Enabling and Analyzing How to Efficiently Extract Information from Hybrid Long Documents with LLMs

Chongjian Yue[1]*, **Xinrun Xu**[3]*, **Xiaojun Ma**[2,†], **Lun Du**[2,†],
Hengyu Liu[4], **Zhiming Ding**[3], **Yanbing Jiang**[1], **Shi Han**[2], **Dongmei Zhang**[2]
[1]School of Software & Microelectronics, Peking University; [2]Microsoft;
[3]Institute of Software Chinese Academy of Sciences; [4]University of Technology Sydney
chongjian.yue@stu.pku.edu.cn, xuxinrun20@mails.ucas.ac.cn,
{xiaojunma, lun.du, shihan, dongmeiz}@microsoft.com,
hengyu.liu@uts.edu.au, zhiming@iscas.ac.cn, jyb@ss.pku.edu.cn

Abstract

Large Language Models (LLMs) demonstrate exceptional performance in textual understanding and tabular reasoning tasks. However, their ability to comprehend and analyze hybrid text, containing textual and tabular data, remains unexplored. Due to the hybrid text often appears in the form of hybrid long documents (HLDs), which is far exceed the token limit of LLMs. Consequently, we apply an naive split-recombination-based framework (SiReF) to enable LLMs process the HLDs and carry out experiments to analysis four important aspects of information extraction from HLDs. Given the findings: 1) The effective way to select and summarize the useful part of a HLD. 2) An easy table serialization way is enough for LLMs to understand tables. 3) The naive SiReF has adaptability in many and complex scenarios. 4) The useful prompt engineering to enhance LLMs on HLDs. To address the issue of dataset scarcity in HLDs and support the future work, we also propose the **Fi**nancial Reports **N**umerical **E**xtraction (**FINE**) dataset. The dataset and code are publicly available in the attachments.

1 Introduction

LLMs have exhibited remarkable capabilities in various natural language tasks, demonstrating their potential to comprehend and process intricate textual data (Wei et al., 2023a; Wang et al., 2023b; Zhou et al., 2022; Kojima et al., 2023). In addition to their success in textual data, the studies by (Chen, 2023; Ye et al., 2023) highlight the effectiveness of LLMs in handling tabular data. However, despite their proven proficiency in understanding and analyzing textual and tabular data individually, research exploring the capacity of LLMs to tackle hybrid documents, which combine these two types of data, remains relatively scarce.

Hybrid documents, adeptly blending textual and tabular content, are widely used across diverse fields and extensive in length typically. Extracting relevant information from hybrid long documents (HLDs) based on user-provided keywords is a crucial upstream task that supports various applications, such as question-answering systems (Gil, 2023), document classification (Mustafa et al., 2023), information retrieval (da Silva et al., 2023), and more.

Considering the excessive length of HLDs, LLMs can't directly process all the content in one HLD. If HLDs are subjected to simple truncation, substantial information loss will occur, resulting in compromised performance. So we apply an easy split-recombination-based framework (SiReF), which splits the document into many segments and extracts information from the retrieved segments. Within the SiReF, the following challenges of HLDs are expected to be addressed effectively by LLMs:

F1. In HLDs, keyword-related information is distributed in many segments. How we effectively select and summarize the useful segments? We carry out experiment from two dimensions: 1) We compare two different summarization strategies: Refine and Map-Reduce. Refine shows a better accuracy, while Map-Reduce shows a better efficiency. 2) We investigate the effect of the number of retrieved segments based of the similarity with keyword.

F2 Tables are commonly utilized in HLDs, but LLMs cannot directly interpret tabular data. What would be an optimal serialization format for LLMs to comprehend this information more effectively? We compare four different table serialization formats, and discovered that a simplified serialization format, devoid of much hierarchical information, is sufficient for LLMs.

*Equal contribution and work done as interns at Microsoft Research Asia.

†Corresponding author

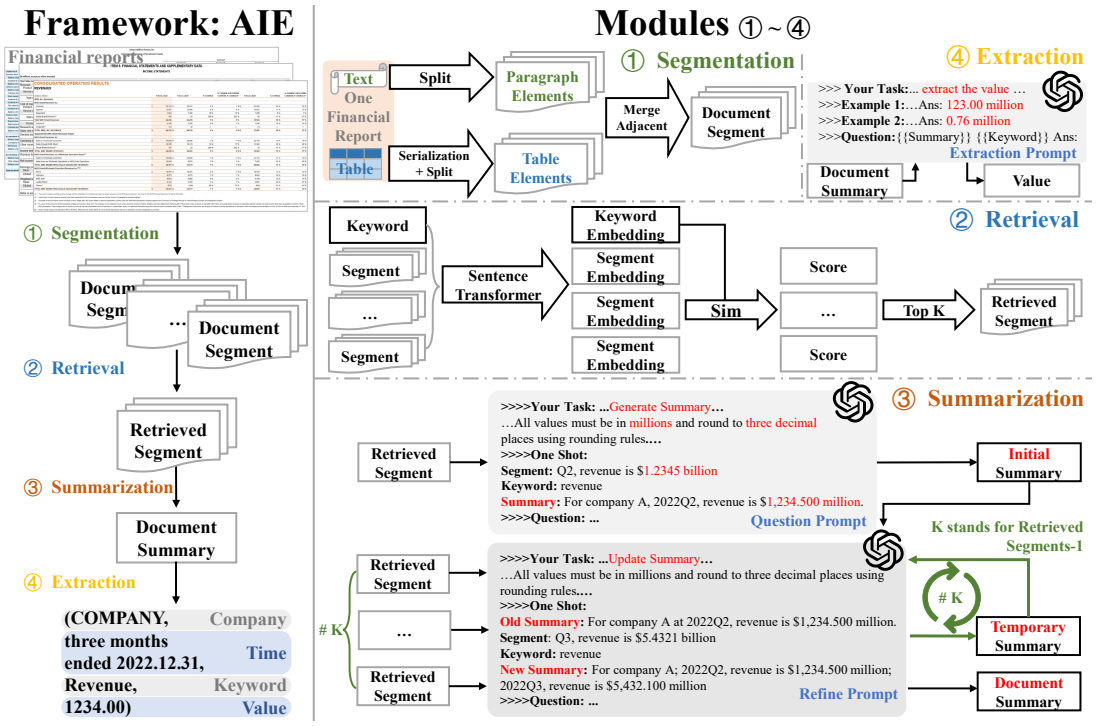

Figure 1: The AIE framework illustrates the end-to-end IE process, consisting of four modules: Segmentation, dividing lengthy documents into short segments; Retrieval, selecting the most relevant segments related to the given keyword; Summarization, using LLMs to generate a concise summary of relevant information; and Extraction, extracting the keyword-corresponding value from the summary. This framework is exemplified using financial reports.

F3 For the adaptability, HLDs is a complex domain. How about the adapability of SiReF? We carry out experiment from three dimension with the findings: 1) SiReF is versatile and can be applied to various domains; 2) SiReF is capable of handling the issue of ambiguity in expressions. 3) SiReF can adapt to LLMs with different capabilities.

F4 Prompt Engineering plays a critical role for LLMs in different scenario. Is there any prompt engineering can improve the effctivness of SiReF on information extraction from HLDs? We carry out experiment and give three kinds of usefule findings: 1) Numerical precision enhancement. In numerically sensitive scenarios, such as financial reports, SiReF can help in extracting more accurate numerical values. 2) Keyword Completion. With the help from the metadata of document, SiReF can better undertand user-given keyword. 3) Few-Shot Learning. The shot is useful to help SiReF undertand the task. But too many shot may decrease the performance of LLMs to specific tasks.

To answer some questions, we need to use the globally optimal implementation of SiReF. So we first introduce the SiReF and integrate globally optimal findings for the findings in each module (in section 2). And introduce the dataset and metric used for experiments (in subsection 3.1). Next, we show the overall potential of SiReF. Then, we will present experimental analyses of various aspects of SiReF in each section.

2 Prepared Work

2.1 Automated Information Extraction

In order to enable LLMs to handle HLDs, we propose a framework called Automated Information Extraction (AIE), specifically designed for extracting information from HLDs. The AIE framework consists of four modules: Segmentation, Retrieval, Summarization, and Extraction, as shown in Figure 1. AIE first segments documents into manageable pieces for LLMs, then retrieves the most relevant segments related to the keyword based on embedding, followed by summarizing the retrieved segments to compress and consolidate critical information and finally extracting the keyword-corresponding value 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 the optimal implementation based on our experimental

analysis.

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 a method to convert tables into a textual format. After comparing various methods (as discussed in section 7), we discover that the *PLAIN* serialization method, which separates cells with spaces and rows with newline characters, provides a simple yet effective way to represent table data in financial reports.

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 based on the LLM's maximum sequence length.

Merge: Merge adjacent elements as segments. The primary reason for merging is to maintain semantic relationships between adjacent elements. Most elements have a small token count (tens of tokens), which makes merging feasible. To achieve this, we concatenate adjacent small elements until the segment length limit is reached

2.3 Retrieval

Long documents contain many tokens, leading to a large number of document segments. Processing all segments would significantly increase LLMs invocations and introduce irrelevant information, potentially affecting extraction accuracy. Therefore, we adopt an embedding-based retrieval strategy (Li et al., 2021) to select the most relevant segments. We calculate the similarity between each document segment and the keyword based on their embeddings and retrieve the top-ranked segments with the highest similarity scores.

To obtain embeddings, we use the Sentence-Transformer model (Reimers and Gurevych, 2019) in this module. Due to its sequence length limitations smaller than LLMs, document segments are divided into multiple slices. The similarity between each slice and the keyword is calculated, and the maximum similarity value among the slices within a segment is considered as the similarity between the segment and the keyword.

2.4 Summarization

The content related to a 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. After comparing two common strategies (as discussed in section 9), we apply the **Refine Strategy** to maintain an evolving summary, updated with information from each segment.

The Refine Strategy process 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.

2.5 Extraction

After the summarization, we obtain a summary that contains the keyword's value along with a considerable amount of irrelevant information. To eliminate irrelevant information and improve the accuracy and efficiency of 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 Prompt Engineering

Prompt Engineering plays an important role in enhancing LLMs' ability (Huang and Chang, 2022). There are various types of prompt engineering that can be used in AIE. In this work, considering the characteristics of the task of extracting information from HLDs, we have explored the following three types of prompts.

Numerical Precision Enhancement: In scenarios with more numerical data, we find that LLMs have difficulties in maintaining accurate 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. However, in financial analysis, precision is essential for the work. To tackle this issue, precision control instructions are incorporated into the prompt, directing the model to produce precise responses. After comparing many precision-enhancing method (as discussed in section 10), we combine the use of two methods: Direct and Shot-Precision. The Direct method directly informs LLMs of the required precision, while the Shot-Precision method demonstrates how to manage precision through input-output examples.

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 section 11), providing more contextual information can greatly improve the accuracy of AIE.

Few-Shot Learning: In-context learning greatly influences LLMs' capabilities to understand the given task. According to our analysis (as discussed in section 12), we find a single well-designed shot is sufficient to guide LLMs to generate accurate answers. On the contrary, excessive shots may potentially reduce the performance of AIE.

3 Dataset

3.1 Datasets on 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: Basic statistics for FINE, WIKIR, and MPP datasets.

To assess LLMs' 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 9. Among these datasets, the financial dataset is used to analyze the various modules of AIE. 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 **F**inancial 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 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 LLMs' 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.

However, this issue does not exist in the WIKIR and MPP datasets. In the WIKIR dataset, the ground truth is a string. In the MPP dataset, the ground truth is a floating-point number, and this floating-point number has no alternative precision representation.

4 Overall Performance on Three Domains

AIE is a flexible framework. To showcase the overall performance of its various modules, we compare AIE with the naive method on all three datasets.

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

The AIE used in this section uses the optimal implementation for each module based on our findings (as discussed in the next sections). The naive method directly uses LLMs adopted to HLDs. We take the GPT-3.5 (text-davinci-003) as our primary subject. For the detailed configurations, please refer to Appendix C.

The Figure 2 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 3 displays the experimental results on WIKIR and MPP. It shows the average accuracy across all samples.

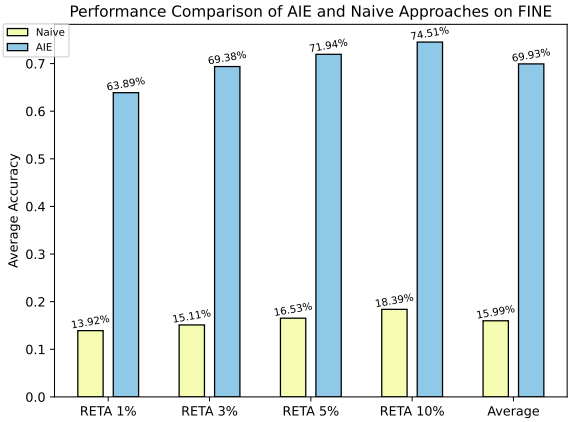

Figure 2: Comparison of the Naive method and AIE at different RETA levels on FINE.

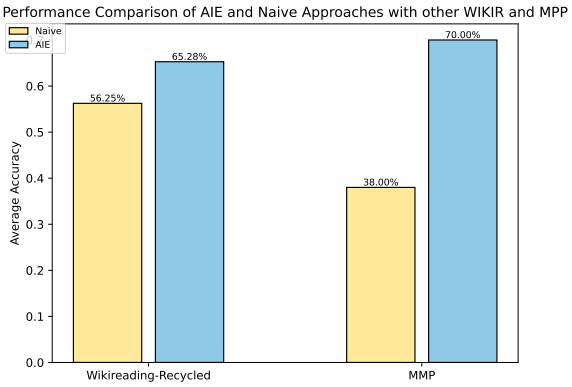

Figure 3: Comparison of the Naive method and AIE on WIKIR and MPP.

The experimental results demonstrate that the AIE method outperforms the naive method in all three datasets. The improvement in average accuracy indicates that the AIE method is more effective in extracting relevant information from various HLDs.

In Figure 2, as the RETA becomes more stringent, the performance gap between the naive method and AIE becomes larger. This indicates that AIE is capable of delivering more accurate results under stricter evaluation metrics.

5 Adaptability for LLMs with Different Capabilities

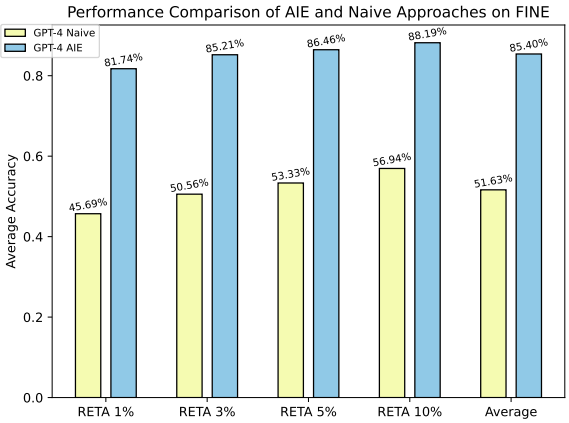

Figure 4: Comparison of the Naive method and AIE at different RETA levels on GPT-4.

To investigate the adaptability of AIE for LLMs with different capabilities, we also conduct experiments on GPT-4. For the reason that GPT-4 is currently the most outstanding LLM in terms of comprehensive capabilities. 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 subject.

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

6 Capability to Handle Ambiguity

In HDLs, the same concept may have multiple representations, which requires LLMs to have the ability to handle ambiguity. To evaluate whether AIE 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 AIE 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 5. From the results, we observe that AIE outperforms the naive method across all RETA levels when handling keyword ambiguity.

Specifically, comparing *Revenue* vs. *Total Net Sales*, AIE shows a 22.52% lower avg. RPD than the naive method. Similarly, for *Total Equity* vs. *Total Stockholders' Equity*, AIE yields a 37.94% lower avg. RPD than the naive method. For more detailed results, please refer to the Appendix E.

7 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 2: Accuracy comparison among PLAIN, CSV, XML, and HTML table serialization formats.

In order 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[2] 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 2, 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.

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

8 Analysis of Retrieved Segment Number

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

In this section, we investigate the effect of the number of retrieved segments on the performance of our framework. Table 3 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.

9 Analysis of Summarization Strategies

In order to extract information from multiple retrieved segments, several popular strategies are available. Besides the Refine Strategy, another widely used strategy is the Map-Reduce Strategy, which is known for its parallel processing capabilities. As illustrated in Figure 6, the Map-Reduce Strategy aims to combine summaries from document segments, comprising two stages: Map and Reduce. 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.

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.

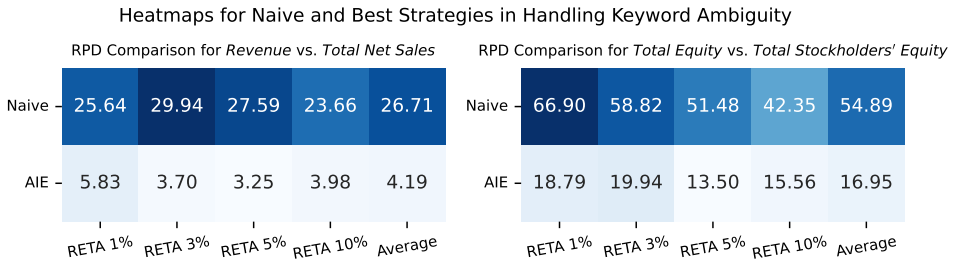

Figure 5: Exploring the Capability to Handle Keyword Ambiguity: Comparison of Naive and AIE on RPD

	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: Accuracy comparison between Map-Reduce and Refine strategies across various RETA levels.

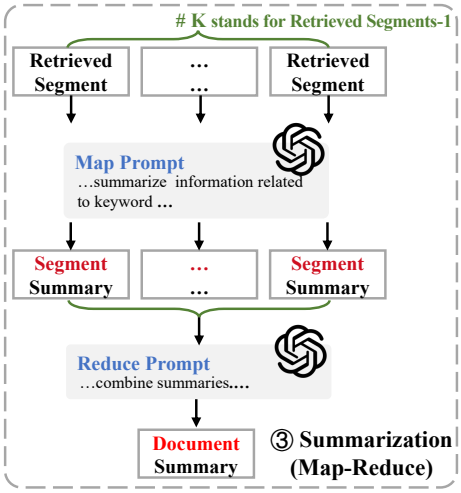

Figure 6: Illustration of the Map-Reduce Strategy, comprising two stages: Map, generating individual segment summaries, and Reduce, combining these summaries to form a single document summary.

10 Analysis of Numerical Precision Enhancement

In order to enable LLMs to extract more accurate numerical values, we design various numerical precision enhancement prompts. To assess the performance of these prompts, we conducted a comparative experiment under finer RETA levels.

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

From Table 5, 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.

	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 5: Accuracy comparison for different prompts aimed at enhancing numerical precision.

11 Analysis of Keyword Completion

To analyze the effectiveness of keyword completion in improving LLMs' performance, we conducted an experiment 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".

	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.

As shown in Table 6, we find that the perfor-

mance 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 LLMs' accuracy. By providing more specific information, such as company names and time periods, the model can better understand the context and generate more accurate responses, leading to an overall improvement in performance.

12 Analysis of Shot Number

Few-shot learning is an important ability of LLMs. To investigate the impact of the number of shots on AIE's performance, we conducted an experiment with different numbers of shots, ranging from 0 to 3.

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

As shown in Table 7, 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 LLMs 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 LLMs for information extraction. 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.

13 Discussion

In addition to our extensive exploration experiments with AIE, we also eliminate the interference of pre-trained datasets on the experiments Appendix F, ensuring the reliability of our results. We find that using AIE significantly reduces the number of LLM invocations Appendix G. Furthermore, it is essential to use both tabular and textual data simultaneously in HLDs Appendix H.

14 Related Work

Information extraction. Early IE methods predominantly relied on rule-based approaches. (Sheikh and Conlon, 2012; Farmakiotou et al., 2000) proposed rule-based Named Entity Recognition (NER) models that utilize domain-specific characteristics. Among them, some approaches concentrate on tables (Brito et al., 2019), losing crucial textual information. Recently, models based on Machine Learning emerged, in which bidirectional RNN classifier (Ma et al., 2020) is employed for learning tables, and BERT (Hillebrand et al., 2022) for text respectively. FinQA (Chen et al., 2022), TAT-QA (Zhu et al., 2021), and MULTIHIERTT (Zhao et al., 2022) learn HLDs for Question Answer (QA) task, which focus on parts of the reports rather than the whole context.

LLMs. 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).

15 Conclusion

In order to enable LLMs to extract information from HLDs, we propose an **A**utomated **I**nformation **E**xtraction (**AIE**) framework, comprising four modules: Segmentation, Retrieval, Summarization, and Extraction. To analyze the various modules of AIE, we construct a dataset from publicly available financial reports, called **Fi**nancial Reports **N**umerical **E**xtraction (**FINE**). Based on the FINE, experiments offer a detailed explanation of the impact of each module on AIE's ability to extract information from HLDs. We also validate the overall

performance of AIE on three different domains: scientific papers, Wikipedia, and financial reports. Experimental results show that AIE significantly improves the ability of LLMs to handle HLDs.

Limitations

Despite the substantial enhancement achieved by LLMs through the utilization of AIE, 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: AIE 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, AIE 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*.

Eduardo Brito, Rafet Sifa, Christian Bauckhage, Rüdiger Loitz, Uwe Lohmeier, and Christin Pünt. 2019. A hybrid ai tool to extract key performance indicators from financial reports for benchmarking. In *Proceedings of the ACM Symposium on Document Engineering 2019*, pages 1–4.

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

Zhiyu Chen, Wenhu Chen, Charese Smiley, Sameena Shah, Iana Borova, Dylan Langdon, Reema Moussa, Matt Beane, Ting-Hao Huang, Bryan Routledge, and William Yang Wang. 2022. Finqa: A dataset of numerical reasoning over financial data.

Sherlon Almeida da Silva, Evangelos E. Milios, and Maria Cristina Ferreira de Oliveira. 2023. Evaluating visual analytics for relevant information retrieval in document collections. *Interact. Comput.*, 35(2):247–261.

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.

Dimitra Farmakiotou, Vangelis Karkaletsis, John Koutsias, George Sigletos, Constantine D Spyropoulos, and Panagiotis Stamatopoulos. 2000. Rule-based named entity recognition for greek financial texts. In *Proceedings of the Workshop on Computational lexicography and Multimedia Dictionaries (COMLEX 2000)*, pages 75–78.

Jorge Martínez Gil. 2023. A survey on legal question-answering systems. *Comput. Sci. Rev.*, 48:100552.

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.

Lars Hillebrand, Tobias Deußer, Tim Dilmaghani, Bernd Kliem, Rüdiger Loitz, Christian Bauckhage, and Rafet Sifa. 2022. Kpi-bert: A joint named entity recognition and relation extraction model for financial reports. In *2022 26th International Conference on Pattern Recognition (ICPR)*, pages 606–612. IEEE.

Jie Huang and Kevin Chen-Chuan Chang. 2022. Towards reasoning in large language models: A survey. *arXiv preprint arXiv:2212.10403*.

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*.

Zhiqiang Ma, Steven Pomerville, Mingyang Di, and Armineh Nourbakhsh. 2020. Spot: A tool for identifying operating segments in financial tables. In *Proceedings of the 43rd International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 2157–2160.

Ghulam Mustafa, Abid Rauf, Ahmad Sami Al-Shamayleh, Muhammad Sulaiman, Wagdi Alrawagfeh, Muhammad Tanvir Afzal, and Adnan Akhunzada. 2023. Optimizing document classification: Unleashing the power of genetic algorithms. *IEEE Access*, 11:83136–83149.

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*.

Nils Reimers and Iryna Gurevych. 2019. Sentence-bert: Sentence embeddings using siamese bert-networks. *arXiv preprint arXiv:1908.10084*.

Mahmudul Sheikh and Sumali Conlon. 2012. A rule-based system to extract financial information. *Journal of Computer Information Systems*, 52(4):10–19.

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*.

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*.

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

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*.

Fengbin Zhu, Wenqiang Lei, Youcheng Huang, Chao Wang, Shuo Zhang, Jiancheng Lv, Fuli Feng, and Tat-Seng Chua. 2021. Tat-qa: A question answering benchmark on a hybrid of tabular and textual content in finance.

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[3]. 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 financial reports 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 the ambiguous, HLDs characteristics of financial reports. In FINE, ground truth is represented as tuples of four elements: (company, time, keyword, value)[4]. These values are expressed in millions and rounded to two decimal places using conventional rounding techniques, providing

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

[4] 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.

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 labeled by four people to minimize labeling 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[5] and Google Finance[6], 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 final set of 9 keywords (as presented in Table 8) 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 9 illustrates the specific representation of *Revenue* in various companies' financial reports. In FINE, we systematically document ambiguous expressions of all keywords across various companies.

Consolidated Condensed Statements of Income

	Three Months Ended		Nine Months Ended	
(In Millions, Except Per Share Amounts; Unaudited)	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.

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

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

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

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.

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* $\times$ $104K/s$..." (Content), we can know that the *cooling rate* (Key) of $Al_{87}Ni_9Ce_4$ is $1.02 \times 104K/s$ (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.

B Detailed Explanation of Four Challenges

Hybrid documents, adeptly blending textual and tabular content, are widely used across diverse fields, and generally exhibit the following characteristics:

1) Long Documents: The three types of HLDs selected for our experiments all encompass extensive text and tables related to relevant keywords, resulting in considerable length and complexity. We found the average length of 59,464 tokens in experiments, equating to 14.5 times the max tokens of GPT-3.5 and 1.8 times that of GPT-4.

2) Hybrid Content: HLDs usually present the same information in different descriptive formats. Take financial reports as an example, Figure 7 presents a table extracted from Intel's 2022Q3 financial report, while Figure 8 illustrates a textual description. Both representations convey the same keywords: *Revenue* with different numerical precision. To obtain keyword values accurately and effectively, it is essential to concurrently analyze both content types.

3) Ambiguous Representation and **4) Numerical Precision: In financial reports**, Varying expressions for the same keyword in HLDs across companies lead to ambiguity. Figure 7 displays a table from Intel's 2022Q3 financial report, highlighting the *Net Revenue* of $15,338 million. While, Figure 8 features a textual description from the same report, indicating a *Revenue* of $15.3 billion. Both representations communicate the same KPI: *Revenue*. The presentation of identical information in varying formats is referred to as financial report ambiguity. Table 9 exemplifies this situation, demonstrating variations in *Revenue* representation across different companies. The textual and tabular contexts exhibit different numerical precision. **In scientific papers**, cases are usually like this: the ground truth is ($Al_{0.07602085}Cu_{49.95655951}Gd_{0.01086012}Zr_{49.95655951}$, 10, K/s), while the source in the original text is "... $(Cu_{50}Zr_{50})_{92}Al_7Gd_1$ *is estimated to be 10Ks*$^{-1}$ ..."; ($Ni_{38}Zr_{62}$, 104, K/s), v.s. "... *104 K/s for the binary* $Zr_{62}Ni_{38}$ ...". And different numerical precision appears again.

These four structural properties render HLDs comprehension challenging for LLMs. To accurately obtain keywords and their values, both textual and tabular types must be analyzed concurrently for numerical precision.

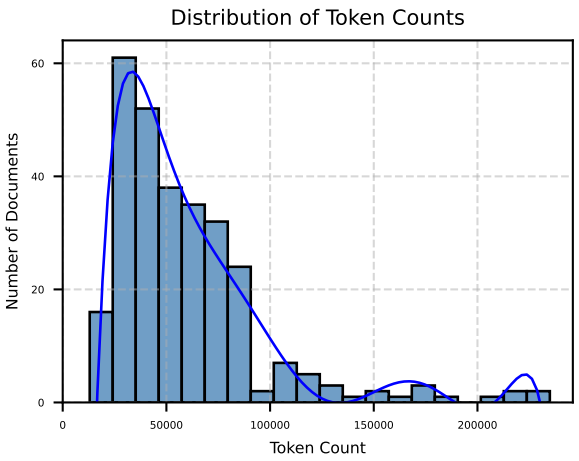

Figure 9: Histogram of token counts in financial documents.

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 8: Nine Keywords in FINE.

C Detailed Experiment Settings

Token Allocation: We allocate tokens to accommodate the model's maximum sequence length and the requirements of each AIE module. The token allocations are presented in the Table 10.

Prompts: In AIE, there are many different types of prompts serving various AIE modules: question prompts, refine prompts, extraction prompts, and so on. In these prompts, we apply our three prompt engineering. Appendix D shows the details of the prompts.

Retrieval Settings: We select the top three document segments with the highest similarity scores for GPT-3.5.

Embedding Model: We use the sentence-transformers/all-mpnet-base-v2[7] model for computing embeddings. This model can handle a sequence length of 384 tokens.

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

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 9: 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.

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

Table 10: Token allocation

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

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 12: 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 13: Accuracy comparison between samples from 2019 and 2022 using GPT-4.

D Prompts

D.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 round it
 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 refine 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 round it
 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:
```

D.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 round it
 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 it 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:
```

D.3 Extraction Prompt

Extraction Prompt for GPT-3.5: This prompt is designed to extract 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 it
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:
```

D.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:

Naive: 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:
```

Direct: 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 it to three decimal
places using rounding rules ...
>>>>Question: ...
```

Naive & Shot: 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: ...
```

Direct & Shot: 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 it to 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: ...
```

Naive & Shot-Precision: 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: ...
```

Direct & Shot-Precision: 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 it 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: ...
```

E Detailed Results of Keyword Ambiguity Experiment

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

Table 14 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 15 displays the experimental results for AIE 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.

F 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

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

AIE	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 15: Experimental results for AIE in handling keyword ambiguity at different RETA levels

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. And 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 GPT3.5 and GPT4 were updated until September 2021. Therefore, we compared the 2019 and 2022 data in the FINE dataset. As shown in the Table 12 and Table 13, the 2022 Average RETA score is higher than the 2019 score for GPT3.5. However, for GPT4, 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.

G Analysis of Computational Costs

For the analysis of time costs, we have already analyzed in section 9. 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, 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.

H Necessity of Considering Both Tabular Data and Textual Data

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 11. It indicates the necessity of using both modalities.