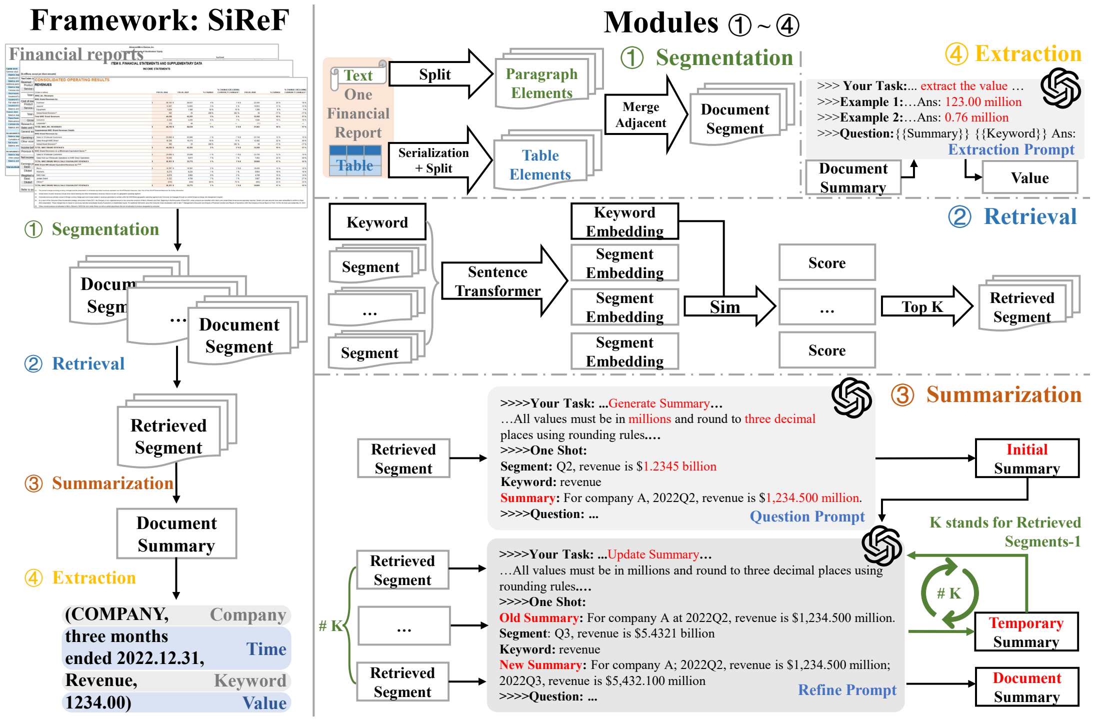

Figure 1: This figure demonstrates the SiReF process using financial reports as an example, with some modules presenting only one implementation. The SiReF framework illustrates the end-to-end IE process, consisting of four modules: Segmentation, dividing lengthy documents into short segments; Retrieval, selecting the most relevant segments related to the given keyword; Summarization, using LLMs to generate a concise summary of relevant information; and Extraction, extracting the keyword-corresponding value from the summary.

plete summary. While the Refine strategy demonstrates superior accuracy, the Map-Reduce strategy exhibits greater efficiency. 2) To enable LLMs to process tables, we introduce table serialization, which converts tables into text format for input. After comparing different serialization methods, we find that LLMs can effectively understand tables without requiring extensive hierarchical information. 3) For the issue of ambiguity, we find that by reducing irrelevant information, LLMs can better adapt to representation ambiguity. Therefore, we introduce an embedding-based retrieval technique. 4) To address the issue of numerical precision, we experiment with prompt engineering by incorporating precision requirements in the task description and showcasing precision preservation within the shots. By integrating both methods, we activate the in-context learning ability, leading to more accurate responses.

Integrating the above technologies, we present an optimal implementation of SiReF. The experimental results demonstrate SiReF's performance across three dimensions: Flexibility across various domains; Adaptability to LLMs with differing capabilities; Proficiency in handling ambiguity in expressions and numerical precision. Our contributions to leveraging LLMs for information extraction from HLDs can be summarized as follows:

1. We construct the **Fi**nancial Reports **N**umerical **E**xtraction (**FINE**) dataset, which is derived from real-world and publicly accessible financial reports.

2. To address the challenges of extracting information from HLDs, we propose the SiReF and give an optimal implementation.

3. We conduct extensive experiments to demonstrate SiReF's adaptability across various scenarios - financial reports, Wikipedia, and scientific papers - revealing the impact of different strategy parameters on SiReF's performance.

2 Framework

2.1 Split-Recombination Based Framework

To enable LLMs to handle HLDs, we propose a **s**pl**i**t-**re**combination based **f**ramework (**SiReF**) that permits LLMs to progressively process the whole document in a step-by-step manner. The SiReF framework consists of four modules: Segmentation, Retrieval, Summarization, and Extraction, as shown in Figure 1. SiReF first splits documents into manageable segments for LLMs, then retrieves the most relevant segments related to the keyword based on embedding similarity, followed by summarizing the retrieved segments to compress and consolidate critical information and finally extracting the keyword-corresponding information from the generated summary. This is a feasible framework, there are many implementations for each module. In the following text, we will introduce each module and provide an optimal implementation based on our exploration of how to address the challenges in HLDs.

2.2 Segmentation

Despite LLMs vastly improving sequence length handling compared to traditional models like text-davinci-003, which can process 4,097 tokens, HLDs often contain even more tokens. To address this challenge, we employ this module to split documents into segments that LLMs can handle. Figure 1 demonstrates this module's three steps: Serialization, Split, and Merge.

Serialization: Serialize tables into text. In hybrid documents, most information is found within tables. However, LLMs are designed for processing text, so we need this module to convert tables into a textual format.

Split: Split long elements. In HLDs, there may be exceptionally long elements, such as large tables and extensive paragraphs, which far exceed the processing capacity of LLMs. To enable LLMs to handle these elements and avoid information loss, we easily divide the overlong paragraphs and tables into small sub-elements.

Merge: Merge small elements as segments. The primary reason for merging is to maintain semantic relationships between adjacent small elements.

2.3 Retrieval

Long documents contain a large number of document segments. Processing all segments would significantly introduce irrelevant information and increase LLM invocations. Therefore, we adopt an embedding-based retrieval strategy (Li et al., 2021) to select the most relevant segments. We retrieve the top-ranked segments with the highest similarity.

2.4 Summarization

The content related to the keyword is often distributed across various segments. To effectively extract and concentrate information, the summarization module leverages LLMs to generate a summary containing relevant information from selected segments. Since LLMs can only process one segment per invocation, a strategy is needed to connect different segments effectively. We implement two summarization strategies: Refine Strategy and Map-Reduce Strategy.

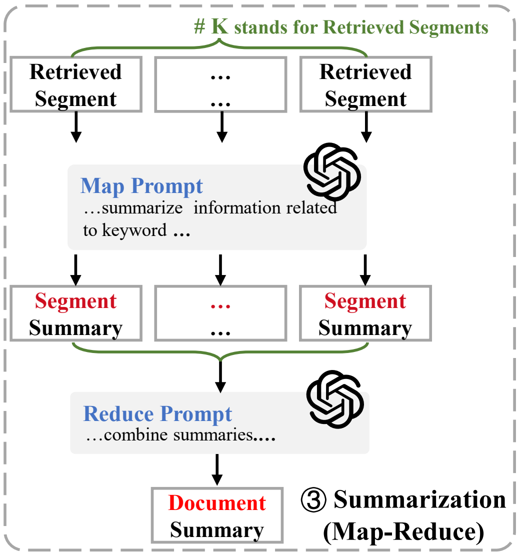

Figure 2: Illustration of the Map-Reduce Strategy, comprising two stages: Map, generating individual segment summaries, and Reduce, combining these summaries to form a single document summary.

The Refine Strategy comprises two main steps, depicted in the Summarization module of Figure 1. First, the *Question prompt* generates an initial summary from the first segment, guiding LLMs to extract relevant information. Next, the *Refine prompt* updates the summary by incorporating information from the remaining segments.

The Map-Reduce Strategy aims to combine summaries from document segments, comprising two stages: Map and Reduce (as illustrated in Figure 2). In the Map stage, LLMs generate a segment summary for each document segment in parallel. During the Reduce stage, LLMs consolidate all the segment summaries to form a cohesive document summary.

2.5 Extraction

After the summarization, we obtain a summary that contains the keyword's value along with some auxiliary information. To remove auxiliary information and facilitate downstream tasks, it becomes essential to extract the numerical value.

As shown in the Extraction module of Figure 1, LLMs are utilized to extract the value from the summary. By leveraging the *Extraction Prompt*, LLMs can accurately achieve this goal.

2.6 Numerical Precision Enhancement

In scenarios with more numerical data, we find that LLMs have difficulties in maintaining numerical precision. For example, the same keyword could correspond to values with different precision levels, all being correct, but the LLMs might not return the most precise result or even a wrong answer. However, in scenarios such as financial analysis, precision is essential for the downstream tasks. To tackle this issue, we incorporate two methods from the aspect of the prompt: task description and input-output case. In the task description, we give the requirement of precision. In the input-output case, we provide an example of how to manage precision.

2.7 Keyword Completion

Incomplete keywords provided by users can lead to inaccurate IE. For example, users might inquire about *Revenue*, but in financial reports, the same keyword might correspond to multiple entities (such as different subsidiaries or time periods). To address this issue, we introduce a keyword completion method. In our implementation, we utilize the document's metadata. According to our analysis (as discussed in subsection 6.3), providing more contextual information can greatly improve the accuracy of SiReF.

3 Experiment Setting

3.1 Datasets of Three Domains

Dataset	FINE	WIKIR	MPP
Max # tokens	234,900	58,512	123,105
Min # tokens	13,022	13,548	3,672
Avg. # tokens	59,464.3	30,922.1	17,553.05

Table 1: Statistics for FINE, WIKIR, and MPP datasets.

To assess SiReF's capacity to comprehend HLDs and support future research, we conduct experiments in three representative domains: financial reports, Wikipedia, and scientific papers. We construct a dataset for each domain. The basic statistics can be found in Table 10. Among these datasets, the financial dataset is used to analyze the various settings of SiReF. The overall performance is tested on all datasets. For more details about these three datasets, please refer to Appendix A.

In the financial reports domain, we introduce a new dataset called the **Fi**nancial Reports **N**umerical **E**xtraction (**FINE**), comprising manually extracted KPIs from SEC's EDGAR[1]. Using the financial report as content, financial KPIs and related values are utilized as (key, value) pairs.

In the Wikipedia domain, we select the Wikireading-Recycled (WIKIR) dataset (Dwojak et al., 2020). A Wikipedia page serves as the content, while the corresponding key and value are extracted from Wikidata.

In the scientific papers domain, we select the MPP (Massive Paper Processing) dataset (Polak et al., 2023). A scientific paper serves as the content, with chemical materials as the keys and their corresponding cooling rates as the values.

3.2 Evaluation Metrics

For the FINE, we use the Relative Error Tolerance Accuracy (RETA) metric, for the two other datasets, we use the Accuracy (Acc) metric.

In FINE, all ground truth values are presented in millions, rounded to two decimal places. However, in original financial reports, the numerical precision is not uniform, as the values can be expressed in different units, such as millions or billions. This leads to the same keyword being associated with multiple values of varying precision, making it difficult to evaluate the accuracy of predictions.

To address this issue, we use the Relative Error Tolerance Accuracy (RETA) metric, which considers predictions as correct if their relative error falls within a specified tolerance threshold (e.g., RETA X% means predictions with a relative error of no more than X% are considered correct). By setting different RETA levels, we can assess the model's performance according to various practical requirements and gain a comprehensive understanding of its capabilities in IE from financial reports.

This issue does not exist in the WIKIR and MPP datasets. In the WIKIR dataset, the ground truth is represented as a string, whereas in the MPP dataset, it is a floating-point number with no alternative precision representation.

3.3 Model and Parameter Settings

In our experiments, we take the GPT-3.5 (text-davinci-003) as our primary subject for analysis. All experimental results are the average of three trials. Based on GPT-3.5, the detailed parameter configurations in SiReF are as follows.

Token Allocation: We allocate tokens to accommodate the model's maximum sequence length and

[1] https://www.sec.gov/edgar/

Alloc.	# Token
Max Seq. Length	4,097
Doc. Elem.	$\leq$ 2,000
Doc. Seg.	$\leq$ 2,500
Keyword	$\leq$ 50
Summary	$\leq$ 500

Table 2: Token allocation

the requirements of each SiReF module. The token allocations are presented in the Table 2. **Embedding Model:** We use the sentence-transformers/all-mpnet-base-v2[2] model for computing embeddings. This model can handle a sequence length of 384 tokens. **Prompts:** In SiReF, there are many different types of prompts serving various SiReF modules: question prompts, refine prompts, extraction prompts, and so on. Appendix F shows the details of the prompts.

3.4 Researrch Questions

In this paper, we are trying to answer the following research questions:

RQ1: How about the effectiveness of SiReF?

RQ2: How to enhance SiReF's sensitivity to numerical precision?

RQ3: How do different strategies influence SiReF?

RQ4: How do various parameters affect SiReF?

4 RQ1: How about the effectiveness of SiReF?

To evaluate the effectiveness of SiReF, we conduct experiments from three dimensions: adaptability in different domains, adaptability to LLMs with varying capabilities, and adaptability to representation ambiguity. In these experiments, we compare SiReF with the naive method on all three datasets. The SiReF used in these experiments uses the optimal implementation for each module based on our findings. The naive method directly uses LLMs adopted to HLDs.

4.1 Adaptability in Different Domains

To demonstrate adaptability across various domains, we conduct a comparison on three different datasets. The Figure 3 displays the experimental results on FINE. It shows the accuracy at different RETA levels, ranging from 1% to 10%, and the average accuracy across all RETA settings. The Figure 4 displays the experimental results on WIKIR and MPP. It shows the average accuracy.

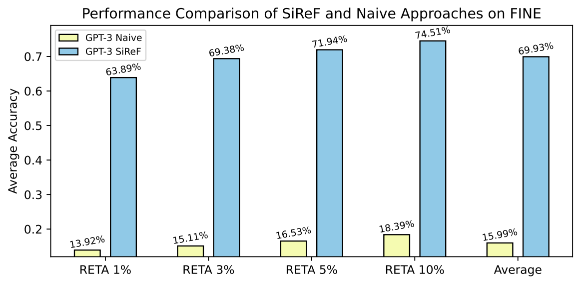

Figure 3: Comparison of the SiReF and Naive method at different RETA levels on FINE.

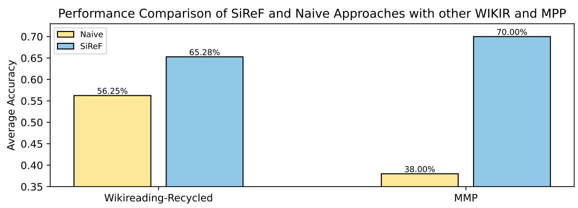

Figure 4: Comparison of the SiReF and Naive method on WIKIR and MPP.

The experimental results demonstrate that the SiReF method outperforms the naive method in all three datasets. The improvement in average accuracy indicates that the SiReF method is more effective in extracting relevant information from various HLDs. This demonstrates the SiReF's adaptability in different domains.

In Figure 3, as the RETA becomes more stringent, we can also find the performance gap between the naive method and SiReF becomes larger. This indicates that SiReF is capable of delivering more accurate results under stricter evaluation metrics.

4.2 Adaptability for LLMs with Different Capabilities

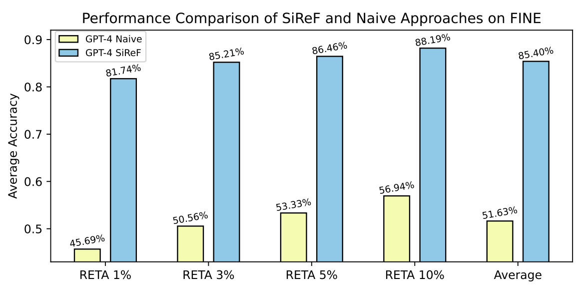

Figure 5: Comparison of the SiReF and Naive method at different RETA levels on GPT-4.

To investigate the adaptability of SiReF for LLMs with different capabilities, we also conduct experiments on GPT-4. For the reason that GPT-4

[2]https://www.sbert.net/docs/pretrained_models.html

is currently the most outstanding LLM in terms of comprehensive capabilities (OpenAI, 2023). GPT-4 can handle sequences with a maximum length of 32,768 tokens, while the average length of each sample on WIKIR and MPP datasets does not exceed 32,768 tokens. Compared to GPT-4, WIKIR and MPP are not **long** documents. Therefore, we chose the FINE as the experimental dataset.

The Figure 5 displays the experimental results of SiReF on GPT-4. From the results, we can see that when using GPT-4, SiReF's performance is still better than the naive strategy under different RETA levels. This demonstrates SiReF's adaptability to LLMs with different capabilities.

4.3 Adaptability to Representation Ambiguity

In HLDs, the same concept may have multiple representations, which requires SiReF to have the ability to handle ambiguity. To evaluate whether SiReF can enhance such ability, we conduct a comparison on two sets of keywords: (*Revenue* vs. *Total Net Sales*) and (*Total Equity* vs. *Total Stockholders' Equity*). We compare the Relative Percentage Difference (RPD) in average accuracy between the naive method and SiReF across various RETA levels. The RPD at a certain RETA level is calculated using the following formula:

$$RPD_{X-Y} = \frac{abs(Acc_X - Acc_Y)}{average(Acc_X, Acc_Y)}$$

where Acc_X and Acc_Y represent the average accuracy of two different keywords.

The experimental results are presented in Figure 6. From the results, we observe that SiReF outperforms the naive method across all RETA levels when handling keyword ambiguity. Specifically, comparing *Revenue* vs. *Total Net Sales*, SiReF shows a 22.52% lower avg. RPD than the naive method. Similarly, for *Total Equity* vs. *Total Stockholders' Equity*, SiReF yields a 37.94% lower avg. RPD than the naive method. For more detailed results, please refer to the Appendix B.

5 RQ2: How to enhance SiReF's sensitivity to numerical precision?

To enable SiReF to extract more precise numerical values, we design various numerical precision enhancement methods in the prompt. To assess the performance of these methods, we conducted a comparative experiment under finer RETA levels.

TD-O: Task description only. **TD-R**: TD-O prompt with precision requirements. **TD-S**: TD-O prompt with input-output example. **TD-RS**: TD-O prompt, precision requirements, and input-output example. **TD-SP**: TD-O prompt with precision-inclusive input-output example. **TD-RSP**: TD-O prompt, precision requirements, and precision-inclusive input-output example. See subsection F.4 for details of these prompts.

	RETA				
	0%	**0.001%**	**0.01%**	**0.1%**	**Average**
TD-O	0.4917	0.4937	0.5187	0.5750	0.5198
TD-R	0.3479	0.3479	0.3597	0.4083	0.3660
TD-S	0.4111	0.4153	0.4493	0.5438	0.4549
TD-RS	0.4403	0.4438	0.4722	0.5396	0.4740
TD-SP	0.5278	0.5299	0.5479	0.5882	0.5484
TD-RSP	**0.5646**	**0.5660**	**0.5750**	**0.5938**	**0.5748**

Table 3: Accuracy comparison for different methods aimed at enhancing numerical precision.

From Table 3, we observe the following: **1)** The TD-RSP strategy achieves the highest accuracy across all fine-grained RETA levels, indicating its effectiveness in enhancing the numerical precision of extracted values. **2)** The performance of TD-R, TD-S, and TD-RS strategies is inferior to that of TD-O. This may suggest that improperly designed or insufficient precision prompts could act as a distractor, hindering its ability to focus on improving numerical accuracy.

6 RQ3: How do different strategies influence SiReF?

To determine the most effective strategies for achieving SiReF, we systematically evaluate different approaches related to summarization, table serialization, and keyword completion.

6.1 Analysis of Summarization Strategies

To extract information from multiple retrieved segments, we introduce two strategies: the Refine Strategy and the Map-Reduce Strategy. We conducted a comparative experiment to investigate their respective strengths and weaknesses.

As shown in Table 4, the Refine Strategy consistently outperforms the Map-Reduce Strategy in terms of accuracy across all RETA levels. However, it is essential to consider the trade-off between accuracy and efficiency when selecting a summarization strategy for a given application. The Map-Reduce Strategy offers the advantage of parallel processing, making it a better choice for situations where processing speed is of higher importance.

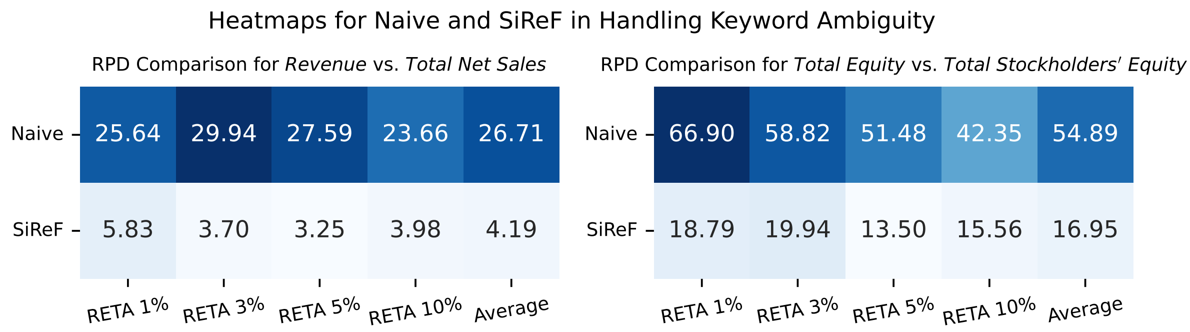

Figure 6: Exploring the Capability to Handle Keyword Ambiguity: Comparison of Naive and SiReF on RPD

	RETA 1%	RETA 3%	RETA 5%	RETA 10%	Average	Time (s\sample)
Map-Reduce	0.5375	0.5729	0.5958	0.6299	0.5840	**13.34**
Refine	**0.6389**	**0.6938**	**0.7194**	**0.7451**	**0.6993**	16.36

Table 4: Comparison between Map-Reduce and Refine strategies across various RETA levels.

6.2 Analysis of Table Serialization Formats

	RETA 1%	RETA 3%	RETA 5%	RETA 10%	Average
PLAIN	**0.6389**	**0.6938**	**0.7194**	**0.7451**	**0.6993**
CSV	0.6264	0.6889	0.7132	0.7361	0.6911
XML	0.3951	0.4507	0.4729	0.5069	0.4564
HTML	0.4542	0.5000	0.5208	0.5590	0.5085

Table 5: Accuracy comparison among PLAIN, CSV, XML, and HTML table serialization formats.

To enable LLMs to handle tabular data, we need to use a specific serialization method to represent tables as text. There are four common serialization methods: PLAIN, CSV, XML, and HTML.

PLAIN serialization extracts text from table cells, separating adjacent cell content with spaces and using newline characters to separate rows. **CSV** serialization separates adjacent cells with comma delimiters. **XML** and **HTML** serialization formats utilize tags[3] to preserve the hierarchical relationships between table elements.

Despite XML and HTML formats retaining hierarchical information, the incorporation of tags results in a higher token count, potentially exceeding the LLMs' maximum sequence length and requiring more frequent table splitting. As shown in Table 5, the PLAIN and the CSV formats outperform the XML and HTML formats in terms of accuracy, likely due to their concise table representation, which reduces table fragmentation and captures the complete semantic information of the tables.

[3] XML employs tags such as <table>, <row>, and <cell>, while HTML utilizes tags like <tr> (for table rows) and <td> (for table cells).

6.3 Analysis of Keyword Completion

	RETA 1%	RETA 3%	RETA 5%	RETA 10%	Average
K	0.3403	0.3917	0.4076	0.4292	0.3922
K_C	0.4681	0.5167	0.5361	0.5604	0.5203
K_T	0.4785	0.5396	0.5500	0.5736	0.5354
K_T_C	**0.6389**	**0.6938**	**0.7194**	**0.7451**	**0.6993**

Table 6: Accuracy comparison for different keyword completion settings across various RETA levels.

To analyze the effectiveness of keyword completion in improving SiReF's performance, we experimented with various settings.

K: Only provide keyword names, such as "Net Income", "Revenue", etc. **K_C**: Provide keyword names and company names, such as "Net Income of Nike". **K_T**: Provide keyword names and time, such as "Net Income of 2022Q4". **K_T_C**: Provide keyword names, time, and company names, such as "Net Income of Nvidia 2022Q4".

As shown in Table 6, we find that the performance of *K_C*, *K_T*, and *K_T_C* strategies is better than that of *K*, with *K_T_C* achieving the best results. This indicates that keyword completion is useful in improving SiReF's accuracy. By providing more meta-data, the model can better understand the context and generate more accurate responses, leading to an overall improvement in performance.

7 RQ4: How do various parameters affect SiReF?

7.1 Analysis of Retrieved Segment Number

In this section, we investigate the effect of the number of retrieved segments on the performance of

	RETA 1%	RETA 3%	RETA 5%	RETA 10%	Average
R@1	0.4757	0.5278	0.5444	0.5694	0.5293
R@2	0.6188	0.6736	0.6931	0.7118	0.6743
R@3	**0.6389**	**0.6938**	**0.7194**	**0.7451**	**0.6993**
R@5	0.6160	0.6799	0.7062	0.7306	0.6832
R@7	0.5917	0.6521	0.6722	0.7090	0.6563
No R	0.3757	0.4986	0.5201	0.5514	0.4865

Table 7: Accuracy comparison for different retrieval quantities (R@n) across various RETA levels.

our framework. Table 7 shows the accuracy for different retrieval quantities, where R@n represents the number of top-ranked segments retrieved.

The results reveal that the highest accuracy across all RETA levels is achieved when the retrieval quantity is set to 3 (R@3). Analyzing the trend, we can observe that the accuracy increases as the retrieval quantity goes from 1 to 3, demonstrating the benefits of retrieving more segments to capture additional information. However, as the retrieval quantity increases beyond 3, the accuracy declines. This suggests that including too many segments may introduce noise or irrelevant information, which adversely affects performance.

7.2 Analysis of Shot Number

	RETA 1%	RETA 3%	RETA 5%	RETA 10%	Average
0-shot	0.4799	0.5229	0.5354	0.5472	0.5214
1-shot	**0.6389**	**0.6938**	**0.7194**	**0.7451**	**0.6993**
2-shot	0.6227	0.6803	0.6966	0.7231	0.6807
3-shot	0.6181	0.6806	0.7007	0.7174	0.6792

Table 8: Accuracy comparison for different numbers of shots across various RETA levels.

In-context learning is important for LLMs. To investigate the impact of the number of shots on SiReF, we experimented with different numbers of shots, ranging from 0 to 3.

As shown in Table 8, the 1-shot setting achieves the highest accuracy across all RETA levels. The performance of 2-shot and 3-shot settings is slightly lower than that of the 1-shot setting but still better than the 0-shot setting. This indicates that a single well-designed example can effectively guide SiReF to generate more accurate responses. However, the slight decrease in performance with additional examples could be attributed to the increased complexity of the input or potential inconsistencies among multiple examples, which may confuse the model rather than provide more guidance.

Based on this experiment, we recommend carefully determining the number of shots when using SiReF for IE. Although providing more shots may still be helpful, it is essential to ensure their consistency and relevance to avoid potential confusion and maintain optimal performance.

8 Discussion

In addition to our extensive exploration experiments with SiReF, we also eliminate the concern of whether pre-trained data affects the experiment results Appendix C, ensuring the reliability of our results. We also analyze computational costs Appendix D. Furthermore, we demonstrated through experiments that it is essential to use both tabular and textual data simultaneously in HLDs Appendix E.

9 Related Work

In our research, we primarily focus on leveraging the capabilities of LLMs across three distinct tasks. 1) Long document processing, helping LLMs exceed their maximum input length limit (Liang et al., 2023). 2) IE, particularly value extraction, where LLMs have shown proficiency in the domains such as IE (Li et al., 2023; Wei et al., 2023b), which includes NER (Gupta et al., 2021; Wang et al., 2023a), Relation Extraction (RE) (Wan et al., 2023; Xu et al., 2023), and Knowledge Graph Extraction (Shi et al., 2023). (Polak et al., 2023; Arora et al., 2023) have successfully demonstrated the extraction of key-value pairs from the text content of academic papers and HTML respectively, thereby substantiating the dependability of LLMs for value extraction. 3) Tabular reasoning, where LLMs have demonstrated considerable ability to perform intricate reasoning tasks with structured data (Chen, 2023; Ye et al., 2023).

10 Conclusion

To assess LLMs' ability to address challenges in information extraction from HLDs, such as handling lengthy documents, understanding tables, adapting to representation ambiguity, and ensuring numerical precision, we construct a dataset from publicly available financial reports, called FINE. We also propose a framework, SiReF, which effectively tackles these challenges through table serialization, embedding-based retrieval, and precision-enhancing prompts. SiReF demonstrates adaptability across various domains and LLMs with different capabilities. Furthermore, we provide a comprehensive analysis of different strategies and parameters of SiReF.

11 Limitations

Despite the substantial enhancement achieved by LLMs through the utilization of SiReF, certain limitations persist.

1. Model ability limitation: This work effectively demonstrates LLMs' ability to extract information from HLDs. However, further evaluation of their capabilities in other aspects, such as formula inferencing, generating abstracts, and keyword extraction, remains necessary.

2. Multimodal limitations: SiReF can effectively extract information from documents containing a mix of textual and tabular data. However, its capabilities in handling other types of content within documents, such as images, diagrams, or complex visualizations, have not been evaluated. In many real-world scenarios, HLDs may contain rich multi-modal information that could be crucial for making informed decisions.

3. Cost constraints: The GPT-3.5 and GPT-4 used in the experiments incur computational costs. For some practical applications, SiReF may not be the most cost-effective method.

References

Simran Arora, Brandon Yang, Sabri Eyuboglu, Avanika Narayan, Andrew Hojel, Immanuel Trummer, and Christopher Ré. 2023. Language models enable simple systems for generating structured views of heterogeneous data lakes. *arXiv preprint arXiv:2304.09433*.

Wenhu Chen. 2023. Large language models are few(1)-shot table reasoners.

Charibeth Cheng, Bernadyn Cagampan, and Christine Diane Lim. 2016. Organizing news articles and editorials through information extraction and sentiment analysis. In *PACIS*, page 258.

Tomasz Dwojak, Michal Pietruszka, Lukasz Borchmann, Jakub Chledowski, and Filip Gralinski. 2020. From dataset recycling to multi-property extraction and beyond. *CoRR*, abs/2011.03228.

Himanshu Gupta, Shreyas Verma, Tarun Kumar, Swaroop Mishra, Tamanna Agrawal, Amogh Badugu, and Himanshu Sharad Bhatt. 2021. Context-ner: Contextual phrase generation at scale. *arXiv preprint arXiv:2109.08079*.

Daniel Hewlett, Alexandre Lacoste, Llion Jones, Illia Polosukhin, Andrew Fandrianto, Jay Han, Matthew Kelcey, and David Berthelot. 2016. Wikireading: A novel large-scale language understanding task over wikipedia. *CoRR*, abs/1608.03542.

Mohamad Yaser Jaradeh, Kuldeep Singh, Markus Stocker, Andreas Both, and Sören Auer. 2023. Information extraction pipelines for knowledge graphs. *Knowl. Inf. Syst.*, 65(5):1989–2016.

Tushar Khot, Ashish Sabharwal, and Peter Clark. 2017. Answering complex questions using open information extraction. In *ACL (2)*, pages 311–316. Association for Computational Linguistics.

Takeshi Kojima, Shixiang Shane Gu, Machel Reid, Yutaka Matsuo, and Yusuke Iwasawa. 2023. Large language models are zero-shot reasoners.

Bo Li, Gexiang Fang, Yang Yang, Quansen Wang, Wei Ye, Wen Zhao, and Shikun Zhang. 2023. Evaluating chatgpt's information extraction capabilities: An assessment of performance, explainability, calibration, and faithfulness. *arXiv preprint arXiv:2304.11633*.

Sen Li, Fuyu Lv, Taiwei Jin, Guli Lin, Keping Yang, Xiaoyi Zeng, Xiao-Ming Wu, and Qianli Ma. 2021. Embedding-based product retrieval in taobao search. In *Proceedings of the 27th ACM SIGKDD Conference on Knowledge Discovery & Data Mining*, pages 3181–3189.

Xinnian Liang, Bing Wang, Hui Huang, Shuangzhi Wu, Peihao Wu, Lu Lu, Zejun Ma, and Zhoujun Li. 2023. Unleashing infinite-length input capacity for large-scale language models with self-controlled memory system. *arXiv preprint arXiv:2304.13343*.

Yongle Luo, Rongsheng Wang, Jiaxi Cui Peter Gam, circlestarzero, Shiwen Ni, Jaseon Quanta, Qingxu Fu, and Siyuan Hou. 2023. Chatpaper: Use llm to summarize papers. `https://github.com/kaixindelele/ChatPaper`.

Norman Meuschke, Apurva Jagdale, Timo Spinde, Jelena Mitrovic, and Bela Gipp. 2023. A benchmark of PDF information extraction tools using a multi-task and multi-domain evaluation framework for academic documents. In *iConference (2)*, volume 13972 of *Lecture Notes in Computer Science*, pages 383–405. Springer.

OpenAI. 2023. GPT-4 technical report. *CoRR*, abs/2303.08774.

Maciej P Polak, Shrey Modi, Anna Latosinska, Jinming Zhang, Ching-Wen Wang, Shanonan Wang, Ayan Deep Hazra, and Dane Morgan. 2023. Flexible, model-agnostic method for materials data extraction from text using general purpose language models. *arXiv preprint arXiv:2302.04914*.

Roy Rachman Sedik and Ade Romadhony. 2023. Information extraction from indonesian crime news with named entity recognition. In *KST*, pages 1–5. IEEE.

Yucheng Shi, Hehuan Ma, Wenliang Zhong, Gengchen Mai, Xiang Li, Tianming Liu, and Junzhou Huang. 2023. Chatgraph: Interpretable text classification by converting chatgpt knowledge to graphs. *arXiv preprint arXiv:2305.03513*.

Bhaskar Tripathi. 2023. Pdf-gpt. https://github.com/bhaskatripathi/pdfGPT.

Zhen Wan, Fei Cheng, Zhuoyuan Mao, Qianying Liu, Haiyue Song, Jiwei Li, and Sadao Kurohashi. 2023. Gpt-re: In-context learning for relation extraction using large language models. *arXiv preprint arXiv:2305.02105*.

Shuhe Wang, Xiaofei Sun, Xiaoya Li, Rongbin Ouyang, Fei Wu, Tianwei Zhang, Jiwei Li, and Guoyin Wang. 2023a. Gpt-ner: Named entity recognition via large language models. *arXiv preprint arXiv:2304.10428*.

Xuezhi Wang, Jason Wei, Dale Schuurmans, Quoc Le, Ed Chi, Sharan Narang, Aakanksha Chowdhery, and Denny Zhou. 2023b. Self-consistency improves chain of thought reasoning in language models.

Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, Brian Ichter, Fei Xia, Ed Chi, Quoc Le, and Denny Zhou. 2023a. Chain-of-thought prompting elicits reasoning in large language models.

Xiang Wei, Xingyu Cui, Ning Cheng, Xiaobin Wang, Xin Zhang, Shen Huang, Pengjun Xie, Jinan Xu, Yufeng Chen, Meishan Zhang, et al. 2023b. Zero-shot information extraction via chatting with chatgpt. *arXiv preprint arXiv:2302.10205*.

Xin Xu, Yuqi Zhu, Xiaohan Wang, and Ningyu Zhang. 2023. How to unleash the power of large language models for few-shot relation extraction? *arXiv preprint arXiv:2305.01555*.

Yunhu Ye, Binyuan Hui, Min Yang, Binhua Li, Fei Huang, and Yongbin Li. 2023. Large language models are versatile decomposers: Decompose evidence and questions for table-based reasoning. *arXiv preprint arXiv:2301.13808*.

Hanwen Zheng, Sijia Wang, and Lifu Huang. 2023. A survey of document-level information extraction. *CoRR*, abs/2309.13249.

Denny Zhou, Nathanael Schärli, Le Hou, Jason Wei, Nathan Scales, Xuezhi Wang, Dale Schuurmans, Olivier Bousquet, Quoc Le, and Ed Chi. 2022. Least-to-most prompting enables complex reasoning in large language models. *arXiv preprint arXiv:2205.10625*.

A Details of Datasets

A.1 FINE

To the best of our knowledge, there is no suitable HLD dataset in the domain of financial reports. So we introduce the **Fi**nancial Reports **N**umerical **E**xtraction (**FINE**) dataset, comprising manually extracted KPIs from SEC's EDGAR[4]. We collect reports from 18 companies across four sectors for a 4-year fiscal period (2019-2022). Within a fiscal year, a company's financial reports consist of three quarterly and one annual financial report. These companies are categorized into four groups based on their operational domains: technology, retail, financial services, and food and beverage. We identify 9 commonly used crucial KPIs that exemplify financial reports' ambiguous, HLDs characteristics of financial reports. In FINE, ground truth is represented as tuples of four elements: (company, time, keyword, value)[5]. These values are expressed in millions and rounded to two decimal places using conventional rounding techniques, providing the most prevalent and precise representation in financial reports. We manually identified pertinent keywords and extracted values while training several individuals to assemble this dataset, ensuring each data point was labelled by four people to minimize labelling errors.

In selecting benchmark keywords, we prioritize their significance within financial reports. We performed an intersection analysis on the essential keywords presented on two statistical websites publicly available from reputable organizations, MSN Money[6] and Google Finance[7], which showcase varying subsets of KPIs. We applied filtering criteria: keywords must exhibit ambiguity, be distributed throughout HLDs, and have values directly extractable from financial reports. We identified a

Consolidated Condensed Statements of Income

(In Millions, Except Per Share Amounts; Unaudited)	Three Months Ended		Nine Months Ended	
	Oct 1, 2022	Sep25, 2021	Oct 1, 2022	Sep25, 2021
Net revenue	$ 15,338	$ 19,192	$ 49,012	$ 58,496
Cost of sales	8,803	8,446	27,646	25,690
Gross margin	6,535	10,746	21,366	32,806
Research and development	4,302	3,803	13,064	11,141
Marketing, general and administrative	1,744	1,674	5,296	4,601
Restructuring and other charges	664	42	(460)	2,597
Operating expenses	6,710	5,519	17,900	18,339
Operating income (loss)	(175)	5,227	3,466	14,467
Gains (losses) on equity investments, net	(151)	1,707	4,082	2,370
Interest and other, net	138	(76)	1,016	(328)
Income (loss) before taxes	(188)	6,858	8,564	16,509
Provision for (benefit from) taxes	(1,207)	35	(114)	1,264
Net income	$ 1,019	$ 6,823	$ 8,678	$ 15,245
Earnings per share—basic	$ 0.25	$ 1.68	$ 2.11	$ 3.76
Earnings per share—diluted	$ 0.25	$ 1.67	$ 2.10	$ 3.73
Weighted average shares of common stock outstanding:				
Basic	4,118	4,061	4,104	4,055
Diluted	4,125	4,086	4,123	4089

Figure 7: Income statement of Intel in 2022-10-01 Quarterly report.

[4] https://www.sec.gov/edgar/

[5] One tuple denotes the value corresponding to a specific keyword for a given company at a specified time. For example, (COMPANY, three months ended 2022.12.31, Revenue, 12345.00) indicates that COMPANY's Revenue for the three months ending on December 31, 2022, is $ 12,345.00 million.

[6] https://www.msn.com/en-us/ money

[7] https://www.google.com/finance/

final set of 9 keywords (as presented in Table 9) for further evaluation. Figure 9 displays the token count distribution in FINE, with the largest document containing 234,900 tokens, the smallest document comprising 13,022 tokens, and an average of 59,464 tokens per document. Table 10 illustrates the specific representation of *Revenue* in various companies' financial reports. In FINE, we systematically document ambiguous expressions of all keywords across various companies.

Q3 2022 vs. Q3 2021

Our Q3 2022 revenue was $15.3 billion, down $3.9 billion or 20% from Q3 2021. CCG revenue decreased 17% from Q3 2021 due to lower Notebook volume in the consumer and education market segments, though Notebook ASPs increased due to a resulting change in product mix. CCG also had lower revenue due to the continued ramp down from the exit of our 5G smartphone modem business. DCAI revenue decreased 27% from Q3 2021. Server volume decreased, led by enterprise customers, and due to customers tempering purchases to reduce existing inventories in a softening datacenter market. Server ASPs decreased due to a higher mix of revenue from hyperscale customers within a competitive environment. NEX revenue increased 14% from Q3 2021, primarily due to increased demand for 5G products, higher Ethernet demand and ASPs, and accelerated demand for Edge products, partially offset by decreased demand for Network Xeon. The decrease in "all other" revenue reflects revenue of $1.1 billion in Q3 2021 related to the divested NAND memory business for which historical results are recorded in “all other.”

Figure 8: A text description of Intel in 2022-10-01 Quarterly report.

A.2 Wikipedia

For this type of data, we chose the Wikireading-Recycled dataset (Dwojak et al., 2020). This dataset is an improved version of the Wikireading dataset (Hewlett et al., 2016), which includes a human-annotated test set. In this dataset, a Wikipedia page serves as the content, while the corresponding key and value are extracted from Wikidata. For example, from the *Wikipedia of "In Search of Lost Time"* (Content), we can know that the *main subject* (Key) of this novel is *memory* (Value). From the human-annotated test set, we filtered out short samples with less than 10,000 tokens and those that would trigger safety restrictions in the text-davinci-003 model. After filtering, a total of 72 test samples remained for our evaluation.

For the Wikireading-Recycled dataset, the ground truth is in text form, and the predictions generated by LLMs often do not match the ground truth in terms of phrasing, despite conveying the same meaning. To evaluate the accuracy of LLM predictions, we combined the assessments of four human judgments and GPT-4's judgments. We then calculated the average of these evaluation results to determine the final metric.

A.3 Scientific Papers

For this type of data, we selected the MPP (Massive Paper Processing) dataset (Polak et al., 2023). In this dataset, scientific papers serve as the content, with chemical materials as the keys and their corresponding cooling rates as the values. For example, from a paper "... *the composition of* $Al_{87}Ni_9Ce_4$ *has the maximum cooling rate of nearly 1.02* × $104K/s$..." (Content), we can know that the *cooling rate* (Key) of $Al_{87}Ni_9Ce_4$ is *1.02* × $104K/s$ (Value). We filtered out short papers and samples containing multiple values for the same key. Ultimately, 50 test samples remained for evaluation.

For the MPP dataset, the ground truth is numeric. This numeric value only appears in a unique form throughout the text. Therefore, we only needed to determine whether LLMs' predictions were consistent with the ground truth.

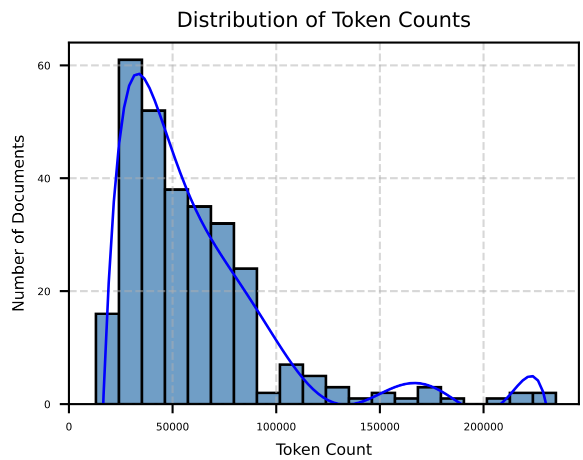

Figure 9: Histogram of token counts in financial documents.

Category	Keywords
Income Statement	*Revenue* *Operating Expense* *Net Income* *Earnings Per Share*
Balance Sheet	*Total Assets* *Total Equity*
Cash Flow	*Operating Activities* *Investing Activities* *Financing Activities*

Table 9: Nine Keywords in FINE.

Corporation	*Revenue*
Amazon	*Total Net Sales*; *Net Sales [span] Consolidated*; *Consolidated [span] Net Sales*;
AMD	*Total net revenue*; *Net revenue*; *Total sales to external customers*;
Apple	*Total Net Sales*
Autoliv	*Consolidated net sales*; *Net Sales*; *Total Net Sales*
BOEING	*Revenues*; *Total revenues*
Cisco	*Total revenue*; *Product revenue: [span] total*; *Revenue: [span] total Revenue*
Coca Cola	*Net operating revenues*
Dell	*Total net revenue*; *Total consolidated net revenue*; *Net revenue*
eBay	*Net revenues*; *Total net revenues*
Intel	*Net revenue*; *Total net revenue*
Meta Platforms	*Revenue*; *Total revenue*
Microsoft	*Revenue*; *Total revenue*
Nike	*Revenues*; *TOTAL NIKE, INC. REVENUES*; *Total revenue*; *Revenue*
Nvida	*Total revenues*
Oracle	*Total net revenues*
Starbucks	*Total revenue*
State Street	*Total revenues*
Walmart	*Total net revenues*; *Total revenue*

Table 10: The appearance of *Revenue* in various company financial reports. We record the different occurrences of the selected keywords in FINE. [span] means that there are merged cells and indented forms in the table.

B Detailed Results of Keyword Ambiguity Experiment

In this section, we present the detailed experimental results for both the naive method and SiReF in handling keyword ambiguity. The results are shown for different RETA levels, as well as the average RPD for each comparison.

Table 11 shows the experimental results for the naive method at different RETA levels. The results include comparisons between Revenue and Total Net Sales, as well as Total equity and Total stockholders' equity. Table 12 displays the experimental results for SiReF at different RETA levels. Similar to the naive method results, it includes comparisons between Revenue and Total Net Sales, as well as Total equity and Total stockholders' equity.

C Effect of Pre-training Data

There is a common concern regarding LLMs: whether LLMs simply memorize the pre-training data, rather than possessing understanding and reasoning abilities. This concern raises the question of **whether pre-training data might interfere with the experimental results**.

The short answer is NO. We use the same pre-training model (e.g., GPT-3.5 or GPT-4) for each comparison, the result will not be affected by the pre-training data. To know the impact of pre-training data containing documents on the results, we conducted relevant experiments in our study.

According to the available information, the datasets used for pre-training GPT-3.5 and GPT-4 were updated until September 2021. Therefore, we compared the 2019 and 2022 data in the FINE dataset. As shown in the Table 13 and Table 14, the 2022 Average RETA score is higher than the 2019 score for GPT-3.5. However, for GPT-4, the 2019 Average RETA score is higher than in 2022. In both sets of experiments, the differences in Average RETA scores are not substantial. Therefore, we believe that the influence of pre-training data can be neglected for our experiments.

D Analysis of Computational Costs

For the analysis of time costs, we have already analyzed in subsection 6.1. For the analysis regarding the number of LLMs calls, it is related to the number of retrieved segments (N_{seg}), the maximum length of the document segment summary (L_{sum}). For the Refine strategy, the number of calls equals the number of retrieved segments plus one: $N_{call} = N_{seg} + 1$, which is four calls of GPT-3.5 for one financial report. For the Map-Reduce strategy, N_{sum} represents the number of segment summaries, and N_{mer} represents the number of LLMs calls required to merge segment summaries,

Naive	**RETA 1%**	**RETA 3%**	**RETA 5%**	**RETA 10%**	**average**
Revenue	0.3056	0.3333	0.3438	0.3611	
Total Net Sales	0.2361	0.2465	0.2604	0.2847	
RPD	25.64%	29.94%	27.59%	23.66%	26.71%
Total Equity	0.0260	0.0303	0.0390	0.0519	
Total Stockholders' Equity	0.0521	0.0556	0.0660	0.0799	
RPD	66.90%	58.82%	51.48%	42.35%	54.89%

Table 11: Experimental results for the naive method in handling keyword ambiguity at different RETA levels

SiReF	**RETA 1%**	**RETA 3%**	**RETA 5%**	**RETA 10%**	**average**
Revenue	0.8576	0.8611	0.8681	0.8889	
Total Net Sales	0.8090	0.8299	0.8403	0.8542	
RPD	5.83%	3.70%	3.25%	3.98%	4.19%
Total Equity	0.4688	0.4861	0.5278	0.5556	
Total Stockholders' Equity	0.5660	0.5938	0.6042	0.6493	
RPD	18.79%	19.94%	13.50%	15.56%	16.95%

Table 12: Experimental results for SiReF in handling keyword ambiguity at different RETA levels

GPT-3.5	**RETA 1%**	**RETA 3%**	**RETA 5%**	**RETA 10%**	**Average**
2019	0.6200	0.6829	0.7029	0.7171	0.6807
2022	0.6417	0.6917	0.7222	0.7361	0.6979

Table 13: Accuracy comparison between samples from 2019 and 2022 using GPT-3.5.

GPT-4	**RETA 1%**	**RETA 3%**	**RETA 5%**	**RETA 10%**	**Average**
2019	0.8543	0.8857	0.8914	0.8914	0.8807
2022	0.7972	0.8444	0.8583	0.8778	0.8444

Table 14: Accuracy comparison between samples from 2019 and 2022 using GPT-4.

L_{mer} represents the length of summary that can be merged in one operation. In our experiment, only one merge operation is needed to merge all the segment summaries, so: $N_{sum} = N_{seg}$, $N_{mer} = 1$, $N_{call} = N_{seg} + 2$, which is five calls of GPT-3.5 for one financial report.

E Necessity of Considering Both Tabular Data and Textual Data

	RETA 1%	**RETA 3%**	**RETA 5%**	**RETA 10%**	**Average**
BOTH	**0.6389**	**0.6938**	**0.7194**	**0.7451**	**0.6993**
TBL	0.5361	0.6014	0.6215	0.6465	0.6014

Table 15: Accuracy comparison between using both tabular and textual data (**BOTH**), and using only tabular data (**TBL**).

In HLDs, there is many of information contained in tables, so there is a concern why not just using tabular data. To evaluate the necessity of considering both tabular data and textual data. We conducted experiments on FINE when using both tabular and textual data v.s. using only tabular data. The results are shown in the Table 15. It indicates the necessity of using both modalities.

F Prompts

F.1 Summarization Prompts - Refine

The Refine strategy consists of two prompts: the Question Prompt and the Refine Prompt. These prompts are designed to guide LLMs in extracting and summarizing key information related to the given keywords from many segments.

Question Prompt: This prompt is designed to instruct the LLMs to generate an initial summary containing information related to the given keywords from the provided document segment. The content of the question prompt is as follows:

```
>>>>>Your Task:
Given a segment of a financial report and
keywords.
You need to summarize the information
related to the keywords.
All values must be in millions and rounded
to three decimal places using rounding
rules.
>>>>>Example:
Financial report's segment: For company A
in 2022Q3, the revenue is $1.2345 billion;
the net income is $50.1245 million
-----
Keywords: Net income and revenue of company
 A in 2022Q3.
-----
Summary: For company A in 2022Q3, net
income is $50.125 million, and revenue is
$1,234.500 million.
>>>>>Question:
Financial report's segment: {
document_segment}
-----
Keywords: {keywords}
-----
Summary:
```

Refine Prompt: The refine prompt is designed to instruct LLMs to update the old summary by incorporating information related to the keywords from the newly provided document segment. The content of the refined prompt is as follows:

```
>>>>>Your Task:
Given a segment of a financial report, a
summary of the previous segments and
keywords.
You should combine the information related
to the keywords to generate a new summary.
All values must be in millions and rounded
to three decimal places using rounding
rules.
>>>>>Example:
Financial report's segment: For company A
in 2022Q4, the net income is $5 billion.
-----
Old summary: For company A, the net income
in 2022Q1 is $3.125 million; the net income
 in 2022Q2 is $123,123.000 million; the net
 income in 2022Q3 is $0.123 million.
-----
Keywords: Net income of company A in 2022.
-----
New summary: For company A, the net income
in 2022 is $128,126.248 million.
>>>>>Question:
Financial report's segment: {
document_segment}
-----
Old summary: {old_summary}
-----
Keywords: {keywords}
-----
New summary:
```

F.2 Summarization Prompts - Map-Reduce

The Map-Reduce strategy also consists of two prompts: the Map Prompt and the Reduce Prompt.

Map Prompt: This prompt is designed to instruct LLMs to generate a summary containing information related to the given keywords from the provided document segment. The content of the Map prompt is as follows:

```
>>>>>Your Task:
Given a segment of a financial report and
keywords.
You need to summarize the information
related to the keywords.
All values must be in millions and rounded
to three decimal places using rounding
rules.
>>>>>Example:
Financial report's segment: For company A
in 2022Q3, the revenue is $1.2345 billion;
the net income is $50.1245 million.
-----
Keywords: Net income and revenue of company
 A in 2022Q3.
-----
Summary: For company A in 2022Q3, net
income is $50.125 million, and revenue is
$1,234.500 million.
>>>>>Question:
Financial report's segment: {
document_segment}
-----
Keywords: {keywords}
-----
Summary:
```

Reduce Prompt: The Reduce prompt is designed to instruct LLMs to consolidate the summaries obtained from the Map process. The "text" in the prompt represents all the summaries generated by the Map process.

```
>>>>>Your Task:
Find the values of keywords in the given
content.
If you can't find the value, please output
"None".
If you find the corresponding value, please
 express it in millions and round to two
decimal places using rounding rules.
>>>>>Example 1:
Content: For company ABC, total net sales
for the three months ended June 25, 2022,
were $65.135 billion.
-----
Keywords: Total net sales of ABC for the
three months ended June 25, 2022.
-----
Result: 65,135.00
>>>>>Example 2:
Content: For company XYZ, total assets for
the three months ended 2022.10.15 were $2
.126 million.
-----
Keywords: Total assets of XYZ for the three
 months ended October 15, 2022.
-----
Result: 2.13
>>>>> Question
Content: {text}
-----
Keywords: {keywords}
-----
Result:
```

F.3 Extraction Prompt

This prompt extracts the numerical values corresponding to the specified keywords from the given content. If the value is not found, the prompt directs LLMs to output "None". If the value is found, it should be expressed in millions and rounded to two decimal places using rounding rules.

```
>>>>> Your task:
Find the values of keywords in the given
content.
If you can't find the value, please output
"None".
If you find the corresponding value,
please express it in millions and round to
two decimal places using rounding rules.
>>>>> Example 1:
Content: For company ABC, Total Net Sales
for the three months ended June 25, 2022,
were $65.135 billion.
Keywords: Total Net Sales of ABC for the
three months ended June 25, 2022.
Result: 65,135.00
>>>>> Example 2:
Content: For company XYZ, Total Assets for
the three months ended 2022.10.15 were $2
.126 million.
Keywords: Total Assets of XYZ for the three
 months ended October 15, 2022.
Result: 2.13
>>>>> Question:
Content: {text}
Keywords: {key_words}
Result:
```

F.4 Numerical Precision Enhancement Prompts

The Numerical Precision Enhancement Prompts aim to improve the precision of extracted numerical values by guiding the LLMs to preserve the required level of precision. These prompts come in different variations, each adding or modifying specific aspects to achieve the desired precision:

TD-O: This version of the prompt contains only a task description and task information. It does not provide explicit guidance on numerical precision.

```
>>>>Your Task:
Given a segment of a financial report and
keywords.
You need to summarize the information
related to the keywords.
>>>>Question:
Financial report's segment: {
document_segment}
-----
Keywords: {keywords}
-----
Summary:
```

TD-R: This version adds a precision requirement to the task description in the Naive prompt. It explicitly states that all values must be in millions and rounded to three decimal places using rounding rules.

```
>>>>Your Task: ... All values must be in
millions and round to three decimal places
using rounding rules ...
>>>>Question: ...
```

TD-S: Building on the Naive version, this prompt includes an input-output example. However, in this example, all the values are represented by variables x, y, and z. Therefore, this example doesn't provide any information about precision.

```
>>>>Your Task: ...
>>>>Example:
Financial report's segment: For company A
in 2022Q3, the revenue is $x billion; the
net income is $y million.
-----
Keywords: Net income and revenue of company
 A in 2022Q3.
-----
Summary: For company A in 2022Q3, net
income is $x million, and revenue is $y
million.
>>>>Question: ...
```

TD-RS: Combining the precision requirements from the Direct version and the example from the

Naive & Shot version, this prompt provides both explicit precision guidance and an example of the task, but without specific numerical values.

```
>>>>Your Task: ... All values must be in
millions and round three decimal places
using rounding rules ...
>>>>Example:
Financial report's segment: For company A
in 2022Q3, the revenue is $x billion; the
net income is $y million.
-----
Keywords: Net income and revenue of company
 A in 2022Q3.
-----
Summary: For company A in 2022Q3, net
income is $x million, and revenue is $y
million.
>>>>Question: ...
```

TD-SP: Building on the Naive & Shot version, this prompt demonstrates how to preserve the required precision by using an input-output example with numbers.

```
>>>>Your Task: ...
>>>>Example:
Financial report's segment: For company A
in 2022Q3, the revenue is $1.2345 billion;
the net income is $50.1245 million.
-----
Keywords: Net income and revenue of company
 A in 2022Q3.
-----
Summary: For company A in 2022Q3, net
income is $50.125 million, and revenue is
$1,234.500 million.
>>>>Question: ...
```

TD-RSP: This is the optimal prompt. It includes a precision requirement in the task description and an example demonstrating how to preserve the precision.

```
>>>>Your Task: ...All values must be in
millions and round to three decimal places
using rounding rules.
>>>>Example:
Financial report's segment: For company A
in 2022Q3, the revenue is $1.2345 billion;
the net income is $50.1245 million.
-----
Keywords: Net income and revenue of company
 A in 2022Q3.
-----
Summary: For company A in 2022Q3, net
income is $50.125 million, and revenue is
$1,234.500 million.
>>>>Question: ...
```