DocFinQA: A Long-Context Financial Reasoning Dataset

Varshini Reddy, Rik Koncel-Kedziorski, Viet Dac Lai
Michael Krumdick, Charles Lovering, Chris Tanner
Kensho Technologies
varshini.bogolu@kensho.com

Abstract

For large language models (LLMs) to be effective in the financial domain – where each decision can have a significant impact – it is necessary to investigate realistic tasks and data. Financial professionals often interact with documents spanning hundreds of pages, but most financial research datasets only deal with short excerpts from these documents. To address this, we introduce a long-document financial QA task. We augment 7,437 questions from the existing FinQA dataset with full-document context, extending the average context length from under 700 words in FinQA to 123k words in DocFinQA. We conduct extensive experiments over retrieval-based QA pipelines and long-context language models. Based on our experiments, DocFinQA proves a significant challenge for even state-of-the-art systems. We also provide a case study on a subset of the longest documents in DocFinQA and find that models particularly struggle with these documents. Addressing these challenges may have a wide-reaching impact across applications where specificity and long-range contexts are critical, like gene sequences and legal document contract analysis. DocFinQA dataset is publicly accessible¹.

1 Introduction

The frequent need to reason over large volumes of textual and tabular data makes financial analysis particularly challenging for LLMs (Azzi et al., 2019). Existing work on automating financial numerical reasoning focuses on unrealistically specific document snippets (Chen et al., 2021; Zhu et al., 2021). Datasets are often limited to pre-selected document sections, failing to reflect the broader and more realistic scenarios faced by analysts (Masson and Montariol, 2020). Financial professionals usually sift through hundreds of pages per document, requiring a deep understanding of

¹https://huggingface.co/datasets/kensho/DocFinQA

Figure 1: DocFinQA extends FinQA to documents often over 150 pages long (100K+ tokens), so it is difficult to find the pertinent information. The question for the example above is: “For the quarter December 31, 2012 what was the percent of the total number of shares purchased in December?” The correct answer is 16.5%.

both content and structure to navigate and extract pertinent information effectively. Current long-document QA datasets such as NarrativeQA Kočiský et al. (2018) do not test the quantitative reasoning skills needed in the financial domain.

In this work, we introduce DocFinQA, a long-document financial question-answering task. We extend the FinQA dataset of expert annotated questions and answers (Chen et al., 2021) with full Securities and Exchange Commission (SEC) reports. This results in a significantly longer context in the DocFinQA dataset – by a factor of 175 – than the FinQA dataset. Additionally, we manually verified and annotated questions of the test set. The resulting long-document QA task offers a more realistic evaluation of a model’s reasoning capabilities over financial documents. In line with recent work on program synthesis for financial QA (Koncel-Kedziorski et al., 2023), the questions in DocFinQA are appended with Python programs to generate the answers, allowing for training and evaluating program synthesis models for use in
realistic financial workflows.

Using this setup, we evaluate retrieval-based and long-context LLM systems. We study a typical retrieval pipeline that chunks and encodes the document, searching for the best chunks given a question, and passing the question and top- \( k \) chunks to a generative QA model (Hsu et al., 2021).

We also evaluate retrieval-free approaches using long-context LLMs (Weston and Sukhbaatar, 2023). Our results show that the successful employment of LLMs in financial settings requires further study of the specific nuances of the financial domain, such as context disambiguation. Our dataset represents a step towards better capturing these nuances.

## 2 Related Work

Prior studies in financial question answering focus on non-numerical reasoning (Day and Lee, 2016; Jørgensen et al., 2023; Maia et al., 2018). Short-context grounded numerical reasoning tasks were introduced with datasets such as FinQA (Chen et al., 2021) and TAT-QA (Zhu et al., 2021). Recently, understanding long documents has attracted more attention for tasks involving events (Yang et al., 2018), table of contents (Bentabet et al., 2020), and causal relations (Mariko et al., 2022). However, to the best of our knowledge, this is the first attempt to address financial numerical QA grounded in long documents with upwards of hundreds of pages of context for each question.

Long-document QA has been studied in NLP with the introduction of datasets such as SearchQA (Dunn et al., 2017), NarrativeQA (Kočiský et al., 2018), QuALITY (Pang et al., 2022), and PDF-Triage (Saad-Falcon et al., 2023). Due to the limited context size of LLMs, retrieval-based models are commonly used to filter irrelevant text (Izacard et al., 2022; Lewis et al., 2020). Recently, advances in attention mechanisms (Beltagy et al., 2020; Dao et al., 2022) and positional embeddings (Press et al., 2021; Su et al., 2023) allow for end-to-end grounded QA with context windows of more than 100k tokens. However, these methods suffer from loss of important context (Zhang et al., 2023) and often fail to make full use of longer inputs (Liu et al., 2023). Our work studies the intersection of numerical reasoning and long-document processing, and our results demonstrate that there is still ample room for improvement in this domain.

## 3 DocFinQA Dataset

### Dataset Representation:
Each question in FinQA is a triplet \( (c^{\text{golden}}, q, a) \) composed of a golden context \( c^{\text{golden}} \), a question \( q \), and an answer \( a \) written in human language. An example of FinQA is shown in Table 5 (See Appendix A). We extend the dataset in two ways: (1) context \( c^{\text{golden}} \) is extended to the full document context \( D \), and (2) we added a Python program \( p \) that produces the answer \( a \). Each final sample in DocFinQA is a quartet \( (D, q, p, a) \). An example of DocFinQA is shown in Table 6 (See Appendix A).

### Filings Collection:
For each question of the FinQA dataset, we identify the corresponding SEC filing from which it was created. We retrieve the filing in HTML/XML format from SEC’s EDGAR service and parse the text and table into clean markdown format (Wang et al., 2023). The collection and parsing processes are presented in more detail in Appendix A and Appendix B, respectively. Figure 2 shows the distribution of document lengths in DocFinQA.

![Figure 2: Histogram of document length (#words) in DocFinQA dataset with dash line representing the average length of the documents. The purple line depicts the proportion of documents where the question context is within the current number of words.](image)

**Chunking and Alignment:** To study retrieval-based QA systems, we split each document \( D \) into a set of chunks \( C = \{c_1, \ldots, c_n\} \). Each chunk consists of 2,750 characters (~509 tokens) with a 20% overlap to avoid missing context at the edges. To compute the performance, we identify the best context chunk, \( c^\star \), from the chunk set \( C \) associated with each document \( D \) that includes the information to answer question.
Table 1: Comparison of DocFinQA and existing Finance QA and Long Document QA datasets. DocFinQA includes multi-page documents with both numeric and tabular data.

<table>
<thead>
<tr>
<th>Dataset</th>
<th>#Docs</th>
<th>#QAs</th>
<th>#Words</th>
<th>Multi-page</th>
<th>Numeric</th>
<th>Tabular</th>
</tr>
</thead>
<tbody>
<tr>
<td>NarrativeQA</td>
<td>1,572</td>
<td>46,765</td>
<td>63,000</td>
<td>✓</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>QuALITY</td>
<td>381</td>
<td>6,737</td>
<td>5,159</td>
<td>✓</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>PDFTriage</td>
<td>82</td>
<td>908</td>
<td>12,000</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>TAT-QA</td>
<td>2,757</td>
<td>16,552</td>
<td>260</td>
<td>-</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>FinQA</td>
<td>2,789</td>
<td>8,281</td>
<td>687</td>
<td>-</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>DocFinQA</td>
<td>801</td>
<td>7,437</td>
<td>123,453</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
</tr>
</tbody>
</table>

4 Retrieval-based QA Evaluation

Retrieval Task: We test three models for context retrieval: ColBERT (ColB) (Khattab and Zaharia, 2020), Sentence-BERT (SentB) (Reimers and Gurevych, 2019), and OpenAI’s Ada (Greene et al., 2022). Further, we finetune the ColBERT model (FT ColB) on the training set of DocFinQA to evaluate an in-domain model. More details on the fine-tuning process are given in Appendix G. We also test a matching-based model, BM25 (Robertson et al., 1995), but observe poor performance (See Appendix F for details).

To retrieve context for a question $q$ over chunk set $C$, we encode both $q$ and $C$ with the encoding models mentioned above. This results in an embedding, $v_q$, for the question and chunk embeddings $V_C = \{v_c | c_i \in C\}$. We compute the
cosine similarity between \( v_q \) and each vector in \( V_C \) to retrieve the top-k most similar chunks. We evaluate these models using HR@k on the test set of DocFinQA using the target \( c^\star \). Results are shown in Figure 3. The FT ColB yields the highest HR, followed by ColB. FT ColB yields an average improvement of 91% HR over SentB and obtains a 0.35 (HR@1) and 0.55 (HR@3).

### 4.1 Question Answering Task

We formulate the QA task as a few-shot in-context learning task (Brown et al., 2020). For each in-context example, we only provide the relevant chunk and the answer. For the actual query, we provide \( k \) chunks. More details of the few-shot settings are provided in Appendix H.

We evaluate Falcon (Penedo et al., 2023), MPT (MosaicML, 2023), LLaMa 2 and CodeLlama (Touvron et al., 2023), Mistral (Jiang et al., 2023), GPT3.5 (Brown et al., 2020; OpenAI, 2023) models. We weren’t able to evaluate proprietary models such as GPT3 (Brown et al., 2020), BloombergGPT (Wu et al., 2023) due to their inaccessibility. We skipped models that were not finetuned for code generation such as PIXIU (Xie et al., 2023) and FinGPT (Yang et al., 2023) due to their poor performance. We also skipped models trained for other languages such as BBT-Fin (Lu et al., 2023) and XuanYuan 2.0 (Zhang and Yang, 2023).

### Performance on the development set

Table 2 reports the full performance of the development set with four retrieval models and three few-shot settings. This results in a total of twelve unique configurations. For both fine-tuned and pre-trained models, we use greedy decoding whenever applicable. One trend noted was that all generic LLMs showed higher accuracy with shorter context and more few-shot examples i.e. top chunk with 3 shots. While code-based LLMs such as Starcoder and CodeLlama showed higher accuracy with longer context i.e. top 3 chunks with 3 shots. This trend is also depicted in Figure 10.

Table 2: Performance of the models on DocFinQA in one-shot and few-shot in-context learning settings for the top 1 and top 3 retrieved chunk contexts on the development set. For each model, the best performance among all configurations is in **bold**. For each model, the best performance among different configurations for the same retrieval model is underlined. Top 1 and Top 3 indicate the number of retrieved chunks used as context for a configuration.

*The single original context chunk from the FinQA test set is used to estimate the upper bound.

**Figure 3:** Hit rate of retrieval models.

**Performance on the test set:** Table 3 reports the performance of the same 10 state-of-the-art models on the test set of DocFinQA. The few-shot setting and retrieval model configuration for each LLM are treated as hyperparameters and are picked based on the performance of the development set. We observe that larger models outperform smaller models (e.g., MPT 30B vs MPT 7B). Models trained on code yield higher accuracy than non-
Table 3: Performance on DocFinQA test set. For each row, the best performance among all retrieval models is in **bold**. The fewshot setting is selected based on the best performance on the development set (See Table 2).

code models (e.g., CodeLlama vs Llama). Models with additional supervised finetuning (e.g., LLama 2/7B+SFT) and instruction tuning (e.g., GPT-3.5) are among the best examined. Notably, Mistral 7B outperforms several larger models, although it lags behind Llama 2/7B+SFT and GPT-3.5.

The **FT ColB** model is the best retrieval model in all but one setting. It yields a marginal but consistent improvement over the **ColB**, and a large improvement over **SentB** and Ada.

5 Case Study w/ 100K+ Token Documents

Recent LLMs can handle context lengths of 128K tokens, but more than 40% of the documents in DocFinQA remain unanswerable even at this content length (see Figure 2). Here, we evaluate performance on a test subsample of 200 randomly selected documents, each of which has 100K or more tokens due to the monetary and temporal costs of human evaluation and GPT4.

We explore two retrieval-free options - System 2 Attention (S2A) and Iterative method. S2A extracts relevant information from each 100K-token chunk of a document before answering the question using the combined extracted information as context (Weston and Sukhbaatar, 2023). The Iterative method produces the output program iteratively as the LLM processes each 100k section of the document. A temporary answer program (initially “None”) is input with each section to the LLM. We also report the performance of the best retrieval-based model (Retrieval) based on the experiment in Section 4.

We conducted human evaluations on these 200 questions highlighting the challenging nature of this dataset with experienced but non-expert human participants (See Appendix J for details). Non-expert human performance on DocFinQA is lower than human performance reported in FinQA (Chen et al., 2021) (41% versus 50.7%). This can be attributed to the difficulty of finding the golden page, compared to the golden page being given in FinQA. Notably, the expert performance reported in FinQA is 91.2%.

Nonetheless, the non-expert human performance is double that of retrieval-free GPT-4 on these long documents, and roughly triple that of retrieval-free Mistral models. The performance of the iterative method was worse than S2A for both GPT-4 and Mistral with a reduced accuracy of 3% and 4%, respectively. With retrieval, both Mistral and GPT-4 outperform their retrieval-free counterparts, with the assisted GPT-4 now on par with the human cohort. Together, these results highlight that DocFinQA is a difficult test for long-document QA and that there is still room for significant improvement in this domain. For instance, further exploration into methods that combine information across multiple calls to a document-processing LLM is warranted.

6 Conclusion

This paper introduces a realistic document-level question-answering dataset over financial reports. Each question includes a full financial report (averaging 123K words), a far greater challenge than previous work that hones in on pre-specified content. Our findings reveal that this more realistic setting presents a significantly more difficult challenge, thereby opening new avenues for research in quantitative financial question answering.

Table 4: Retrieval-free performance on a case-study of 100K+ token documents.
Acknowledgement

We express our gratitude to our colleagues at Ken- sho Technologies and S&P Global for their invaluable contributions to data annotation, which greatly enhanced the completion of this project. We extend our appreciation to the anonymous reviewers for their supportive input.

Limitation

This work introduced an extension of the existing FinQA dataset. Due to limited human resources, we only validated the test set while the training and the development set were not fully validated. As a result, we can not make any claim of bias and question quality in the not-yet-validated data points offered in this paper. Additionally, as discussed in section 3, the code provided in this work was generated by WizardCoder LLMs. We assume that the code is correct if it produces correct or approximately close to the golden answer. This method may generate both false positive codes (the code that generates the correct answer with incorrect rationales) and false negative codes (the correct code that fails the approximation test).

Broader Impact and Ethical Considerations

We do not foresee any considerable risks associated with our work given that it is an extension of an open-source dataset and uses publicly available documents. To uphold transparency, the paper provides detailed documentation of the dataset creation process, including the sources of data and annotation details. Our dataset serves as a resource to underscore the need for longer context-oriented benchmarks both within and outside the financial domain and does not intend to criticize one or more LLMs.

The annotation in this work is done automatically or in-house, so no crowd-sourced or contract annotators were hired throughout the process. The human evaluation in this study was done by full-time paid coworkers known to the authors.

References


A SEC Filing Collection

Each data point in the FinQA dataset consists of a document identification field as shown in Table 5. This field is made up of 3 sections separated by a forward slash. The first is a string called company ticker symbol, the second refers to the year in which this document was filed and the third is the page number in the document where the answer can be found.

Downloading the right 10-K filing from the SEC begins with identifying the company code from the company ticker symbol. For example, C/2017/page_328.pdf-1 in FinQA maps to the CITIGROUP INC with company code 831001. This mapping is obtained from the official file released by SEC which can be found here https://www.sec.gov/file/company-tickers. We automatically generate a URL using the company code obtained. From the SEC website, either filings are downloaded as TXT, HTML, or XBRL using the generated URL. At this stage, approximately 6.5% (or 543) data points corresponding to approximately 9.4% (or 17) documents were dropped, either due to lack of mapping or non-availability of older documents. Further, the conversion of the downloaded files to PDF caused a loss of 117 data points (19 unique documents) due to formatting issues.

ID: C/2017/page_328.pdf-1

Context:

Performance graph comparison of five-year cumulative total return the following graph and table compare the cumulative total return on Citi 2019s common stock, which is listed on the NYSE under the ticker symbol 201cc 201d and held by 65691 common stockholders of record as of January 31, 2018, with the cumulative total return of the S&P 500 index and the S&P financial index over the five-year period through December 31, 2017. The graph and table assume that $100 was invested on December 31, 2012 in Citi 2019s common stock, the S&P 500 index and the S&P financial index, and that all dividends were reinvested. Comparison of five-year cumulative total return for the years ended date Citi S&P 500 financials.

<table>
<thead>
<tr>
<th>DATE</th>
<th>CITI</th>
<th>S&amp;P 500</th>
<th>S&amp;P FINANCIALS</th>
</tr>
</thead>
<tbody>
<tr>
<td>31-Dec-2012</td>
<td>100.0</td>
<td>100.0</td>
<td>100.0</td>
</tr>
<tr>
<td>31-Dec-2013</td>
<td>131.8</td>
<td>132.4</td>
<td>135.6</td>
</tr>
<tr>
<td>31-Dec-2014</td>
<td>137.0</td>
<td>150.5</td>
<td>156.2</td>
</tr>
<tr>
<td>31-Dec-2015</td>
<td>131.4</td>
<td>152.6</td>
<td>153.9</td>
</tr>
<tr>
<td>31-Dec-2016</td>
<td>152.3</td>
<td>170.8</td>
<td>188.9</td>
</tr>
<tr>
<td>31-Dec-2017</td>
<td>193.5</td>
<td>208.1</td>
<td>230.9</td>
</tr>
</tbody>
</table>

Question:

What was the percentage cumulative total return for the five year period ended 31-dec-2017 of citi common stock?

Answer:

93.5%

Table 5: Example from FinQA dataset. The context provided here has been formatted from the original dataset values.
What was the average cash flow from 2004 to 2006?

Program:

```
net_cash_2006 = 957.4
net_cash_2005 = 769.1
net_cash_2004 = 950.4

total_net_cash = net_cash_2006 + net_cash_2005 + net_cash_2004

average_net_cash = total_net_cash / 3

answer = average_net_cash
```

Answer:

892.3

Table 6: Examples from DocFinQA dataset with text and tables from entire SEC document as context (truncated for legibility), question, associated program and answer. A full report can be founded here https://www.annualreports.com/HostedData/AnnualReportArchive/k/NYSE_K_2006.pdf
### B Parsing SEC Filings

Since each filing contains many tables, maintaining the structure and order during extraction is critical for numerical reasoning. We convert each HTML-formatted filing to PDF format and use a finance-specific PDF extractor to parse the filing into markdown format. This process ensures that: (i) our dataset is grounded in the relevant financial documentation and (ii) all the tables in the filings are parsed with high precision into a consistent format without any HTML-tag noise.

We explore different methods for parsing SEC filings consisting of HTML and XML markup into text and markdown tables for use in our QA systems. To evaluate parsing strategies, we measure HR@k (Hit Rate @ k) when searching for the gold chunk among all document chunks for a single document using the FinQA question as the search query. Queries and document chunks are encoded with OpenAI’s ADA model. We compare BeautifulSoup, a standard library for manipulating HTML and XML formatted data, and Kensho Extract, a finance-specific text and table extraction model. Figure 4 shows the performance of these two methods.

Additionally, we note a better downstream performance of finance-specific models with Kensho Extract retrieved-context compared to that of Beautiful Soup. Qualitative analysis of the different parsers reveals that Kensho Extract is better at structuring the tables used in financial documents, resulting in better readability which seems to extend to the encodings.

![Figure 4: Accuracy for varying HR@ for two context extraction methods.](image)

### C Code Conversion

Figure 5 shows the steps of converting (a) derivation of the result in FinQA into (b) dummy Python code with dummy variable names, and finally transforming it to (c) a meaningful Python program in DocFinQA following the work by Koncel-Kedziorski et al. (2023).

(a) \[ \text{subtract}(34.8, 1.2), \text{divide}(\#0, 34.8) \]

(b) \[
    \begin{align*}
    a &= 34.8 - 1.2 \\
    b &= a/34.8 \\
    c &= b * 100
    \end{align*}
\]

(c) \[
    \begin{align*}
    \text{payments\_decrease} &= 34.8 - 1.2 \\
    \text{change} &= \text{payments\_decrease} / 34.8 \\
    \text{answer} &= \text{change} * 100
    \end{align*}
\]

Figure 5: Example of code conversion. (a) Original FinQA’s derivation. (b) Dummy Python Program (c) Meaningful Python Code in DocFinQA.

### D Model Details

In this work, we used the base models of Falcon, MPT, Llama 2, CodeLlama, and Mistral throughout our work. These models were not trained with supervised finetuning or reinforcement learning human feedback. The GPT-3.5 model employed in this study is gpt-3.5-turbo-0613 while the GPT-4 model used is gpt-4-1106-preview.

We also included the Llama 2/7B + SFT that was finetuned on the training set of DocFinQA with golden chunk from FinQA (\(e^{\text{golden}}\)). The fine-tuning process takes 3 epochs with a batch size of 32. We use the context provided by the FinQA dataset as the input due to the limited maximum token length of the model. The maximum token length is set to 2048. The model is finetuned on 8 x Nvidia A100-80GB GPUs. We use AdamW optimizer with learning rate of 2e-6. The training process takes 4 hours to complete.
E Distribution of question by question types

Figure 6 shows the distribution by question types of the dataset before (FinQA, green) and after (DocFinQA, purple) the automatic data selection.

Figure 6: Distribution of questions grouped by question types in the original FinQA and DocFinQA. The x-axis (percentage) is presented in log scale to magnify the differences between the two sets.

F Performance of retrieval methods

Figure 7 shows a pilot study comparing dense retrieval with OpenAI ADA and Sentence BERT versus sparse retrieval (BM 25) on the development set. We can see that the dense retrieval model offers a much higher hit ratio.

Figure 7: Accuracy for varying HR@ for three search methods on the development set

Figure 8 shows the prompt template with in-context learning that we used.

Figure 8: Prompt template with Top-3 context and 3-shot In-Context Learning.
**G ColBERT Finetuning**

We finetune the original ColBERT v1 model on the train set of DocFinQA. For each data point, we perform chunking and alignment to generate one golden chunk and \( n - 1 \) negative chunks. For training, we generate a list of tuples \( (qid, pid+, pid-) \), where \( qid \) refers to the question, \( pid+ \) refers to the golden chunk and \( pid- \) refers to each of the negative chunks in that document. We train the model for a total of 3 epochs and store the checkpoints at the end of each epoch. The hit rate of the Finetuned ColBERT model after each epoch on the development set is shown in Figure 9. We observe that after the first epoch, additional finetuning does not show any performance improvement. The Finetuned ColBERT model referred to in this study thus uses the weights after the first epoch of training.

![Figure 9: Hit rate of different ColBERT variants on the development set of DocFinQA.](image)

**H Few-shot Settings**

Due to the limited context length of the LLMs, the number of few-shot demonstrations and the number of chunks fed into the In-Context Learning must be optimized. We explore 3 settings of the number of few-shot examples and 4 settings of the number of chunks used as context in the query. Figure 10 shows the performance of these settings in retrieval and answered by LLama 13B and CodeLlama 13B on the development set. We see that a higher number of few-shot examples (numshot=3) yield consistently better performance compared to a lower one (numshot=1).

![Figure 10: A QA performance plot on the development set of DocFinQA for the LLama 2 13B and CodeLlama 2 13B models for each of the 12 configurations](image)
I Golden Chunk Position

Figure 11 shows the distribution of the position golden chunk with the documents. We see that most of the golden chunks appear within the first 250 chunks (approximately 125K tokens which can be fed into the newest generative models). Nonetheless, there are a substantial number of questions that the golden chunk appears beyond this threshold.

![Histogram of the position of the FinQA context in the original SEC filing that is split into chunks of size 2750.](image)

J Human Evaluation Setting

We recruited three data professionals with 4-5 years of experience working with financial documents, including but not limited to 10-K filings, to estimate human evaluation. The professionals were provided with the entire document in PDF format, maintaining the SEC’s original format for ease of reading. They were allowed to use the keyword-search feature of PDF reader applications and a simple calculator for basic arithmetic operations required for this task. On average, the professionals spent 25 minutes per question.