Context Overloading in RAG Based Systems

Muhammad Affan Naved
Hassan Ali
Talha Tariq
25100283@lums.edu.pk
25100037@lums.edu.pk
25100041@lums.edu.pk

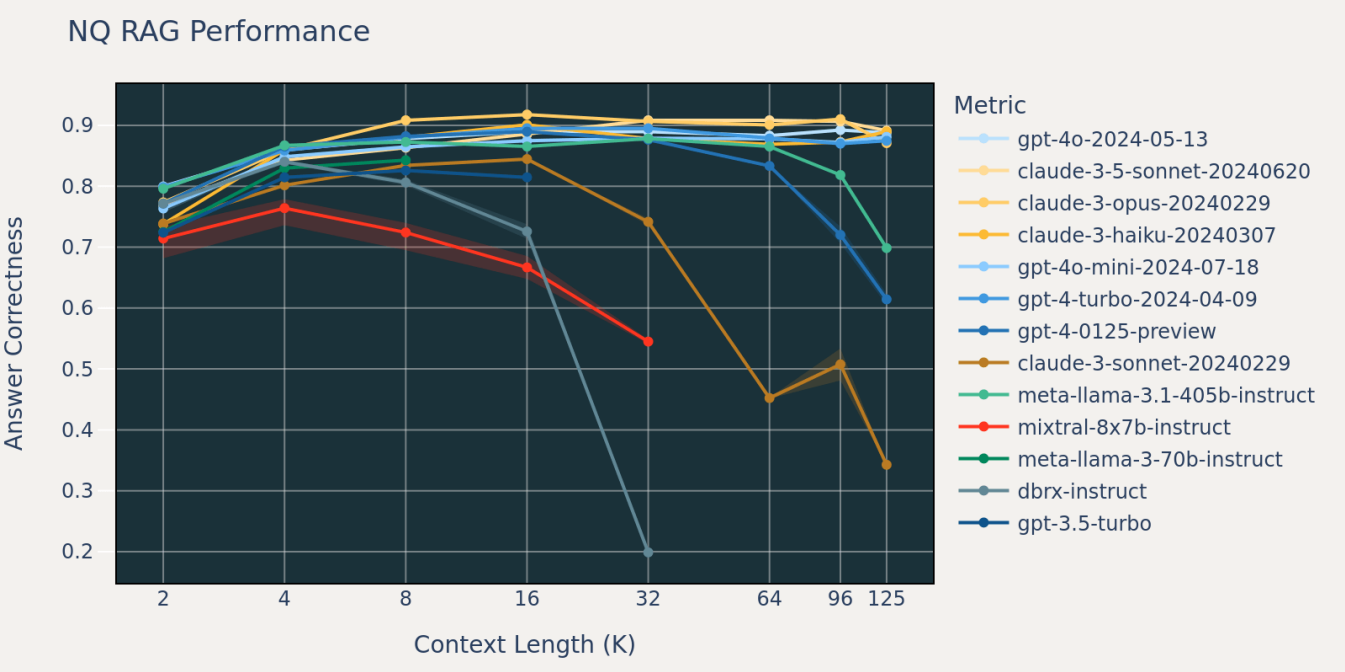

Figure 1: Image released by Databricks showing performance degradation of LLMs as context is increased. [1]

ABSTRACT

As large language models (LLMs) become increasingly capable, their ability to process and respond accurately to long-context inputs remains a significant challenge. This study investigates the performance degradation of LLMs, specifically Mistral-7B Instruct, as the context length increases, with a focus on multi-question answering tasks. We propose a two-stage methodology: an initial approach involving naive context chunking and answer interleaving, and a revised strategy using a keyword-based heuristic to pre-assign questions to the most relevant context chunks. Our experiments, conducted across varying context chunks from SQuAD dataset (K) and question counts from each chunk (Q), reveal that LLM performance declines with longer inputs, particularly beyond 1000 tokens. The revised approach demonstrated notable improvements, reducing incorrect answers by up to 20.2% compared to baseline methods. Despite limitations in compute resources and model access, our findings underscore the importance of context-aware question assignment and prompt optimization for long-context QA tasks. Future work will explore enhanced chunking strategies and evaluations across multiple LLMs in local environments.

KEYWORDS

Large Language Models (LLMs), Long-Context Processing, Question Answering, Mistral-7B, Prompt Engineering, Context Chunking, Heuristic Matching, Hallucination, Hugging Face API, Token Limitations

1 INTRODUCTION

With the rapid advancements in Natural Language Processing (NLP) and the increasing use of Large Language Models (LLMs), the capability to handle vast amounts of text and perform complex tasks, such as question answering, has become more feasible. LLMs like GPT-3, GPT-4, and specialized models such as Mistral-7B have demonstrated remarkable performance across a range of NLP tasks. However, despite these advancements, LLMs still struggle with processing long-form textual contexts effectively [1], especially when tasked with answering multiple questions based on a lengthy input. This challenge stems from the fact that as the context length increases, the model's ability to accurately retain, understand, and generate answers for the provided information often deteriorates. The performance degradation in LLMs when processing extended

contexts is primarily attributed to the inherent difficulty in maintaining context over long passages of text. Current models tend to either truncate or fail to capture essential details when the input exceeds a certain token threshold. Consequently, when asked multiple questions based on such long contexts, LLMs often generate inaccurate, incomplete, or hallucinated answers. This paper aims to address these limitations by proposing an efficient methodology for answering multiple questions from long-form contexts. Our work builds upon the observation that LLMs tend to perform better with shorter, more focused inputs. We first evaluate the performance of Mistral-7B on multi-question tasks using different context lengths, and then develop a novel strategy to optimize question-to-context matching. Specifically, we introduce a keyword-based heuristic approach for pre-processing the input, which pairs questions with the most relevant context chunks before querying the model. This ensures that the model only processes contextually relevant information for each question, thereby improving the accuracy and relevance of the answers. In the following sections, we outline the methodology and experimental setup used in this study, the results obtained from various configurations, and the insights gained from analyzing LLM behavior on long-context tasks. Our findings suggest that employing a pre-processing step significantly improves the performance of LLMs in multi-question answering tasks, and lays the groundwork for more effective strategies in handling long-form inputs in future research.

2 LITERATURE REVIEW

The paper [2] raises a question, It asks whether the LLMS which can work with long contexts (aka Long-LLMs) are really a necessity to handle long context tasks? Such as a long document Q/A. One possible replacement is to fine-tune short-context LLMs into solving long-context related problems, but the paper raises 2 concerns: first, fine-tuning is expensive, and second, it hinders the llm's ability to answer short context problems effectively.

To cater to this problem, the solution presented in the paper is built upon the principle: "the reading comprehension or summarization of a book can be solved based on the extraction of necessary key facts from the book." [2] specifically, arbitrary long-form problems can always be decomposed and solved on top of a limited memory capacity [2]. The framework they propose is: LC-Boost, which has 2 critical reasoning steps, first one is Access, where the LLM prompts itself to plan for how to access the appropriate part of context within the input. [2] and the second is Utilize, where the LLM figures out how to make effective use of the accessed context [2]. As explained in the paper, they break down a large context into smaller chunks (of 4k context length), and then from these short chunks, they aim to extract the minimal necessary context to answer the question. Through a series of experiments, the authors claim their framework outperforms traditional approaches, such as brute-force and RAG based.

To address gaps in this approach, and to discuss the problem we are addressing: the authors only focus on extracting the answer of 1 query from the entire long context. Whereas we are answering more than 10 questions whose answers are spanned throughout the large context. Furthermore, the authors did not use SQuAD dataset in their experimentation, which is the dataset we plan to use.

The paper [3] explores the challenge of handling multi-document question answering (Multi-doc QA) tasks, which require aggregating and comprehending information from numerous documents. They are difficult to solve because of noisy candidate documents, and the common problem faced by LLMs: "Lost in the middle" in which, it is claimed LLMs often overlook the context or instructions given in the middle of the input context, To address this, the paper introduces an approach called Position-Agnostic Multi-step QA (PAM QA). The core principle behind PAM QA is based on the observation that complex multi-document problems can be effectively solved by breaking them down into step-by-step processes. Specifically, instead of relying on a model to process all documents at once, PAM QA guides the model through a structured reasoning framework involving three distinct steps. First, the model repeats the question to create a context-aware prompt (Question Repetition) [3]. Second, it predicts the indexes of relevant documents within the context (Index Prediction), allowing it to focus attention on the most useful sources rather than quoting text verbatim [3]. Finally, the model formulates an answer by aggregating the necessary information, following an answer indicator to structure its response [3]. Through this decomposition, the authors claim to improve the model's ability to navigate long contexts and noisy information, enabling more accurate answers compared to traditional single-step approaches. However, it is important to note that the authors of this paper focus on extracting the answer for a single query at a time, applying their multi-step approach sequentially for individual questions. In contrast, our work aims to answer multiple questions (more than 10) from the same long context.

The paper [4] investigates why large language models (LLMs) struggle with long contexts, particularly the positional bias phenomenon where models prioritize information from the beginning or end of inputs while neglecting the middle. The researchers introduce a novel probing framework to analyze LLMs' internal representations [4]. They conduct experiments using two tasks from [5]: Key-Value Pairs Retrieval and Multi-Document Question Answering, testing three open-source models (LLaMa3-8B-Instruct, Mistral-7B-Instruct, and Gemma-7B) [4].

What they found was: LLMs can encode the position of target information within their intermediate representations (as revealed through probing classifiers) but often fail to leverage this knowledge when generating responses [4]. This reveals a disconnect between information encoding and utilization. Middle-context information requires more layers to be properly encoded [4]. The layer at which peak probing accuracy is achieved correlates with response quality [4]. Earlier peak locations correlate with higher accuracy in the model's final output, suggesting that when models identify key information in earlier layers, they're more successful at integrating it into responses.

This research contributes to our understanding of positional bias in LLMs and suggests that addressing the disconnect between information encoding and utilization could be key to improving long-context performance [4]. The paper distinguishes between "knowing" (encoding information) and "telling" (utilizing it in responses), which provides a new perspective on long-context challenges [4]. Furthermore, to incorporate into our research, we aim to split the context in such a way to place the otherwise middle section at the start of our individual chunked prompts.

The paper [6] addresses a significant challenge in using Large Language Models (LLMs): the computational cost and memory requirements when processing long documents or extended conversations [6]. When inputs exceed an LLM's fixed context length, truncation occurs, potentially losing important information.

The paper proposes the idea of "Selective Context," a method that "identifies and prunes redundancy in the input context" [6] to make it more compact while preserving performance. The technique is based on information theory, specifically using "self-information" [6] to identify which parts of text are most informative. Elements with higher self-information are considered more important to retain. The method works by:

- Using a causal language model to compute self-information for each token [6]
- Merging tokens into lexical units (phrases or sentences) [6]
- Filtering out units with lower self-information values [6]

The researchers evaluated their approach on "arXiv papers, news articles, and long conversations" [6], testing tasks including summarization, question answering, and response generation. They achieved 50% reduction in context cost, 36% reduction in inference memory usage, 32% reduction in inference time [6]. They also found that Phrase-level filtering performed better than sentence or token-level approaches [6].

3 EDA REPORT

3.1 Dataset Preprocessing

Our chosen dataset is the Stanford Question Answering Dataset (SQuAD), which is a reading comprehension dataset consisting of questions posed by crowdworkers on a set of Wikipedia articles. The answer to every question is a segment of text, or span, from the corresponding reading passage. There are 100,000+ question-answer pairs on 500+ articles. [7]

To prepare the SQuAD dataset for EDA, several preprocessing steps were applied. The dataset was originally in JSON format, which was first converted to CSV for easier handling. The data was then loaded into a pandas DataFrame, and column names were adjusted for better readability.

To ensure data quality, duplicate entries were identified and removed, followed by the elimination of missing values. Text normalization techniques were applied, including converting all text to lowercase and trimming whitespace. Additionally, stopwords were removed using NLTK, and punctuation was stripped to focus on meaningful words.

3.2 Exploratory Data Analysis

3.2.1 Length Distribution. The following table provides a statistical overview of the lengths of questions, context and answers in the SQuAD dataset.

	question_length	context_length	answer_length
count	87596.000000	87596.000000	87596.000000
mean	10.061064	119.762832	3.162233
std	3.559231	49.365597	3.392368
min	1.000000	20.000000	1.000000
25%	8.000000	89.000000	1.000000
50%	10.000000	110.000000	2.000000
75%	12.000000	142.000000	3.000000
max	40.000000	653.000000	43.000000

Figure 2: Length Distribution

3.2.2 Visualization of Lengths against Frequencies. The goal of analyzing the length distributions of questions, contexts, and answers is to understand the structure of the dataset. This helps in designing efficient retrieval and generation strategies for a RAG (Retrieval-Augmented Generation) model.

- Questions Length Distribution: Most are short (5–15 words), peaking around 8–10 words, indicating a preference for concise queries.

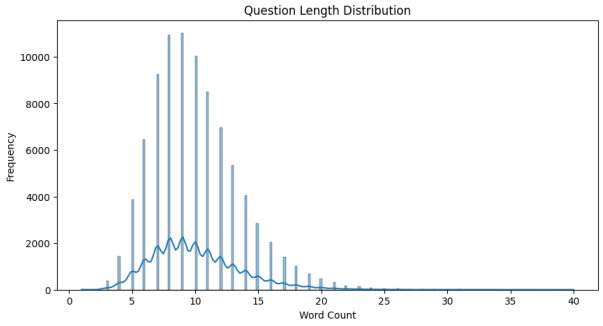

Figure 3: Question Length Distribution

- Context Length Distribution Right-skewed distribution, with most around 100–150 words but some exceeding 600 words.

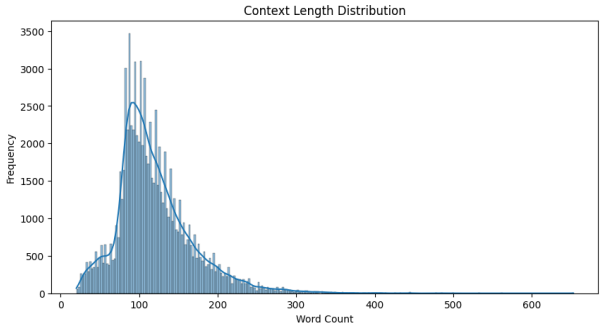

Figure 4: Context Length Distribution

- Answer Length Distribution Highly skewed towards short responses (1–5 words), suggesting fact-based, extractive answers.

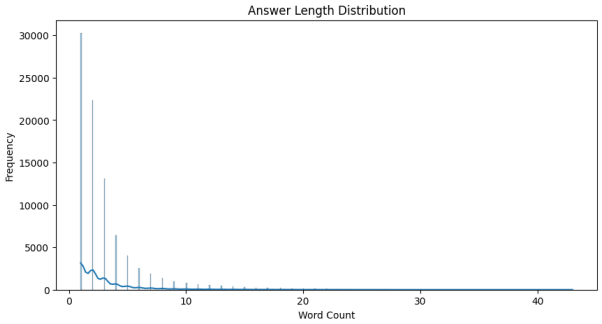

Figure 5: Answer Length Distribution

In the context of LLMs and RAG:

- Short questions suggest minimal need for restructuring, but long contexts may require chunking.
- Since contexts vary in length, efficient retrieval should prioritize extracting relevant passages rather than considering the entire text.
- Implement strategies like sliding windows or hierarchical retrieval to manage lengthy contexts without losing relevant details.
- For longer questions or answers, breaking questions into sub questions and answering those might help in generating a longer answer without needing to train the model on extra data with considerably longer contexts.

3.2.3 Frequent Word Visualisations. The purpose of these visualizations is to identify the 20 most frequently used words in the questions, contexts, and answers within the dataset. This helps understand the key themes and focus areas of the dataset and provides insights into common linguistic patterns.

- Questions: The most frequent words include many, year, first, name, type, and used, indicating that questions often focus on quantities, chronology, and categorization.

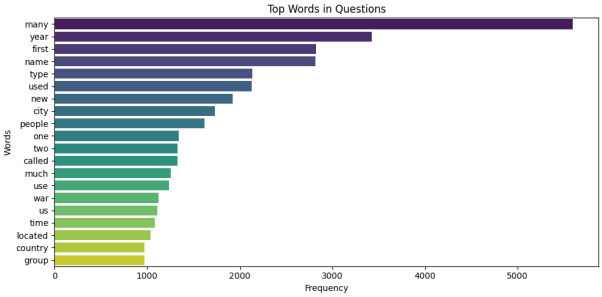

Figure 6: Top Words in Questions

- Context: Words like also, new, first, city, war, and states suggest that historical, geographical, and political topics are common in the dataset.

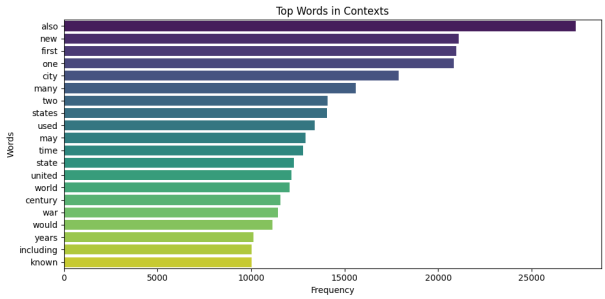

Figure 7: Top Words in Context

- Answers: The prominence of million, new, century, two, one, and united implies that numerical values, time periods, and country-related terms dominate responses.

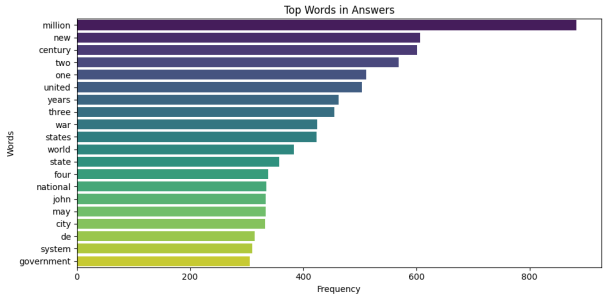

Figure 8: Top Words in Answers

3.2.4 Question Types in the Dataset. We check the questions types with the help of regex by searching for text containing who, what, when, where, why, how, which, whom, whose. We then plot their frequencies to get an idea what kind of questions are present in the dataset.

The bar plot shows that "What" questions dominate the dataset, indicating a strong focus on definitions, explanations, and identification-based queries. Other question types like "How," "Who," "Which," and "When" appear significantly less frequently, while "Where" and "Why" questions are even rarer. This suggests that the dataset contains fewer causal reasoning (why) and location-based (where) questions

In the context of Context Overloading If the model is trained on this dataset, it might perform exceptionally well on factual recall (what) but struggle with reasoning (why) or location-based retrieval (where). To ensure the model generalizes well, additional training data focusing on "Why" and "Where" questions should be introduced, or data augmentation strategies could be employed to balance question types.

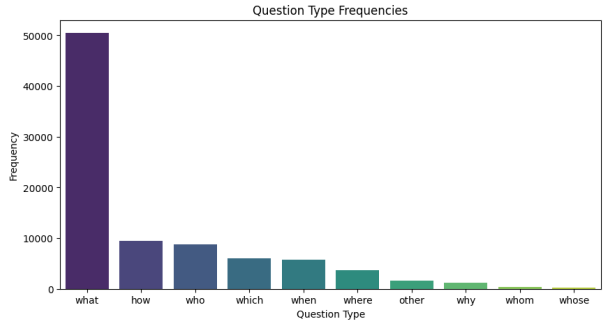

Figure 9: Question Type Frequencies

3.2.5 Length of Answers by Question Types. The results show that "whom" questions tend to have shorter answers, likely because they ask for specific names, while "whose" questions have the shortest answers overall, as they usually expect a single-word possessive response. Additionally, for "why" questions, the minimum answer length starts at a higher value, suggesting that most answers require at least a few words for meaningful context, while other question types show relatively similar distributions. We can use this as a sanity check to cross validate the answers of LLMs with their expected length, before proceeding with the finalized answers, specially for "whom", "other", "whose" and "why" question types.

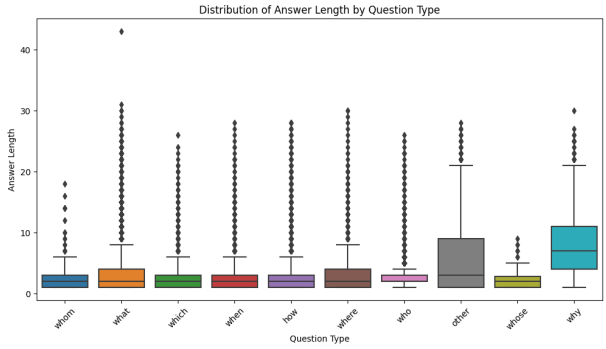

Figure 10: Length of Answers by Question Types

3.2.6 Answer Starting Positions vs Length of Context Paragraph. By computing the correlation coefficient, we aimed to determine if longer contexts tend to push answers further into the passage or if answer positions are more randomly distributed. We concluded that the correlation coefficient of 0.4129 indicates a moderate positive relationship between context length and answer start position, meaning longer contexts tend to have answers appearing later but not in a strictly linear fashion. This suggests that while longer passages increase the likelihood of later answers, other factors such as passage structure and question type also play a role.

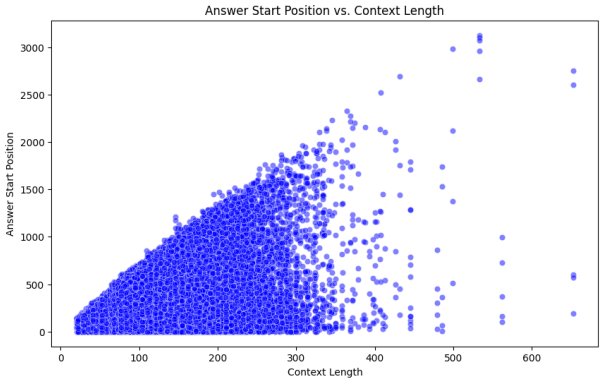

Figure 11: Answer Start Position vs. Context Length

3.2.7 Length/Complexity of Question vs Length of Context. This test examines whether context length and question length are correlated, determining if longer passages lead to longer questions. With a correlation of 0.0161, there is almost no relationship, meaning question length remains fairly independent of passage length. This suggests that questions are likely formulated based on key information rather than the overall passage size. Hence, we can discard complexity of questions being a factor when answering from longer contexts, and treat all questions as equal, in our experiments.

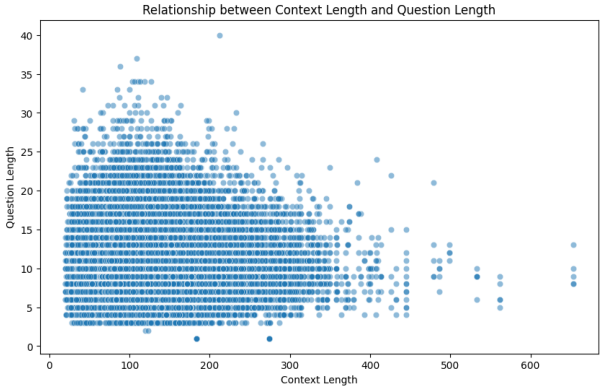

Figure 12: Relationship between Context Length and Question Length

3.2.8 Answer Start Points Analysis of All Questions. The graph shows most of the answers can be found at the beginning of the context paragraphs, which implies that like LLMs, humans also have a bias towards certain areas of the context to focus on and generate questions from. It also gives us an insight towards our implementation of context overloading. when we will be making chunks of the context to give it to smaller LLMs, it will be ideal if we chunk while separating by paragraph starting or endings.

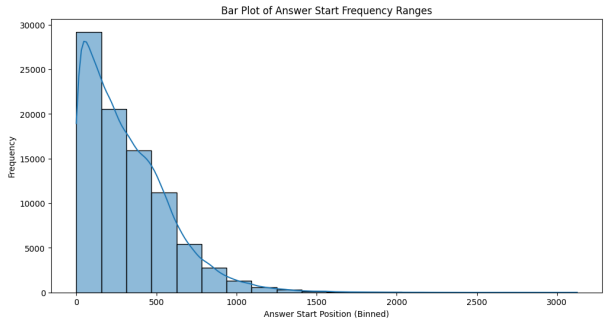

Figure 13: Answer Start Frequency Ranges

3.2.9 Evaluating Answer Overlap with Context Paragraphs. This analysis measures how much the answer text overlaps with the context using Jaccard similarity, helping to understand if answers tend to be direct extracts or paraphrased interpretations. We can see that the exponentially decreasing curve shows most answers have low overlap with their context, meaning they are often paraphrased or only partially extracted. Fully copied answers are much less common. This indicates that the dataset likely encourages comprehension-based responses rather than direct retrieval, so we will have to inform the llm accordingly

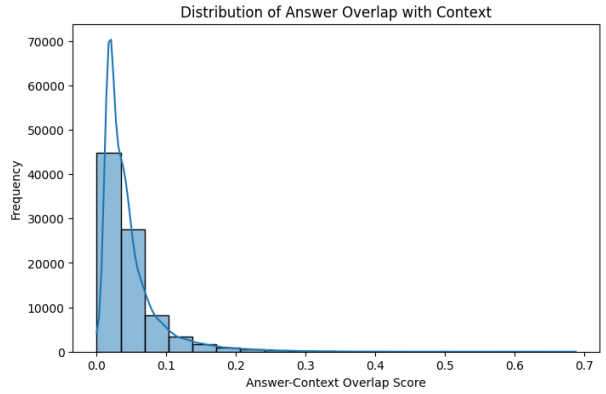

Figure 14: Answer Overlap with Context

3.2.10 Contraction Expansion in Preprocessing. Many text entries contain contractions (e.g., "I'm" → "I am", "we'll" → "we will", "it's" → "it is"). These can lead to inconsistencies in text analysis, affecting tokenization, word frequency distributions, and downstream NLP tasks. Expanding contractions ensures that all words are in their full form, making the text clearer and easier to process. We applied contraction expansion using the contractions library. This function automatically replaces shortened words with their full versions. By doing so we found that:

- Text normalization improved by reducing inconsistencies in word forms.
- Tokenization became more uniform, leading to better word frequency analysis.
- Helps in improving model interpretability by ensuring that contractions do not introduce ambiguity.

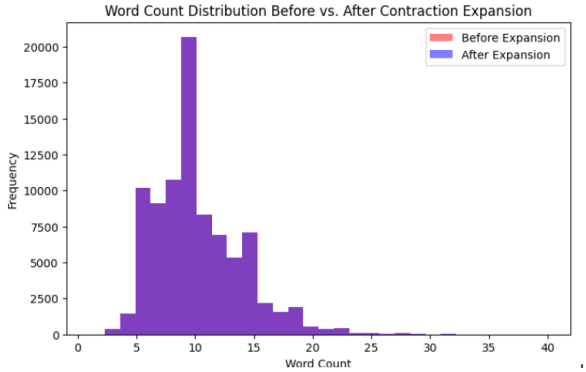

Figure 15: Enter Caption

3.2.11 Handling Numbers in Text. Since numerical values play a crucial role in understanding context, this step ensures that numbers are uniformly represented. This prevents inconsistencies when training NLP models and allows for better interpretability in downstream tasks like Named Entity Recognition (NER) and language modeling. Numbers frequently appear in text data, often in different formats (e.g., "5" vs. "five", "2020" vs. "two thousand twenty"). Inconsistent number formatting can affect tokenization, word frequency analysis, and downstream NLP tasks. To address this, we identified and processed numerical values in the dataset to ensure uniform representation. We found that:

- The dataset contains a significant number of numerical values, many of which represent years, quantities, or rankings.
- Converting numbers to words ensures better text tokenization and consistency, leading to improved word frequency analysis.
- The presence of specific dominant numbers (e.g., "one", "two", "million") suggests that many questions are related to quantities and historical references.

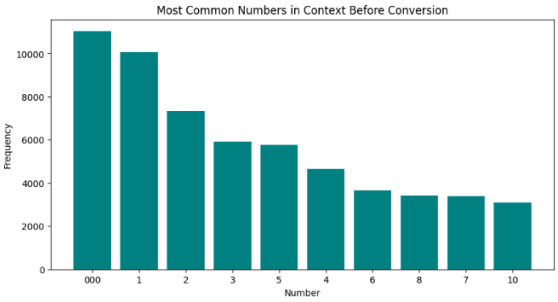

Figure 16: Enter Caption

4 PROPOSED METHODOLOGY

4.1 Experimental Setup

The first step in our research is to find the optimal and breaking context lengths of the input on which the LLM performs well vs on

those which the LLM breaks very frequently. For that, we decide to make a class to assist us in parsing through the dataset and crafting a prompt to combine all the questions (number of questions from each context "q" and number of contexts "k" will be given as function parameters) and print the number of characters and tokens and then query our LLM on that context then parse the response and present a side by side comparison of the ground truth answers and LLM responses. These are then used to compute accuracy of the number of "k" and select a better "k."

4.1.1 Loading Dataset. We created a shareable google drive link and uploaded the dataset (as csv files, which we transformed from json in the last deliverable), and used gdown and their public urls to download dataset whenever the notebook is run (since we are using google colab for that) We then read the csv into a pandas dataframe.

4.1.2 Transforming Dataset. Now, to make the dataset easier to work with for our purpose, we create a hashmap for easier access to questions for each paragraph in the dataset. The map is structured as follows: each context paragraph is treated as a key, and its value is a list of question answer pairs stored as tuples. This lets us access "q" questions from "k" paragraphs easily for our testing.

4.1.3 Agent Class. To establish our baseline model, we designed the Agent class. The class is responsible for interacting with a language model (LLM) to generate responses based on provided context and questions. The core components of the Agent class include:

- Context Map: The hashmap containing relevant datasets transformed for use with the model.
- Model: The Mistral-7B model from Hugging Face, which serves as the baseline LLM. This model is accessed via the Hugging Face API using a provided API key.
- Functionality: The class can generate responses to questions by leveraging the context map. It selects specific context and question-answer pairs, formats them into a prompt, and passes it to the model. The model's responses are then parsed to get the relevant answers and its answers are compared to the ground truth for evaluation.

4.1.4 Experiments Performed. The exact experiments we performed include varying "k" (the number of contexts used to form the prompt) and then printing the prompt, the number of characters and the number of tokens, calling the LLM, printing its response and then parsing the response to print a side by side comparison of the LLM's response and gold labels (answers). More specifically, we performed the experiments for:

- K=1, Q=3 (196 tokens)
- K=2, Q=4 (462 tokens)
- K=4, Q=5 (830 tokens)
- K=6, Q=5 (1282 tokens)
- K=8, Q=5 (1587 tokens)
- K=16, Q=5 (2256 tokens)
- K=32, Q=5 (7000 tokens)
- K=64, Q=5 (13535 tokens)

Out of the above, the last 2 gave a timeout error and we were only able to record for the remaining ones. We couldn't test out further combinations because of research limitations mentioned at the end of this document.

4.2 Evaluation Method

We stored the outputs from the experiments performed on varying values of "k" in a .txt file and we loaded answers from them. To compare, we simply parsed the "Ground Truth" and "LLM Answer" labels for each question and we checked if one was a substring of the other, both in their raw form and in their normalized form. The theory was since we need exact answers for it to qualify as a correct answer, but LLMs tend to add additional text around the correct answers, so we decided to check for substrings and it works well. For additional accuracy, we printed the list of question numbers classified as "incorrect" by this script so we can manually compare the answers ourselves.

Later, during the evaluation of the revised and final approach, we further categorized the mismatches into three distinct groups. The first category, "Actual Mistakes," included answers that were genuinely incorrect when compared to the gold-standard labels. The second category, "Partially Correct," encompassed answers that contained the key information or main keywords required, but did not exactly match the gold label—often differing in phrasing, synonyms, or the presence/absence of minor elements such as stop words.

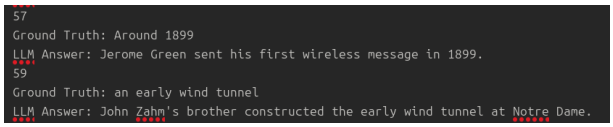

Figure 17: Example of Partially Correct Answers

Finally, we introduced the "Not Specified" category, which is specific to the revised approach. This category captures instances where questions were assigned to chunks that did not contain the relevant answers. Such cases typically arose due to limitations of the current keyword-matching heuristic used for question-to-chunk assignment.

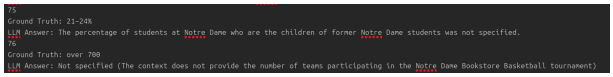

Figure 18: Example of Not specified Answers

4.3 Initial Approach to tackle context overloading

In our initial approach, the context was segmented into chunks of approximately 800 tokens each, based on insights from our initial experiments of passing the entire contexts, which indicates LLM performs better for context lengths of around 1000 tokens. All questions were then passed along with each chunk to a language model, which was instructed to answer only those questions relevant to the given chunk, leaving the others blank. Based on the nature of the responses obtained, we then decided to either interleav the

answers manually or employed a language model to perform the interleaving automatically.

```
prompt_lines = [
    "Answer these questions as precisely as you can, in as minimum words as you can",
    "Format your answers with 'A1:', 'A2:', etc. at the beginning of each answer",
    "If you cannot find enough information in the provided context to answer a question, respond with 'Not specified'. Do NOT guess.",
]
```

Figure 19: Instruction headlines for each prompt

Snippet of instructions of a prompt we passed is given as well, for reference. for interleacing answers manually, we expected to replace "Not Specified" answers with answers from other LLM calls where the question was seemingly addressed.

4.3.1 Challenges. However, the challenge we faced with this approach is that sicne we are using mimstral 7b instruct, and that too a commercial API from hugging face, It might be instruct tuned to not answer "I dont know" or "not specified" to questions from a context, as also claimed in this github issue: [8]. The LLM we are using seems to hallucinate answers to questions in an attempt to answer all of them. Furthermore, there is a lack of time and resources to properly finetune the model for our use case so it can be trained to reply "Not Specified" to questions whose context is not in the chunk.

```
A20: The First Year of Studies program at Notre Dame was recognized as outstanding by U.S. News & World Report.
A20: The First Year of Studies program at Notre Dame was declared "outstanding" by the American Council on Education.
```

Figure 20: Hallucinations by LLM

Snippet of comparison of answers of the same question from 2 different LLM calls, and hence, different context chunks, is attached and it can be seen that both answers appear to be right, given their chunked context, however, this is not the results we were expecting.

4.4 Revised approach

In the updated approach, we shifted from a post-processing strategy to a pre-processing methodology. Prior to querying the language model, we employed a keyword-based heuristic to assign each question to the most relevant context chunk. The specifics of which include firstly removing stopwords from questions and chunk and shifting all remaining keywords to lowercase. Then for each question, every chunk if evaluated based on number of matching keywords and the chunk with highest keyword is given the question. Then, individual prompts of that specific chunk and those questions are made and made to run on the LLM sequentially. Once all responses for all chunks are recieved, they are parsed for their question in the actual order and the question along with its LLM's answer and golden answer is stored in a hashmap for evaluation and printing later on.

This pre-processing step aimed to ensure that only contextually appropriate questions were paired with each chunk, thereby reducing ambiguity and improving the relevance and accuracy of the model's responses.

5 EXPERIMENTAL RESULTS

5.1 Passing the entire context

5.1.1 Hypothesis. The general hypothesis from our literature review and from past courses on LLMs is that the LLMs perform poorly on the context information or instructions hidden within the middle of the input context; we intend to test this hypothesis with our research as well.

5.1.2 Insights. As for the insights, we saw that accuracy of answers indeed decreases as we increase the context. Images are attached for reference. The images contain exact statistics for our performed experiments. K=1 was manually examined by humans and we saw that it has a 100% accuracy, so we did not include it in our graphs.

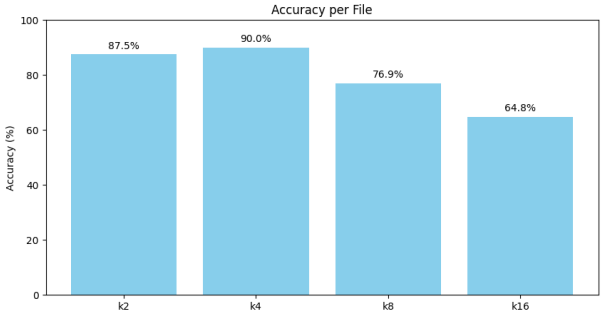

Figure 21: Accuracy per File

As for the incorrect counts per file, it seems to follow an exponential pattern, and looking at accuracy percentage and number of incorrect answers, K=4 (with context length 830) seems to be a viable context length if we intend to keep accuracy to good measures. And rounding it up, a token length of 1000 tokens is what we should be chopping off our input contexts to. And to account for questions making a part of that 1000 token as well, we plan to exactly make context chunks of 700 tokens, leaving margin for questions and instructions to make up the rest of the roughly 300 tokens.

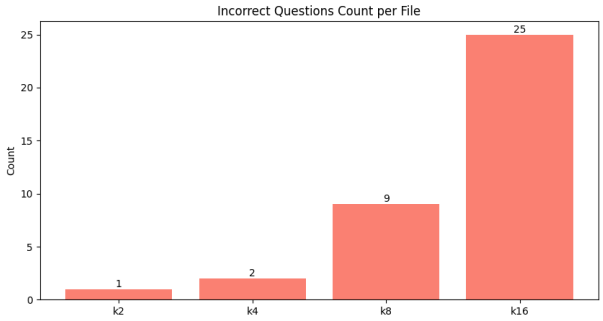

Figure 22: Incorrect Question Count per File

Another insight we got is if we look at one of the wrong questions in k=16, most of them relate to numbers being misplaced which also seems to be a common occurrence with LLMs that they don't seem to be good with numbers and calculations.

5.2 Initial Approach

The results of the initial approach were partially discussed in the challenges subsection of the methodology. To reiterate, our strategy involved submitting all questions alongside each context chunk—including those questions whose answers were not present within the chunk. After receiving the model's responses, we evaluated them to determine the feasibility of interleaving the partial answers.

Based on our observations, the Mistral 7B Instruct model consistently attempted to answer all of the provided questions, regardless of whether the relevant information was present in the context [8]. This behavior limited the effectiveness of our interleaving strategy: it was not feasible to manually or programmatically interleave the answers because the model's responses often included unsupported or fabricated answers. Furthermore, automating the interleaving process using an LLM introduced additional uncertainty, as the model lacked a reliable mechanism to distinguish between accurate and inaccurate answers.

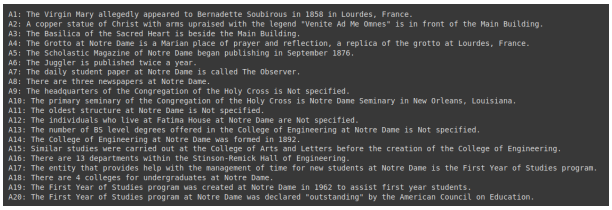

Figure 23: Snippet of results from Initial Approach

The screenshot of the LLM's response for Chunk 1 is provided above. It is evident that starting from Question A9, the answers pertain to content located in Chunk 2. We are confident in this assessment because the context was divided linearly, and the questions were also ordered sequentially. Therefore, if the answer to A9 resides in Chunk 2, it follows that subsequent questions (A9 onward) should similarly be associated with Chunk 2. This is supported by the fact that several of these responses are marked as "Not Specified." However, within this range, there are instances where the LLM still attempted to provide or fabricate answers despite the absence of relevant context. These cases highlight the core challenges of this approach and underscore why it cannot be fully relied upon in its current form.

Therefore, we decided to shift from a post-inference processing mindset to a pre-inference processing mindset, as discussed in the "Revised Approach" section of methodology.

5.3 Revised Approach

The revised approach demonstrated notable improvements in performance. We maintained the same benchmarks as in our previous experiments, specifically with K=8,Q=5 and K=16,Q=5, to allow for direct comparison of results.

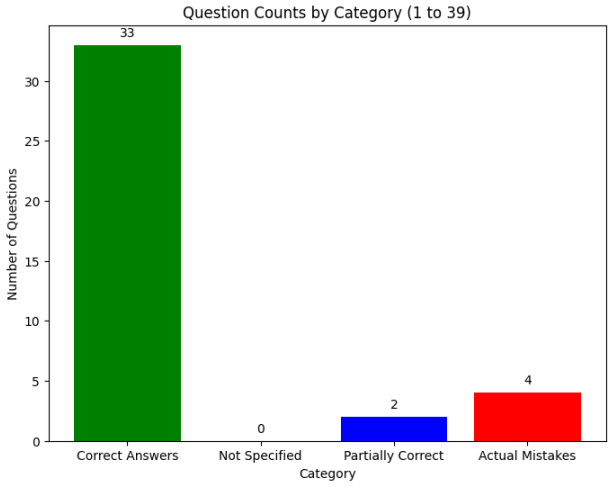

Figure 24: Results of the revised methodology on K=8, Q=5

As shown in Figure 24, for K=8, the number of incorrect answers decreased from 9 in the initial approach to 4 fully incorrect and 2 partially correct answers. This represents an improvement of approximately 12.8% over the traditional method of passing the entire context (1587 tokens).

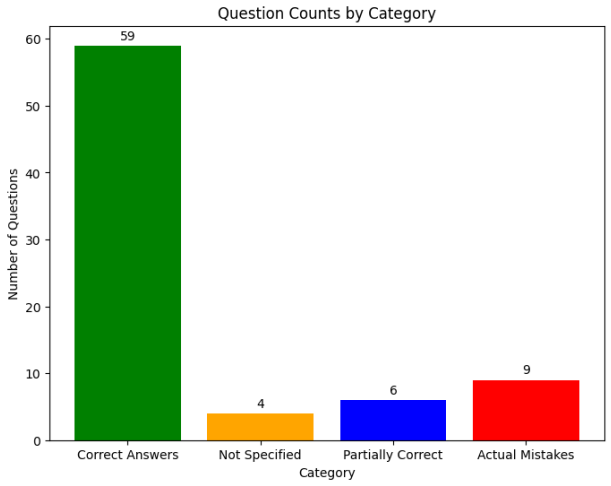

Figure 25: Results of the revised methodology on K=16, Q=5

An even more substantial improvement was observed for K=16 (2256 tokens), as illustrated in Figure 26. Previously, 25 questions were classified as incorrect; under the revised methodology, this number was reduced to 9 fully incorrect answers. This reflects an improvement of approximately 20.2% compared to the baseline method.

It is important to note that human intervention was required to classify "mismatched" answers into the three categories defined earlier. Given additional time, further refinements—such as fine-tuning the model or enhancing the question assignment algorithm—could

likely have improved the handling of partially correct or ambiguous responses.

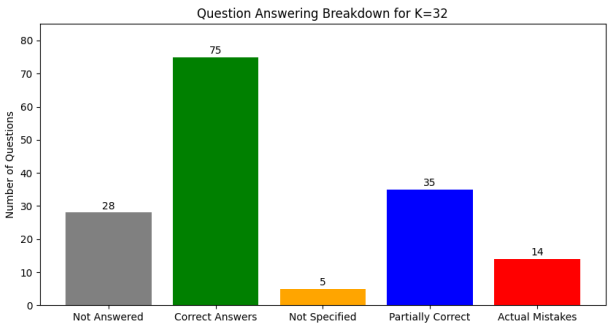

Figure 26: Results of the revised methodology on K=32, Q=5

Now coming towards the context size of interest: while experimenting, we focused on a context size of approximately 7000 tokens, which corresponded to our parameters of K=32 and Q=5. At this token length, we encountered a limitation: the Hugging Face API consistently returned a timeout error, preventing full resolution of the task.

Interestingly, our novel approach—chunking the long contexts and assigning questions to chunks based on relevance—helped mitigate this issue to some extent. We successfully conducted experiments at K=32, as shown in the graph. Although the API still timed out before processing all individual prompts (likely due to sending them all simultaneously), this is a technical constraint that could be addressed. With throttling and staggered request delays, we anticipate that responses could be received and processed successfully.

Nonetheless, as indicated in the graph, out of 157 questions, 129 were answered before the timeout occurred. The accuracy of these responses is promising; if we include the 35 partially correct responses (which did not pass substring comparisons but were contextually accurate), the effective accuracy rises to 110 out of 129—approximately 85%. This performance exceeds what we would have likely achieved had we inputted the full 157 questions along with a 7000-token prompt directly into the LLM.

Overall, the revised methodology yielded positive results and demonstrated a promising direction for mitigating performance degradation in LLMs when processing long contexts.

6 LIMITATIONS AND FURTHER WORK

Several limitations affected the scope and depth of this project. First, we faced significant time constraints, with a delivery deadline of approximately three weeks. Additionally, we relied exclusively on the Hugging Face API for a single model—`mistralai/ Mistral-7B-Instruct- v0.3`—and were limited by the available access tokens, which provided a restricted monthly credit for inference. This constraint reduced the range of experiments we were able to conduct. Moreover, we were unable to fully test the 32k-token input limits of large language models, as the API response consistently timed out after two minutes. This technical barrier also required us to adjust our original research question.

For future work, we plan to conduct experiments using locally hosted LLMs, which would allow us to explore the full capabilities of the models without API-imposed limitations. We also aim to evaluate our methodology on additional commercial LLMs, such as ChatGPT-4o, and Claude Sonnet. Furthermore, the current keyword-based heuristic for matching questions with context chunks can be enhanced through more sophisticated techniques, such as cosine similarity or the use of word embedding models, to improve precision and reliability. Moreover, the chunking strategy can be improved as well, rather than chunking on hard number of tokens, we can use semantic or agentic chunking to intelligently chunk the context of the prompt.

7 CONCLUSION

This project explored methods for efficiently answering multiple questions from long-form textual contexts using large language models (LLMs). Our initial approach involved dividing the context into smaller chunks and passing all questions along with each chunk to the model, with the aim of extracting only the relevant answers. However, we observed that the Mistral 7B Instruct model tended to answer all questions regardless of contextual relevance, limiting the effectiveness of our interleaving strategy. These findings highlighted key challenges in using off-the-shelf LLMs for selective question answering without fine-tuning or advanced prompt engineering.

To address these challenges, we proposed a revised methodology that incorporates pre-processing through keyword-based heuristics to better match questions with their relevant context chunks. While implementation was constrained by time and API limitations, the project established a foundational workflow and identified clear directions for improvement. Future work will focus on refining the question-chunk matching process and expanding testing across a broader range of LLMs in more flexible, locally hosted environments.

In summary, this project underscored both the potential and the current limitations of LLMs in handling multi-question tasks on lengthy inputs, laying the groundwork for more robust and scalable solutions.

ACKNOWLEDGMENTS

To Professor Asim Karim for the guest lecture on recent advancements around the AI space, which sparked our curiosity to tackle new emerging challenges, and their TA Faizad for continuous guidance throughout the project.

REFERENCES

[1] Databricks. How long-context rag improves performance of llms, 2024. Accessed: 2025-05-04.
[2] Hongjin Qian, Zheng Liu, Peitian Zhang, Kelong Mao, Yujia Zhou, Xu Chen, and Zhicheng Dou. Are long-llms a necessity for long-context tasks?, 2024.
[3] Junqing He, Kunhao Pan, Xiaoqun Dong, Zhuoyang Song, Yibo Liu, Qianguo Sun, Yuxin Liang, Hao Wang, Enming Zhang, and Jiaxing Zhang. Never lost in the middle: Mastering long-context question answering with position-agnostic decompositional training, 2024.
[4] Taiming Lu, Muhan Gao, Kuai Yu, Adam Byerly, and Daniel Khashabi. Insights into llm long-context failures: When transformers know but don't tell, 2024.
[5] Nelson F. Liu, Kevin Lin, John Hewitt, Ashwin Paranjape, Michele Bevilacqua, Fabio Petroni, and Percy Liang. Lost in the middle: How language models use long contexts, 2023.

[6] Yucheng Li, Bo Dong, Chenghua Lin, and Frank Guerin. Compressing context to enhance inference efficiency of large language models, 2023.
[7] Stanford University. Stanford question answering dataset (squad). https://www.kaggle.com/datasets/stanfordu/stanford-question-answering-dataset, 2016. Accessed: 2025-05-04.
[8] PromptEngineer. localgpt issue #674: Multiple questions on long context. https://github.com/PromtEngineer/localGPT/issues/674, 2024. Accessed: 2025-05-01.

Received 04 May 2025