LLM-Ref: Enhancing Reference Handling in Technical Writing with Large Language Models

Anonymous ACL submission

Abstract

Large Language Models (LLMs) are effective at synthesizing knowledge but often lack accuracy in domain-specific tasks. Retrieval-Augmented Generation (RAG) systems, utilizing user-provided data, can mitigate the issue and assist in article writing. However, such systems lack the capability to generate proper references. In this paper, we present LLM-Ref, a writing assistant tool that aids researchers in writing articles from multiple source documents with enhanced reference synthesis and handling capabilities. Unlike traditional RAG systems, which rely on chunking and indexing, LLM-Ref retrieves and generates content at the paragraph level, allowing for seamless reference extraction for the generated text. Furthermore, the tool incorporates iterative response generation to accommodate extended contexts within language model constraints while actively mitigating hallucinations. Compared to baseline RAG-based systems, our approach achieves a $1.62\times$ to $6.26\times$ increase in Ragas score, a comprehensive metric that provides a holistic view of a RAG system's ability to produce accurate, relevant, and contextually appropriate responses.

1 Introduction

Scientific research is fundamental in enriching our knowledge base, and contributing to the betterment of human lives. Precisely written research articles play a vital role in conveying new findings and innovations to a wide audience, preventing misinterpretations that might impede further developments. Writing research papers presents a challenge as it requires balancing technical complexity with readability while ensuring logical coherence. There exist writing assistant tools that utilize the latest advancements in natural language processing (NLP) to help researchers refine grammar and style, strengthen coherence, and ultimately contribute to high-quality, impactful articles.

Large Language Models (LLMs) have significantly advanced NLP applications by improving language understanding, generation, and interaction. While they excel in many NLP tasks, they require substantial computational resources and may struggle with specialized tasks without domain-specific knowledge. LLMs often produce inaccurate responses or 'hallucinations' when handling tasks beyond their training data. Developing an effective writing assistant using LLMs requires fine-tuning with domain-specific data from various fields, a process that demands extensive computational resources and a diverse dataset, making it costly to create a versatile and effective tool for diverse writing challenges.

Retrieval-Augmented Generation (RAG) (Lewis et al., 2021) systems have gained popularity as a means to alleviate the challenges involving LLMs in downstream tasks by incorporating user-provided information. Domain specific and up-to-date data integration enables language model to generate more factually accurate responses (Gao et al., 2024). RAGs preprocess external data in chunks and often utilize the top-k chunks as context for a particular query, which plays a crucial role in output generation. In research articles, the trade-off between chunk size and top-k context selection often leads to the omission of important nuanced information. Since the data is not chunked in a structured manner, it hinders the retrieval of citations present in the context.

In this paper, we present LLM-Ref, a writing assistant tool that helps researchers with enhanced reference extraction while writing articles based on multiple source documents. To preserve the citations in the context paragraphs, our writing assistant tool preserves the hierarchical section-subsection structure of source documents. Rather than dividing texts into chunks and transforming them into embeddings, our approach directly utilizes the paragraphs from research articles to identify information relevant to specific queries. To efficiently retrieve all the relevant information from the source documents, an LLM is utilized due to its superior performance in finding semantic relevance. Furthermore, iterative generation of output allows handling long context and accurate responses. Efficient retrieval and preservation of hierarchical source information enable the listing of comprehensive references, ensuring that users have access to detailed citation details. The proposed LLM-Ref can provide both primary references—the source documents—and secondary references, which are listed in the context paragraphs of the source documents. To the best of our knowledge, no other similar work focuses on providing both primary and secondary references.

Evaluation results show the superior performance of our tool over existing RAG-based systems. The proposed LLM-Ref demonstrates significant performance improvements over other RAG systems, achieving a $5.5\times$ higher Context Relevancy compared to Basic RAG. Additionally, it delivers an impressive increase in the Ragas Score, outperforming the best alternative by around $2\times$. These results highlight that the proposed tool provides more accurate, relevant, and contextually precise outputs, enhancing the overall utility and reliability of the writing assistance it offers. The contributions of this paper include:

- We develop LLM-Ref, an advanced system that employs a hierarchical document processing approach, setting it apart from traditional RAGs while outperforming them.
- A key innovation of LLM-Ref is its ability to generate both primary and secondary references, a functionality absent in traditional systems.
- LLM-Ref's generation mechanism adopts an iterative approach, effectively managing long contexts while minimizing hallucinations through source-context alignment.

2 Background and Related Works

Large Language Models (LLMs) like ChatGPT (OpenAI, 2023; Brown et al., 2020) and LLaMa (Touvron et al., 2023) have propelled the landscape of natural language processing (NLP) (Bubeck et al., 2023; Hendrycks et al., 2021; Srivastava et al., 2023), leveraging vast amounts of data to understand, generate, and interact with human language in a deeply nuanced and contextually aware manner. However, this remarkable performance of LLMs incurs huge computational costs to train the several billions of parameters of the model on enormous amounts of data (Kaddour et al., 2023). Moreover, unless fine-tuned for domain-specific downstream tasks, the performance of LLMs degrades notably (Kandpal et al., 2023; Gao et al., 2024). Being transformer-based models (Vaswani et al., 2023), LLMs have restrictions on how much input context they can utilize for response generation which affects the quality of the output. Conversely, LLMs with long context lengths fail to relate the content in the middle. Compounding the challenges, LLMs exhibit 'hallucinations' when tasks require up-to-date information that extends beyond their training data (Zhang et al., 2023; Kandpal et al., 2023; Gao et al., 2024). These drawbacks often complicate developing custom downstream applications with LLMs.

Retrieval-Augmented Generation (RAG) tackles the problem of "hallucinations" by pulling in real facts from an external knowledge base as it writes (Lewis et al., 2021). First, RAG builds an index—cleaning and tokenizing text to make it searchable. Next, it uses a semantic retriever to find the most relevant passages for your query. Finally, the generator utilizes those contexts with the original question to produce a concise, accurate answer that can still offer fresh insights.

Building on RAG, recent LLM-based methods introduce novel solutions for long-context management, query refinement, and content distillation. MemWalker (Chen et al., 2023) builds a memory tree over segmented text to overcome context-window limits and support long-range querying, while KnowledGPT (Wang et al., 2023) and Rewrite-Retrieve-Read (Ma et al., 2023) programmatically rewrite queries to better capture user intent—though multi-hop questions still suffer error propagation. To hone in on the most relevant information, PRCA uses domain-specific abstractive summarization (Yang et al., 2023), FiD-light (Hofstätter et al., 2022) applies listwise autoregressive re-ranking that links generated passages back to their sources, and RECOMP (Xu et al., 2023) compresses retrieved content into concise summaries for efficient generation (Xu et al., 2023). Together, these advances make the retrieval-generation loop more coherent and accurate, yet none of them address the crucial challenge of reference handling.

Citation generation is the task of having an LLM answer a question while clearly showing which text passages it used as citations, helping to close the reference gap. ALCE uses in-context learning on the top 100-word snippets to prompt LLMs to answer questions with cited support, then benchmarks fluency, correctness, and citation quality—showing that models often undercite when synthesizing multiple sources (Gao et al., 2023). To boost citation accuracy, Chain-of-Thought prompting guides the model through explicit reasoning steps and a Citation Insurance Mechanism spots and corrects missing references, though this adds complexity and compute overhead (Ji et al., 2024). Citation-Enhanced Generation (CEG) takes a post-hoc approach: it retrieves documents after an initial answer, uses natural language inference to verify factual claims, and asks the model to regenerate any unsupported text (Li et al., 2024). Evaluations by Byun et al. reveal that GPT-4 surpasses earlier models in citation accuracy—but not relevance—and performs better on NLP than HCI papers (Byun et al., 2024). Citekit (Shen et al., 2024) includes modules for loading data, generating citations, enhancing citation quality, and evaluating results. The toolkit simplifies comparing existing citation-generation methods and facilitates developing new ones. Experiments demonstrate its effectiveness, though challenges remain in achieving fine-grained citations and consistently balancing citation quality with answer accuracy. Existing works rely on chunks of contexts, whereas LLM-Ref feeds the contexts in the exact source paragraph, allowing the extraction of the citations accurately. Such context in paragraphs is paramount in references (citations) extraction since chunking often makes erroneous or missing citations.

GPT-based models are highly effective at paraphrasing, and grammar correction, and also excel in crafting informative paragraphs suitable for research papers. Current popular LLMs can conduct question-answering tasks using user-provided data, marking a significant advancement in its functionality. Despite supporting multiple user files as inputs, most of them do not return the specific context utilized in the generation process nor do they offer comprehensive secondary references. While commercial tools exist to assist researchers, the lack of sufficient documentation limits transparency and prevents direct evaluation against LLM-Ref. The implementation of our method is open-source and available at [GitHub][1]for reproducibility and further research. LLM-Ref is a novel system designed to enhance document processing through a hierarchical approach. A significant contribution of LLM-Ref is its capacity to generate both primary and secondary references, a capability lacking in existing systems. Furthermore, its iterative generation process improves long-context handling and reduces hallucinations by aligning generated content with source contexts, making it a robust and reliable solution for reference synthesis.

3 Architecture of Proposed LLM-Ref

In this section, we describe LLM-Ref, a writing tool designed to assist researchers by providing enhanced reference synthesis and handling capabilities, while synthesizing responses based on the information found within the context of provided research articles. Most RAG-based systems face challenges in the retrieval of relevant and correct input contexts and do not provide primary sources or secondary references when synthesizing results from multiple source documents. In contrast, LLM-Ref extracts a hierarchical structure of the contents from the source documents and provides appropriate references with the synthesized output. The architecture of the system is shown in Figure 1.

A research article is typically structured into sections and subsections to present and elucidate a particular problem, background information, and analysis. Within a section or subsection, each paragraph conveys a specific intention. To develop a writing assistant for research articles, it is crucial to extract source content efficiently with proper hierarchy. Given this, the proposed LLM-Ref begins with ① *Content Extractor* that extracts texts and the reference list from the source documents, ensuring that the original organization of the paragraphs is kept intact. It stores information from each document, including summaries of paragraphs generated by an LLM, in an offline repository. For any particular query, ② *Context Retrieval* finds the most relevant sections of texts using an LLM. A specialized component, ③ *Iterative Output Synthesizer* then processes these contexts with the corresponding query, using a language model to generate text based on the given input and predefined context length. In the final step, accurate citations of the synthesized output are extracted from the contexts by ④ *Reference Extractor*. All the prompts utilized in our work are given in the Appendix A.5.

[1]https://github.com/dummy-anonymous-git/

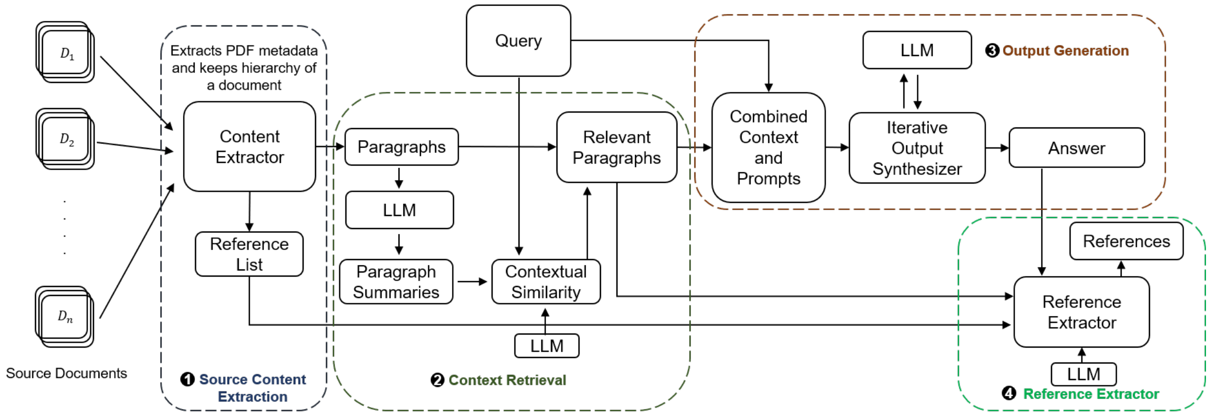

Figure 1: The overall architecture of the LLM-Ref. ① *Content Extractor* extracts texts and references of each source documents. Each article metadata along with respective paragraph summaries extracted from an LLM are stored offline. For a given *query*, in ② *Context Retrieval*, relevant paragraphs are extracted and combined with prompts to generate answers. The ③ *Iterative Output Synthesizer* feeds the combined prompt and context to an LLM for output text generation based on context length limit. Finally, the ④ *Reference Extractor* extracts respective references for output text from relevant paragraphs. Formal definition of LLM-Ref is presented in the Appendix A.2.

3.1 Source Content Extraction

RAG systems typically process source documents as discrete chunks, often disregarding section- and subsection-level abstraction. Capturing this abstraction requires machine learning-based classification and segmentation trained on domain-specific research article datasets. While identifying sections and subsections is challenging, the standardized styles and formats of research articles provide valuable cues for document hierarchy. Therefore, we utilize text formatting to infer the structural abstraction of the source documents.

Our text extractor, *Content Extractor*, reads each source file and extracts its contents while maintaining the abstraction of the content flow, utilizing the Python library `pdfminer`. This library offers fine-grained access to most content objects, allowing the *Content Extractor* to understand the research writing template. First, *Content Extractor* extracts the page layout and font-related statistics from all the pages in a document to identify article formatting details, such as the number of columns and font attributes (name, size, and style). Section and subsection labels are identified by searching for common keywords like ‘Introduction’, ‘Abstract’, ‘References’, ‘2.1’, ‘3.1’, ‘4.1’, ‘a.’, ‘(a)’, etc. However, keyword searching alone is not sufficient to accurately position and extract sections or subsections due to multiple possible instances of same section or subsection name. For precise positioning and extraction, we verify the position and text details of each search item against the formatting details initially acquired. Once the sections and subsections labels are accurately extracted, the text organized in paragraphs is extracted. To identify paragraph separation, we leverage indentation, line spacing, and column information. Thus, we store paragraphs within each section and subsection, preserving the correct abstraction.

In general, RAGs process and store documents by dividing them into chunks and applying embeddings. These embeddings are indexed and later used to retrieve relevant chunks through a similarity operation that compares the input chunks with the query. On the contrary, in our approach, we store source information offline in existing paragraphs. To retrieve relevant context, we additionally store concise and informative summaries of each paragraph which are used in the retrieval stage. However, we utilize corresponding original paragraphs for output generation and reference extraction.

3.2 Context Retrieval

In conventional RAG systems, effective text chunking is essential for transforming text segments into vector embeddings for similarity-based retrieval, ensuring both accuracy and relevance to a given query. The effectiveness of chunking depends on content type, embedding model specifications, query complexity, and application needs. Chunk size, if too large or too small, can greatly impact retrieval accuracy and system efficiency.

To mitigate the existing challenges in the retrieval stage, we perform contextual similarity between the query and the summarized paragraphs of the source documents using an LLM. The prompt consists of the user query and a paragraph from a source document. Once the relevant paragraphs are identified using the corresponding summaries, the original paragraphs are selected and fed as context for the output generation step. In our experiments, LLM-based contextual similarity performs better than embedding-based approaches due to their superior performance in understanding underlying context. Although overlapping or sliding window-based large chunking positively affects retrieving contexts, LLM-based contextual similarity on paragraphs has a better outcome on output generation and reference extraction. Using paragraphs as context can be challenging due to the LLM's context length limitations, a problem we mitigate with our iterative output generation step should it arise.

3.3 Output Generation

In the output generation step, the query and the relevant context paragraphs are combined and fed to the LLM. Usually, it is observed that research paper-related queries tend to have many context paragraphs which often do not fit within the context limit of the LLM. Moreover, LLM suffers from the 'Lost in the Middle' phenomenon when the context is too long. To address these issues, the *Iterative Output Synthesizer* is capable of synthesizing responses iteratively by processing input paragraphs and ensuring they fit within the context limit of the language model. Initially, the unit feeds the first paragraph (as context) along with the query to an LLM to generate output. The response from the LLM is then continuously updated with the rest of the relevant paragraphs. While the system generates output through continuous updates, it enforces the context limit by monitoring the size of the query, the paragraphs, and the response.

3.4 Reference Extraction

Despite the popularity, RAG-based systems fall short in offering references. While the popular commercial LLMs now has the capability to process user data, it does not provide definite necessary contexts or references that are essential for academic research. LLM-ref extracts the references from input context paragraphs. Our system adeptly identifies the source documents, referred to as 'primary references', along with the citations found within the source context paragraphs, which we term 'secondary references'. During the generation phase, LLMs omit citation notations, posing challenges in reference extraction. So our system adopts two presentations of references: Coarse-grain references for broader citation identification and Fine-grain references for more detailed citation tracking. Most research papers use either 'enumerated' (e.g., '[1]', '[2-5]', '[3,9]') or 'named' (e.g., '(Author name et al., 2024)') reference notations and our reference extractor is adept at recognizing both types.

3.4.1 Coarse-grain References

In coarse-grain reference extraction, the *Reference Extractor* catalogs all the references identified within the contexts. As contexts are extracted as paragraphs containing information relevant to the queries, this approach offers a comprehensive overview of a specific issue. The tool enumerates all the source papers and secondary references found within these context paragraphs, enabling users to analyze the referenced materials.

3.4.2 Fine-grain References

In fine-grain reference extraction, the *Reference Extractor* meticulously identifies the context lines most relevant to each line in the output text with the help of an LLM. This method of pinpointing the most similar context lines enables us to discover more specific references, thus achieving greater precision in our reference extraction process. We determine the highest relevance between response lines and source context lines using an LLM. By identifying the most relevant source contexts, we can extract primary and secondary references with high precision. This process facilitates the rapid compilation of synthesized outputs from a multitude of source documents.

4 Experimental Setup

4.1 Evaluating RAG Approaches

Our evaluation compares LLM-Ref with four other RAG implementations: Basic RAG (Lewis et al., 2021), Parent-Document Retriever (PDR) RAG (LangChain, 2023c), Ensemble RAG (LangChain, 2023a), and RAPTOR (Sarthi et al., 2024), highlighting their methodologies and applications.

RAG systems divide documents into chunks, embed those chunks, and store them in a vector database so a retriever can pull back the most relevant pieces for a language model to craft an answer. PDR RAG builds on this by organizing content into larger "parent" chunks and smaller "child" chunks;

a ParentDocumentRetriever first finds the right parents and then grabs matching children for tighter, more focused context. Ensemble RAG goes further by running multiple retrievers—like BM25 for exact keyword hits and a vector-based model for semantic matches—and then combines their results so the language model gets a richer, more robust set of passages to handle complex queries. RAPTOR builds a hierarchical tree by recursively clustering text chunks and generating summaries, capturing both detailed and high-level context for long documents. At inference, it retrieves the most relevant nodes across these abstraction levels, enabling more coherent and accurate question answering.

4.2 Dataset

The evaluation of systems similar to RAG necessitates human-annotated ground truth answers for a variety of questions, a requirement that proves difficult to fulfill across multiple domains. To address this challenge, Ragas (Es et al., 2023) and ARES (Saad-Falcon et al., 2023) employ datasets generated by ChatGPT as ground truth from specific documents. We follow this approach by leveraging GPT-4, simulating an advanced researcher, to create research question-answer-context pairs based on the provided source documents. These generated question-answer-context pairs serve as a benchmark to assess the relevance and accuracy of contexts retrieved and outputs generated by RAG, facilitating a comprehensive analysis of evaluation metrics in conjunction with Ragas.

To evaluate our system on domain-specific tasks, we curated a diverse arXiv dataset with question-answer-context pairs from Physics, Mathematics, Computer Science, Quantitative Finance, Electrical Engineering and Systems Science, and Economics. The dataset contains 955 question-answer-context pairs derived from multiple documents within the same subject area. All question–answer–context pairs were created and reviewed under strict human oversight to ensure that each answer logically aligns with its supporting context and is free of hallucinations or formatting errors. For every source document, we extracted 5–8 pairs, and any sample that did not match our prompt or answer templates was discarded—about 20% of the data was removed in this process. Annotators checked all entries for consistency, while the NLP subset received an extra layer of scrutiny by the authors themselves because of their domain expertise.

4.3 Evaluation Metrics

We employ the Ragas (Es et al., 2023) framework to evaluate the performance of the RAG systems. *Faithfulness* ensures the generated response is based on the provided input context, avoiding false or misleading information ('hallucinations'). It is crucial for transparency and accuracy, ensuring the context serves as solid evidence for the answer.

Answer Relevance measures how well the generated response directly addresses the question, ensuring responses are on-topic and accurately meet the query's requirements. *Answer Similarity* measures how closely the generated answer aligns with the ground truth in both content and intent, reflecting the RAG system's understanding of the concepts and context (Es et al., 2023).

Context Relevance ensures the retrieved context is precise and minimizes irrelevant content, which is crucial due to the costs and inefficiencies associated with processing lengthy passages through LLMs, especially when key information is buried in the middle (Liu et al., 2023). *Context Precision* gauges the system's ability to prioritize relevant items, ensuring that the most pertinent information is presented first and distinguishing it from irrelevant data. *Context Recall* measures the model's ability to retrieve all relevant information, balancing true positives against false negatives, to ensure no key details are missed. (Es et al., 2023).

The Ragas score combines key metrics: faithfulness, answer relevancy, context relevancy, and context recall (LangChain, 2023b). By integrating these metrics, the Ragas score provides a holistic view of a RAG system's ability to produce accurate, relevant, and contextually appropriate responses, guiding improvements for enhanced performance. A comprehensive explanation of the calculations is provided in the Appendix A.6.

5 Results and Analysis

5.1 Metric Analysis

We first present the performance metrics of LLM-Ref compared to Basic RAG, PDR RAG, Ens. RAG, and RAPTOR using GPT-3.5 as the LLM in Table 1. LLM-Ref significantly outperforms five of the seven metrics, performs similarly in the remaining two, and achieves an overall superior performance in the Ragas Score.

During evaluation with the Ragas framework, LLM-Ref consistently outperforms the other methods across most metrics, demonstrating its superior

Name	Answer Relevancy	Answer Correctness	Answer Similarity	Context Relevancy	Context Precision	Context Recall	Faith fulness	Ragas Score
Basic RAG	0.598	0.448	0.892	0.049	0.857	0.697	0.547	0.158
PDR RAG	0.575	0.458	0.896	0.023	0.852	0.716	0.622	0.082
Ens. RAG	0.613	0.459	0.905	0.043	0.851	**0.717**	0.600	0.143
RAPTOR	0.688	0.535	0.930	0.125	0.91	0.614	**0.64**	0.316
LLM-Ref	**0.948**	**0.568**	**0.942**	**0.268**	**0.976**	0.705	0.629	**0.513**

Table 1: Metric evaluation result comparison of LLM-Ref with Basic RAG, Parent Document Retriever RAG, Ensemble Retrieval RAG, and RAPTOR, using GPT-3.5 as the LLM. Higher values indicate better performance. The highest scores are highlighted in bold. Additional results for GPT-4o-mini, Llama, and Claude models are given in Table 5.

performance in terms of accuracy and relevance. It achieves an Answer Relevancy score of 0.948, substantially higher than Basic RAG (0.598), PDR RAG (0.575), Ens. RAG (0.613), and RAPTOR (0.688), indicating its effectiveness in providing pertinent and aligned answers to the questions. Its Answer Correctness is 0.568, surpassing others ranging from 0.448 to 0.535, demonstrating better accuracy. LLM-Ref also attains the highest Answer Similarity score of 0.942 compared to others between 0.892 and 0.930. These metrics based on the final responses demonstrate the superior efficacy of LLM-Ref in generating answers that are well-aligned with the queries and underlying intent. For Context Relevancy and Precision, LLM-Ref scores 0.268 and 0.976 respectively, are significantly higher than the other methods, which indicates its exceptional ability to retrieve and utilize relevant information. While Context Recall scores are similar across all methods, LLM-Ref achieves the competitive Faithfulness score at 0.629, showing that its answers are well-grounded in the provided context. The composite Ragas Score for LLM-Ref is 0.513, notably higher than Basic RAG (0.158), PDR RAG (0.082), Ens. RAG (0.143), and RAPTOR (0.316), highlighting its overall effectiveness in synthesizing responses for research articles. LLM-Ref outperforms other RAG systems by retrieving more relevant information, providing precise context, and delivering accurate, consistent, and high-quality responses.

5.2 Performance across LLMs

The proposed LLM-Ref method outperforms baseline methods in terms of accuracy, as demonstrated by the Ragas scores in Table 2. Across various large language models (LLMs)—GPT-3.5, GPT-4o-mini, Llama 3.1-405b, and Claude 3.5 Sonnet—LLM-Ref consistently achieves the highest scores. For instance, with GPT-4o-mini, LLM-Ref records a Ragas score of 0.413, substantially higher than Basic RAG (0.138), PDR RAG (0.112), Ens. RAG (0.096), and RAPTOR (0.299). Similar trends are observed across all LLMs, with LLM-Ref maintaining a lead.

Method	GPT 3.5	GPT 4o-mini	Llama 3.1-405b	Claude 3.5 Sonnet
Basic RAG	0.143	0.138	0.114	0.142
PDR RAG	0.052	0.112	0.079	0.055
Ens. RAG	0.097	0.096	0.105	0.096
RAPTOR	0.319	0.299	0.198	0.245
LLM-Ref	**0.389**	**0.413**	**0.300**	**0.422**

Table 2: Ragas score comparison for LLM-Ref across various LLMs.

These results underscore LLM-Ref's ability to generate accurate, contextually relevant outputs essential for research article writing. By integrating paragraph summarization, context extraction, and reference alignment, it addresses the limitations of standard RAG systems reliant solely on embeddings. The improved accuracy justifies its higher computational cost, proving its superiority for precision-critical applications.

5.3 Computational Cost

LLM-Ref prioritizes accuracy in research article writing, resulting in higher computational costs compared to traditional Retrieval-Augmented Generation (RAG) systems. These costs are primarily attributed to four stages: content extraction, context retrieval, response generation, and reference validation. Each step involves calls to large language models (LLMs), with overhead during content summarization and reference extraction, both of which are unique to our approach. While this increases input and output tokens due to summarization and similarity search, it ensures enhanced precision and relevance in the generated outputs.

Query: Pros and Cons of Discrete vs Cascaded Systems

Answer: Discrete units in direct speech-to-speech translation (S2ST) systems offer several advantages over cascaded systems [1-8]. Firstly, they have lower computational costs and inference latency because they require fewer decoding steps [1, 8, 10-12]. This makes them more efficient and faster in translating speech [1-3, 7, 8, 10-12]. Additionally, direct S2ST is a natural approach for languages that do not have a writing system, as it eliminates the need for intermediate text representation. Another benefit of direct S2ST systems with discrete units is their flexibility in training. They can be trained with or without text transcripts, making them suitable for both written and unwritten languages [1, 6, 7, 9, 13]. . . .

References:

[1] A. Lee et al., "Direct speech-to-speech translation with discrete units," arXiv, 2021.
[2] A. Lee et al., "Textless speech-to-speech translation on real data," arXiv, 2021.
[3] S. Popuri et al., "Enhanced Direct Speech-to-Speech Translation Using Self-supervised Pre-training and Data Augmentation," arXiv, 2022.
[4] Ye Jia et al., Direct speech-to-speech translation with a sequence-to-sequence model. Proc. Interspeech 2019.
. . .
[13] A. Tjandra et al., "Speech-to-speech translation between untranscribed unknown languages." In 2019 IEEE Automatic Speech Recognition and Understanding Workshop (ASRU), 2019.

Figure 2: Fine-grained reference samples generated by LLM-Ref when GPT-3.5 is used as the LLM.

Table 3 highlights the comparative cost, Ragas score, and token usage for LLM-Ref and baseline systems for 150 queries of the computer science subset. While LLM-Ref incurs higher computational costs than existing RAG systems, the expense is justified by its ability to minimize human labor and streamline academic writing. Considering that research expenses run into thousands of dollars, LLM-Ref remains affordable. By generating precise, contextually relevant responses, LLM-Ref significantly eases researchers' workload, making it a valuable tool in scholarly writing. Appendix A.3.2 details the computational cost of each LLM-Ref stage, while Appendices A.3 and A.4 provide additional analyses with different LLMs.

	Cost ($)	Ragas Score	Input Tokens	Output Tokens
Basic RAG	0.06	0.14	359,115	11,562
PDR RAG	0.08	0.11	589,582	13,911
Ens. RAG	0.09	0.10	569,531	13,885
RAPTOR	0.05	0.30	185,337	35,050
LLM-Ref	1.79	0.42	5,430,489	156,222

Table 3: Comparison of cost, Ragas score, input tokens, and output tokens when GPT-4o-mini is used as the LLM.

5.4 Reference Extraction

To demonstrate the effective functionality of LLM-Ref, we present a sample of the fine-grained references in Figure 2. For a specific query, LLM-Ref successfully generates fine-grained references, which include both enumerated references such as '[11, 12]' and named references such as '(Jia et al., 2021)'. This capability highlights the system's ability to seamlessly integrate both numerical and textual citation styles, ensuring compatibility with diverse referencing standards used across academic disciplines. For improved clarity and presentation, we organize all references in an enumerated format in the figure.

In this example, we utilize three primary source documents to generate the response. References '[1]', '[2]', and '[3]' correspond to the primary sources directly informing the response. Additionally, the secondary references, ranging from '[4]' to '[13]', are citations found within the primary sources themselves. By integrating primary and secondary references, LLM-Ref ensures a traceable foundation for responses and emphasizes its use-case for in-depth source synthesis.

More examples are presented in Appendix A.7 that showcase LLM-Ref's ability to consistently identify and organize fine-grained references across LLM architectures and its model-agnostic nature.

6 Conclusion

We present a novel writing assistant that can assist researchers in the extraction of relevant references while synthesizing information from source documents. The proposed system can alleviate the challenging optimization required in RAGs and generate output responses effectively. Moreover, our system can list primary and secondary references to assist researchers where in paying more attention to literature investigation. We intend to explore the opportunities of offline open-source LLMs to build a more flexible system in the future.

7 Limitations and Ethical Considerations

Our contribution to this work begins with the PDF file reading component, the Content Extractor, which is designed to handle the most common template styles of research articles. The extraction process is based on various heuristics; however, our *Content Extractor* may not efficiently handle all template styles. Extracting references, particularly reference lists, presents challenges that limit the support capabilities of LLM-Ref. We extract reference lists and store them with their identifiers in the texts. Our system has been tested with various research paper templates, including IEEE, ACL, and many arXiv formats. It has demonstrated proficiency in successfully extracting context, especially when reference styles are enumerated (e.g., [1], [2], [4, 28]) or named (author et al., year). We acknowledge that the Content Extractor's reliance on heuristics may not cover every possible template. While a heuristic-based framework cannot process all template variations, our experiments show that LLM-Ref generates effective results whenever we successfully extract the relevant paragraphs. To support broader applicability, we will open-source our code and provide guidance, enabling users to tweak a few straightforward heuristics to accommodate most templates commonly used in the research community. Moreover, we developed this writing assistant tool primarily to guide researchers in exploring different aspects of research, rather than to enable the writing of a research article overnight without in-depth investigation. Both our coarse-grain and fine-grain reference extraction methods can guide researchers on where to focus their efforts more intensively.

In this paper, we present the evaluation of our system using GPT models. Additionally, we apply our writing assistant tool to the Llama and Claude models, demonstrating similar results, which underscores the efficacy of our approach across a broad range of LLMs. We plan to extend our comprehensive evaluation of the tool across diverse domain-specific research articles, utilizing open-source Large Language Models (LLMs). Given that LLM-Ref leverages the LLM API, mitigating model bias poses a significant challenge. To minimize potential bias in responses, several measures have been implemented. Specifically, when generating responses to a query, only the contexts identified within the relevant uploaded PDF files are used. Furthermore, the 'temperature' parameter is set to zero, thereby eliminating randomness in the generation process. This approach ensures that the generated responses are closely aligned with the input contexts and maintain a high degree of specificity.

References

Tom B. Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, Sandhini Agarwal, Ariel Herbert-Voss, Gretchen Krueger, Tom Henighan, Rewon Child, Aditya Ramesh, Daniel M. Ziegler, Jeffrey Wu, Clemens Winter, Christopher Hesse, Mark Chen, Eric Sigler, Mateusz Litwin, Scott Gray, Benjamin Chess, Jack Clark, Christopher Berner, Sam McCandlish, Alec Radford, Ilya Sutskever, and Dario Amodei. 2020. Language models are few-shot learners.

Sébastien Bubeck, Varun Chandrasekaran, Ronen Eldan, Johannes Gehrke, Eric Horvitz, Ece Kamar, Peter Lee, Yin Tat Lee, Yuanzhi Li, Scott Lundberg, Harsha Nori, Hamid Palangi, Marco Tulio Ribeiro, and Yi Zhang. 2023. Sparks of artificial general intelligence: Early experiments with gpt-4.

Courtni Byun, Piper Vasicek, and Kevin Seppi. 2024. This reference does not exist: An exploration of LLM citation accuracy and relevance. In *Proceedings of the Third Workshop on Bridging Human–Computer Interaction and Natural Language Processing*, pages 28–39, Mexico City, Mexico. Association for Computational Linguistics.

Howard Chen, Ramakanth Pasunuru, Jason Weston, and Asli Celikyilmaz. 2023. Walking down the memory maze: Beyond context limit through interactive reading.

Shahul Es, Jithin James, Luis Espinosa-Anke, and Steven Schockaert. 2023. Ragas: Automated evaluation of retrieval augmented generation.

Tianyu Gao, Howard Yen, Jiatong Yu, and Danqi Chen. 2023. Enabling large language models to generate text with citations. *arXiv preprint arXiv:2305.14627*.

Yunfan Gao, Yun Xiong, Xinyu Gao, Kangxiang Jia, Jinliu Pan, Yuxi Bi, Yi Dai, Jiawei Sun, Qianyu Guo, Meng Wang, and Haofen Wang. 2024. Retrieval-augmented generation for large language models: A survey.

Dan Hendrycks, Collin Burns, Steven Basart, Andy Zou, Mantas Mazeika, Dawn Song, and Jacob Steinhardt. 2021. Measuring massive multitask language understanding.

Sebastian Hofstätter, Jiecao Chen, Karthik Raman, and Hamed Zamani. 2022. Fid-light: Efficient and effective retrieval-augmented text generation.

Bin Ji, Huijun Liu, Mingzhe Du, and See-Kiong Ng. 2024. Chain-of-thought improves text generation with citations in large language models. In *Proceedings of the Thirty-Eighth AAAI Conference on Artificial Intelligence and Thirty-Sixth Conference on Innovative Applications of Artificial Intelligence and Fourteenth Symposium on Educational Advances in Artificial Intelligence*, AAAI'24/IAAI'24/EAAI'24. AAAI Press.

Jean Kaddour, Joshua Harris, Maximilian Mozes, Herbie Bradley, Roberta Raileanu, and Robert McHardy. 2023. Challenges and applications of large language models.

Nikhil Kandpal, Haikang Deng, Adam Roberts, Eric Wallace, and Colin Raffel. 2023. Large language models struggle to learn long-tail knowledge.

LangChain. 2023a. Ensemble retriever. https://python.langchain.com/v0.1/docs/modules/data_connection/retrievers/ensemble/. Accessed: 2024-03-13.

LangChain. 2023b. Evaluating rag pipelines with ragas + langsmith. https://blog.langchain.dev/evaluating-rag-pipelines-with-ragas-langsmith/. Accessed: 2024-01-12.

LangChain. 2023c. Parent document retriever. https://python.langchain.com/v0.1/docs/modules/data_connection/retrievers/parent_document_retriever/. Accessed: 2024-03-13.

Patrick Lewis, Ethan Perez, Aleksandra Piktus, Fabio Petroni, Vladimir Karpukhin, Naman Goyal, Heinrich Küttler, Mike Lewis, Wen tau Yih, Tim Rocktäschel, Sebastian Riedel, and Douwe Kiela. 2021. Retrieval-augmented generation for knowledge-intensive nlp tasks.

Weitao Li, Junkai Li, Weizhi Ma, and Yang Liu. 2024. Citation-enhanced generation for llm-based chatbots. *arXiv preprint arXiv:2402.16063*.

Nelson F. Liu, Kevin Lin, John Hewitt, Ashwin Paranjape, Michele Bevilacqua, Fabio Petroni, and Percy Liang. 2023. Lost in the middle: How language models use long contexts.

Xinbei Ma, Yeyun Gong, Pengcheng He, Hai Zhao, and Nan Duan. 2023. Query rewriting in retrieval-augmented large language models. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pages 5303–5315, Singapore. Association for Computational Linguistics.

OpenAI. 2023. GPT-4 Technical Report.

Jon Saad-Falcon, Omar Khattab, Christopher Potts, and Matei Zaharia. 2023. Ares: An automated evaluation framework for retrieval-augmented generation systems.

Parth Sarthi, Salman Abdullah, Aditi Tuli, Shubh Khanna, Anna Goldie, and Christopher D Manning. 2024. Raptor: Recursive abstractive processing for tree-organized retrieval. In *The Twelfth International Conference on Learning Representations*.

Jiajun Shen, Tong Zhou, Yubo Chen, and Kang Liu. 2024. Citekit: A modular toolkit for large language model citation generation. *arXiv preprint arXiv:2408.04662*.

Aarohi Srivastava, Abhinav Rastogi, Abhishek Rao, Abu Awal Md Shoeb, Abubakar Abid, Adam Fisch, Adam R. Brown, Adam Santoro, Aditya Gupta, Adrià Garriga-Alonso, Agnieszka Kluska, Aitor Lewkowycz, Akshat Agarwal, Alethea Power, Alex Ray, Alex Warstadt, Alexander W. Kocurek, Ali Safaya, Ali Tazarv, Alice Xiang, Alicia Parrish, Allen Nie, Aman Hussain, Amanda Askell, Amanda Dsouza, Ambrose Slone, Ameet Rahane, Anantharaman S. Iyer, Anders Andreassen, and et. al. 2023. Beyond the imitation game: Quantifying and extrapolating the capabilities of language models.

Hugo Touvron, Thibaut Lavril, Gautier Izacard, Xavier Martinet, Marie-Anne Lachaux, Timothée Lacroix, Baptiste Rozière, Naman Goyal, Eric Hambro, Faisal Azhar, Aurelien Rodriguez, Armand Joulin, Edouard Grave, and Guillaume Lample. 2023. Llama: Open and efficient foundation language models.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. 2023. Attention is all you need.

Xintao Wang, Qianwen Yang, Yongting Qiu, Jiaqing Liang, Qianyu He, Zhouhong Gu, Yanghua Xiao, and Wei Wang. 2023. Knowledgpt: Enhancing large language models with retrieval and storage access on knowledge bases.

Fangyuan Xu, Weijia Shi, and Eunsol Choi. 2023. Recomp: Improving retrieval-augmented lms with compression and selective augmentation.

Haoyan Yang, Zhitao Li, Yong Zhang, Jianzong Wang, Ning Cheng, Ming Li, and Jing Xiao. 2023. PRCA: Fitting black-box large language models for retrieval question answering via pluggable reward-driven contextual adapter. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pages 5364–5375, Singapore. Association for Computational Linguistics.

Yue Zhang, Yafu Li, Leyang Cui, Deng Cai, Lemao Liu, Tingchen Fu, Xinting Huang, Enbo Zhao, Yu Zhang, Yulong Chen, Longyue Wang, Anh Tuan Luu, Wei Bi, Freda Shi, and Shuming Shi. 2023. Siren's song in the ai ocean: A survey on hallucination in large language models.

A Appendix

Contents

A.1 Retrieval-Augmented Generation (RAG)

Basic Retrieval-Augmented Generation (RAG) is an advanced technique that combines information retrieval with text generation, making it particularly effective when generating responses that require specific contextual information from an external knowledge base. The process is typically divided into three main stages: Ingestion, retrieval, and response generation.

Ingestion: Once an input file is read, the first stage in RAG involves chunking and embedding, where source texts are segmented into smaller, manageable units, which are then converted into embedding vectors for retrieval. Smaller chunks generally enhance query precision and relevance, while larger chunks may introduce noise, reducing accuracy. Effective chunk size management is crucial for balancing comprehensiveness and precision. Embedding transforms both the user's query and knowledge base documents into comparable formats, enabling the retrieval of the most relevant information.

Retrieval: In the next stage, the relevant information is retrieved from a vector knowledge base such as FAISS. The retriever searches this vector store to find the most relevant chunks of information based on the user's query. This stage is crucial for ensuring that the model has access to the necessary context for generating accurate and contextually relevant responses.

Response Generation: In the final stage, the retrieved context is combined with the user's query and fed into the LLM, such as GPT-4, to generate a coherent and relevant response. The model uses the context provided by the retrieved documents to produce answers that are informed by the most pertinent information available. This step highlights the synergy between retrieval and generation, ensuring that the output is not only accurate but also contextually grounded.

Each stage of the RAG process is designed to leverage the strengths of both retrieval and generation, enabling the creation of responses that are informed by specific and relevant external knowledge. By combining these components, RAG systems can significantly enhance the quality and relevance of generated content, making them a powerful tool for applications requiring precise and contextually aware responses.

The Basic RAG approach integrates a retriever and a language model to answer questions based on retrieved documents. It involves splitting documents into chunks, embedding them with models, and storing them in a vector database. The retriever fetches relevant chunks based on the query, which the language model uses to generate accurate responses.

The PDR RAG enhances retrieval precision by structuring documents into parent-child relationships. Larger parent chunks and smaller child chunks are embedded and stored in a vector database and in-memory store. A ParentDocumentRetriever fetches relevant chunks, providing refined context to the language model, ensuring more precise context and accurate responses.

The Ensemble RAG combines multiple retrievers to leverage their strengths, resulting in a more robust retrieval system. It uses different retrievers, such as BM25 for keyword matching and vector-based retrievers for semantic similarity. An EnsembleRetriever balances their contributions, using the aggregated context for the language model to generate responses, enhancing retrieval robustness and accuracy for complex queries.

A.2 Our System: LLM-Ref

In contrast to traditional RAG-based systems, our approach emphasizes preserving the hierarchical structure of source data in research writing, enabling the sequential retrieval of relevant contexts and references. During the ingestion stage, our method eliminates the need for a vector store, allowing extracted source information to be stored either online or offline, thereby enhancing flexibility. In the retrieval stage, we leverage large language models (LLMs) to identify the most relevant context paragraphs corresponding to the user query. This approach is particularly well-suited for research article writing, where our findings indicate that each paragraph typically presents a coherent argument, sufficient for establishing contextual similarity. Embedding-based approaches like FAISS rely on pre-computed vector similarities for similarity search and retrieval, which can lead to a loss of subtle contextual nuances present in the data. In contrast, large language models (LLMs) dynamically process and interpret text to capture complex, nuanced relationships within the text. Finally, in the generation stage, our system iteratively produces and refines the response, ensuring accuracy and relevance. While our approach invokes the LLM multiple times across various stages, the associated financial costs are minimal in the context of overall research expenditures.

Extracting both primary and secondary references from source documents requires the LLM to be deterministic. In research articles, the ability to extract contexts from exact paragraphs is crucial. Our experiments with popular commercial LLMs indicate that while such models can refer to uploaded source documents, their generative nature prevents them from providing exact reproductions of contexts or references from the original sources. As a result, it is challenging to precisely identify specific references or corresponding contexts in the original documents based on output responses.

A.2.1 Formal Definition

In this section, we formally define the LLM-Ref system for enhanced reference handling in scientific writing. LLM-Ref operates on a collection of hierarchical source documents and, given a user query, retrieves the most relevant contexts, synthesizes a response, and extracts both primary and secondary references. The following formalism specifies the data structures, functions, and system workflow that together characterize the LLM-Ref approach.

Let

$$D = \{d_1, d_2, \dots, d_N\}$$

be a set of source documents (e.g., research articles), each hierarchically organized into sections, subsections, and paragraphs. For each document $d \in D$, denote its set of paragraphs as

$$P_d = \{p_{d,1}, p_{d,2}, \dots, p_{d,M_d}\}.$$

Let the union of all paragraphs be

$$P = \bigcup_{d \in D} P_d.$$

Given a user query q, the objective is to generate a response r and extract relevant references through the following process:

1. **Contextual Synthesis:** Define a selection function
$$\mathcal{S} : (q, P) \to P_q \subseteq P$$
that retrieves a subset P_q of paragraphs most relevant to q, using LLM-based semantic similarity and the preserved document hierarchy.

2. **Response Generation:** Let G denote a response generation function that uses q and P_q to generate a text response:
$$r = G(q, P_q)$$
Here, G may operate iteratively to handle long contexts, concatenating outputs as needed.

3. **Reference Extraction:** Let X denote a reference extraction function, which produces:
   - *Primary References (R_p)*: the set of source documents from which any paragraph in P_q is drawn:
   $$R_p = \{d \in D \mid \exists p \in P_q \cap P_d\}$$
   - *Secondary References (R_s)*: the set of in-text citation details (enumerated or named) found in P_q, aligned with segments of r via a fine-grained matching process $X_{\text{fine}}(r, P_q)$.

The overall system function is defined as:

$$F : (q, D) \to (r, (R_p, R_s))$$

operating as follows:

1. **Content Extraction:** Extract all paragraphs P from the documents D, preserving hierarchical structure.

2. **Context Retrieval:** Select relevant paragraphs $P_q = \mathcal{S}(q, P)$ for the given query.

3. **Iterative Output Generation:** Generate the response $r = G(q, P_q)$, processing context in steps if needed.

4. **Reference Extraction:** Identify primary and secondary references via X.

Remark. Preserving the hierarchical document structure (sections, subsections, paragraphs) during extraction and retrieval enables LLM-Ref to maintain fine-grained alignment between queries, contexts, and references. This facilitates accurate synthesis and citation, outperforming traditional RAG systems which rely on flat or arbitrary chunking.

A.3 Result and Analysis of GPT-4o mini

A.3.1 Metric Analysis

Table 4 presents a comparison of performance metrics for LLM-Ref, Basic RAG, PDR RAG, Ens. RAG, and RAPTOR using GPT-4o-mini as the LLM.

LLM-Ref consistently outperforms all other methods across key dimensions. It achieves the highest Answer Relevancy score of 0.966, well above RAPTOR's 0.898, Basic RAG's 0.675, PDR RAG's 0.557, and Ens. RAG's 0.709. Its Answer Correctness of 0.546 is competitive to RAPTOR's 0.581 and the baselines' range of 0.465–0.531. With an Answer Similarity of 0.947, LLM-Ref remains competitive with RAPTOR's top score of 0.950 and surpasses other systems. In Context Relevancy, LLM-Ref leads with 0.246 versus RAPTOR's 0.116, highlighting its superior retrieval of pertinent passages. Although its Context Recall (0.732) slightly outperforms RAPTOR (0.613) and Ens. RAG (0.726), LLM-Ref's Context Precision of 0.980 far exceeds RAPTOR's 0.891 and the other methods. While RAPTOR attains the highest Faithfulness score of 0.667, LLM-Ref's 0.569 remains robust in grounding answers. Finally, LLM-Ref's composite Ragas Score of 0.486 substantially outstrips RAPTOR's 0.311 and the baseline RAGs' 0.116–0.159, underscoring its balanced gains in relevancy, accuracy, and overall answer quality.

A.3.2 Computation Costs

The proposed method is meticulously designed to support the writing of research articles, a task that requires a high degree of precision. Compared to traditional Retrieval-Augmented Generation (RAG) systems, our approach incurs higher computational costs due to its focus on achieving enhanced accuracy. However, leveraging open-source large language models (LLMs) fine-tuned for specific tasks can help mitigate these expenses.

The computational overhead of our system, in contrast to traditional RAG systems, can be articulated as follows:

1. **Content Extraction:** The system generates summaries for each paragraph extracted from the documents, storing these summaries for subsequent context extraction. The number of LLM calls made during this step is equal to the number of paragraphs, denoted as N. Traditional RAG systems typically do not invoke LLMs at this stage, instead generating embeddings and storing them in a vector index.

2. **Context Extraction:** During this phase, the LLM is invoked N times to find relevant paragraphs to the query, utilizing the paragraph summaries to minimize the token count, thereby reducing the computational load.

3. **Generation:** The generation of responses is conducted iteratively based on the retrieved contexts. The number of LLM calls in this phase depends on the number of contexts retrieved, denoted as c. Our experiments indicate that LLM-Ref retrieves approximately half the number of contexts compared to traditional RAG systems when all the relevant contexts are chosen, leading to reduced computational demands.

4. **Reference Extraction:** This step is unique to our system and involves additional LLM calls, denoted as $p \times q$, where p represents the number of lines in the generated response and q corresponds to the lines present in the context. This process ensures the precision and relevance of the extracted references.

LLM calls in content extraction are executed only once during the initial reading of the document and storage of summaries. However, each query necessitates LLM calls in context extraction, answer generation, and reference extraction.

Therefore, each query requires $(N + c + p \times q)$ LLM calls. Assuming we have $N = 50$ paragraphs, $c = 8$ contexts, $p = 7$ generated lines, and $q = 8$ lines per context, the total is 56 lines. Additionally, each paragraph contains 220 tokens on average, each line approximately 25 tokens, and prompts contain 60 tokens.

Name	Answer Relevancy	Answer Correctness	Answer Similarity	Context Relevancy	Context Precision	Context Recall	Faith fulness	Ragas Score
Basic RAG	0.675	0.517	0.890	0.049	0.846	0.698	0.582	0.159
PDR RAG	0.557	0.465	0.861	0.034	0.828	0.587	0.590	0.116
Ens. RAG	0.709	0.531	0.899	0.037	0.851	0.726	0.615	0.129
RAPTOR	0.898	**0.581**	**0.950**	0.116	0.891	0.613	**0.667**	0.311
LLM-Ref	**0.966**	0.546	0.947	**0.246**	**0.980**	**0.732**	0.569	**0.486**

Table 4: Metric evaluation result comparison of LLM-Ref with Basic RAG, Parent Document Retriever RAG, Ensemble Retrieval RAG, and RAPTOR using GPT 4o-mini as the LLM. A higher value of a metric indicates better performance.

$$
\begin{aligned}
N &= 50 \times (220 + 60) \\
&= 14,000 \,\text{tokens} \\
c &= 8 \times (7 \times 25 + 60) + 1000 \\
&= 2,880 \,\text{tokens} \\
p \times q &= 7 \times 8 \times 7 \\
&= 392 \,\text{LLM calls} \\
\text{Total tokens} &= 14,000 + 2,880 \\
&\quad + 392 \times (25 + 25 + 15) \\
&= 42,360 \,\text{tokens}
\end{aligned}
$$

Thus, the total input tokens amount to 42,360 tokens.

During both content extraction and reference extraction, the LLM returns only 'True' or 'False' for comparison, producing just one token. However, during generation, as it iteratively generates and refines the response, we estimate approximately 1,500 tokens are generated.

Output tokens = $50 + 1500 + 392 = 1942$ tokens.

If we use GPT-4o-mini, which costs \$0.150 per 1M input tokens and \$0.600 per 1M output tokens as of October 2024, the cost per query (CpQ) in USD is calculated as:

$$\text{CpQ} = \frac{0.150}{10^6} \times 42360 + \frac{0.600}{10^6} \times 1942 \approx 0.0075$$

Considering the funds typically allocated to research, the cost of using our proposed LLM-Ref for article writing is minimal. Table 3 provides a detailed account of the actual expenses associated with conducting the experiments outlined in Table 4.

In conclusion, while our system incurs higher computational costs, such costs are common in similar applications. Evaluation frameworks like Ragas and ARES, which rely on LLMs to assess similarities, incur similar expenses. In return, LLM-Ref offers enhanced accuracy and precision in content generation, crucial for research article writing.

A.4 Ablation Study

A.4.1 Performance Analysis on Different LLMs

Table 5 compares the performance metrics of LLM-Ref against Basic RAG, PDR RAG, Ens. RAG, and RAPTOR across various language models, including GPT-3.5, GPT-4o-mini, Llama 3.1-405b, and Claude 3.5 Sonnet. In this experiment, we focus exclusively on the computer science subset of the dataset. As before, a higher value across the metrics signifies superior performance. The results demonstrate LLM-Ref's consistent advantage over other methods, particularly in providing more relevant, correct, and similar answers.

In the GPT-3.5 evaluation, LLM-Ref achieves an Answer Relevancy score of 0.960, markedly higher than Basic RAG (0.545), PDR RAG (0.619), and Ens. RAG (0.629). It also leads in Answer Correctness with 0.555, surpassing the others' range of 0.412 to 0.471. With an Answer Similarity of 0.950, LLM-Ref maintains a strong advantage over its peers, which hover between 0.899 and 0.936. These metrics confirm LLM-Ref's superior capability to generate answers that are relevant and aligned with the provided context. Notably, while its Context Relevancy (0.157) is significantly higher than the others, it still lags behind in Context Recall, with scores slightly above those of Basic RAG (0.676 vs. 0.665) and Ens. RAG (0.775), but it compensates with a strong Faithfulness score of 0.721. The composite Ragas Score of 0.389 further highlights LLM-Ref's overall effectiveness compared to the other methods, which range from 0.052 to 0.143. By comparison, RAPTOR achieves an Answer Relevancy of 0.972, Answer Correctness of 0.626, and an overall Ragas Score of 0.319, outperforming baseline RAGs but still trailing LLM-Ref.

For GPT-4o-mini, LLM-Ref retains its dominance with an Answer Relevancy score of 0.953,

Name	Answer Relevancy	Answer Correctness	Answer Similarity	Context Relevancy	Context Precision	Context Recall	Faith fulness	Ragas Score
				GPT 3.5				
Basic RAG	0.545	0.412	0.899	0.044	0.999	0.665	0.588	0.143
PDR RAG	0.619	0.460	0.926	0.014	0.999	0.783	0.607	0.052
Ens. RAG	0.629	0.471	0.936	0.027	0.999	0.775	0.624	0.097
RAPTOR	0.972	0.626	0.951	0.118	0.999	0.677	0.664	0.319
LLM-Ref	0.960	0.555	0.950	0.157	0.993	0.676	0.721	**0.389**
				GPT 4o-mini				
Basic RAG	0.765	0.540	0.916	0.041	0.999	0.689	0.564	0.138
PDR RAG	0.606	0.482	0.875	0.033	0.993	0.569	0.524	0.112
Ens. RAG	0.857	0.572	0.939	0.027	0.993	0.757	0.668	0.096
RAPTOR	0.932	0.589	0.948	0.112	0.999	0.587	0.602	0.299
LLM-Ref	0.953	0.575	0.951	0.179	0.999	0.683	0.640	**0.413**
				Llama 3.1-405b				
Basic RAG	0.571	0.443	0.875	0.035	0.987	0.538	0.390	0.114
PDR RAG	0.642	0.439	0.887	0.022	0.999	0.682	0.570	0.079
Ens. RAG	0.744	0.491	0.915	0.030	0.999	0.725	0.641	0.105
RAPTOR	0.246	0.394	0.664	0.107	0.993	0.567	0.2	0.198
LLM-Ref	0.958	0.556	0.950	0.112	0.987	0.650	0.564	**0.300**
				Claude 3.5 Sonnet				
Basic RAG	0.634	0.544	0.941	0.042	0.999	0.694	0.691	0.142
PDR RAG	0.702	0.550	0.942	0.015	0.999	0.762	0.723	0.055
Ens. RAG	0.799	0.601	0.945	0.027	0.993	0.741	0.741	0.096
RAPTOR	0.326	0.385	0.668	0.124	0.999	0.632	0.275	0.245
LLM-Ref	0.964	0.637	0.954	0.195	0.999	0.654	0.561	**0.422**

Table 5: Metric Evaluation result comparison of LLM-Ref with Basic RAG, Parent Document Retriever RAG, and Ensemble Retrieval RAG for different LLMs. A higher value of a metric indicates better performance.

considerably higher than Basic RAG (0.765), PDR RAG (0.606), and Ens. RAG (0.857). Its Answer Correctness of 0.575 is on par with Ens. RAG (0.572) and significantly higher than other systems, reinforcing LLM-Ref's consistent accuracy. With the highest Answer Similarity (0.951) and a Ragas Score of 0.413, LLM-Ref continues to outperform other methods. However, its Context Recall (0.683) remains lower than PDR RAG (0.757) and Ens. RAG (0.689), suggesting room for improvement in extracting complete information from the context. By comparison, RAPTOR achieves an Answer Relevancy of 0.932, Answer Correctness of 0.589, Answer Similarity of 0.948, and a Ragas Score of 0.299, outperforming basic RAG variants but still trailing LLM-Ref in overall precision and effectiveness.

In the Llama 3.1-405b evaluation, LLM-Ref again exhibits superior performance with an Answer Relevancy score of 0.958 and an Answer Correctness score of 0.556, well above Basic RAG and PDR RAG, whose scores remain below 0.650. Its Answer Similarity of 0.950 and Faithfulness of 0.564 confirm that LLM-Ref provides high-quality, accurate responses while grounding its answers in relevant context. Although its Context Precision (0.987) is competitive, LLM-Ref still falls behind in Context Recall, with a score of 0.650 compared to Ens. RAG's 0.725. The Ragas Score for LLM-Ref is 0.300, much higher than Basic RAG (0.114) and PDR RAG (0.079). In contrast, RAPTOR attains an Answer Relevancy of 0.246, Answer Correctness of 0.394, Answer Similarity of 0.664, and a Ragas Score of 0.198, underscoring LLM-Ref's retrieval robustness across different model architectures.

Finally, with Claude 3.5 Sonnet, LLM-Ref maintains its strong performance across multiple metrics. It achieves the highest Answer Relevancy of 0.964, Answer Correctness of 0.637, and Answer Similarity of 0.954, outperforming other systems by substantial margins. While it continues to deliver accurate and relevant answers, its Context Recall score of 0.654 and Faithfulness score of 0.561 remain slightly lower compared to Ens. RAG (0.741 for both). Despite this, LLM-Ref achieves the highest overall Ragas Score of 0.422, highlighting its superior performance in generating accurate and consistent answers across varied language models. By contrast, RAPTOR records an Answer Relevancy of 0.326, Answer Correctness of 0.385, Answer Similarity of 0.668, and a Ragas Score of 0.245, reinforcing the comprehensive advantage of LLM-Ref.

Across all LLM evaluations, LLM-Ref excels in delivering answers that are relevant, correct, and well-aligned with the input context. Its higher Ragas Scores across all models demonstrate its effectiveness in handling complex retrieval tasks.

A.4.2 LLM-Ref with BERT

Table 6 compares RAG variants, LLM-Ref-L, and LLM-Ref-B using an RTX 2080 GPU and a Core i9-9900 CPU. For this BERT experiments, we used huggingface's default hyper-parameter configuration. LLM-Ref-B—employing RoBERTa-large for similarity—achieves strong Answer Relevancy (0.952) and Answer Similarity (0.951), but its Context Relevancy (0.090) and overall Ragas Score (0.259) lag behind LLM-Ref-L (0.182 and 0.416, respectively), reflecting the limitations of an un-fine-tuned BERT model in domain-specific retrieval. Interestingly, RAPTOR—our new baseline—attains an Ragas Score of 0.299, outperforming LLM-Ref-B (0.259) by better balancing retrieval quality (Context Relevancy 0.112 vs. 0.090) and faithfulness (0.602 vs. 0.594). However, RAPTOR still falls short of LLM-Ref-L across most dimensions, underscoring the benefit of LLM-based similarity matching and hierarchical context extraction for robust, high-quality answer generation.

LLM-Ref-B achieves a high Answer Relevancy score (0.952) and Answer Similarity score (0.951), indicating its ability to retrieve and generate semantically aligned responses. However, its Context Relevancy score (0.090) is significantly lower than that of LLM-Ref-L (0.182), suggesting that the BERT-based retrieval mechanism struggles to identify the most relevant supporting contexts. This limitation directly impacts its overall Ragas score (0.259), which is lower than LLM-Ref-L (0.416), but still higher than the baseline RAG models.

One of the primary reasons for LLM-Ref-B's lower performance in retrieval quality is that BERT was not trained or fine-tuned for the specific domain of research article retrieval, making it less effective in capturing nuanced contextual dependencies. While embedding-based models like BERT provide a cost-effective alternative, their reliance on pre-trained representations without task-specific fine-tuning results in suboptimal performance when compared to LLM-based retrieval approaches. The drop in RAGAS score further suggests that as the number of relevant contexts decreases, LLM-Ref-B becomes less effective at retrieving and aligning information.

Despite these limitations, LLM-Ref-B still outperforms baseline RAG systems, particularly in Answer Relevancy, Faithfulness, and overall retrieval effectiveness. The results suggest that while BERT-based models offer a computationally efficient alternative, the lack of domain adaptation hinders their ability to match the performance of LLM-based similarity computation methods. These findings reinforce the importance of hierarchical paragraph extraction and the need for fine-tuning similarity models to enhance retrieval performance.

A.4.3 Stability Study

As presented in Table 1 and Table 4, we provide comprehensive sets of evaluation metrics that underscore the effectiveness of our system. To assess our system's performance, it is essential to consider it holistically. Specifically, the context precision and context recall metrics are crucial for evaluating the retrieval stage, while faithfulness and answer relevancy are key indicators of the system's performance during the generation stage. Our metrics demonstrate superior performance across these stages.

In the content extraction stage, the process is deterministic; the system can either successfully extract text from a document or not. However, the summarization process introduces variability, as different summaries may be generated in each run, potentially impacting context extraction and the final response. To evaluate the stability of our system, we conducted multiple runs, with results indicating consistent performance with respect to Table 1 given in the paper.

In the retrieval stage, unlike traditional RAG systems that typically select the top-k contexts, our approach involves retrieving all available contexts. This comprehensive retrieval method enhances the system's ability to generate accurate responses.

During the generation stage, we used a temperature setting of zero, ensuring that the model relies solely on the input context to generate responses, thereby minimizing randomness. We also experimented with varying the temperature parameter to observe its impact on response quality, as detailed in Table 7. We observed that as the temperature set-

Name	Answer Relevancy	Answer Correctness	Answer Similarity	Context Relevancy	Context Precision	Context Recall	Faith fulness	Ragas Score
Basic RAG	0.765	0.540	0.916	0.041	0.999	0.689	0.564	0.138
PDR RAG	0.606	0.482	0.875	0.033	0.993	0.569	0.524	0.112
Ens. RAG	0.857	0.572	0.939	0.027	0.993	0.757	0.668	0.096
RAPTOR	0.932	0.589	0.948	0.112	0.999	0.587	0.602	0.299
LLM-Ref-L	0.957	0.574	0.951	0.182	0.999	0.676	0.623	0.416
LLM-Ref-B	0.952	0.591	0.951	0.090	0.999	0.614	0.594	0.259

Table 6: Metric evaluation result comparison of different systems. Higher values indicate better performance. Here, LLM-Ref-L employs an LLM for similarity computation, whereas LLM-Ref-B utilizes a BERT model.

ting increases, the model tends to incorporate more of its pre-existing knowledge, which may include biases from its training data, potentially impacting the final Ragas score. The temperature parameter's influence on the model's output highlights the delicate balance between utilizing retrieved context and minimizing reliance on potentially biased or extraneous information stored within the model. Consequently, adjusting the temperature parameter is crucial for maintaining the accuracy and integrity of the generated responses.

These ablation studies highlight the robustness and adaptability of our system in generating precise and contextually relevant responses.

A.5 Prompt Designs

In our tool, we employ a large language model (LLM) to determine contextual similarity. To find the relevant contexts, we utilize the following prompt (given in Figure 3) which returns 'True' when a paragraph is relevant to the query. This prompt instructs the LLM to evaluate a given paragraph in the context of a specific query, determining if it provides direct answers or significant contributions. Since we utilize entire paragraphs that convey specific concepts, the LLM can discern relevance to the query by understanding subtle nuances. By responding with 'True' or 'False', the model identifies relevant information without additional explanation, thereby enhancing the accuracy and efficiency of our tool.

To address challenges associated with long contexts, we employ an iterative approach to output generation. Initially, a response is generated using the first context and query, utilizing the LLM prompt provided in Figure 4.

This prompt (given in Figure 4) directs the LLM to summarize and synthesize the paragraph to address the query coherently. By preserving the original vocabulary and style, the LLM ensures a natural and consistent tone. This iterative approach manages long contexts and enhances the relevance and cohesiveness of the responses, improving our tool's efficiency and accuracy. After the initial response is generated, subsequent responses are refined by incorporating later contexts using the following prompt (shown in Figure 5). This iterative approach not only enhances the comprehensiveness of the synthesized output but also helps in mitigating any errors present in the earlier responses.

This prompt (given in Figure 5) guides the LLM to integrate new paragraph information into the existing synthesis, maintaining coherence, relevance,

```
You are an experienced researcher tasked
     with identifying relevant
    information.
Paragraph: {paragraph}
Query: {query}
Instructions: Determine whether the
    paragraph provides information that
    directly answers or significantly
    contributes to the query.
If the paragraph is relevant to the
    query, respond with 'True'. If it is
     not relevant, respond with 'False'.
     Provide no additional explanation.
```

Figure 3: Prompt to find relevant contexts to a query.

```
You are a researcher writing a research
    paper.
**Paragraph**: {paragraph}
**Query**: {query}
**Instructions**: Summarize and
    synthesize the provided paragraph to
     create a cohesive and informative
    paragraph that addresses the query.
Ensure the synthesis uses the vocabulary
     and writing style of the original
    paragraph to maintain a natural and
    consistent tone.
```

Figure 4: Prompt used to generate the response based on the context for query.

Impact of temperature change							
Temperature	**Answer Relevancy**	**Answer Correctness**	**Context Relevancy**	**Context Precision**	**Context Recall**	**Faith fulness**	**Ragas Score**
0.0	0.94	0.72	0.44	0.29	0.71	0.44	0.57
0.05	0.94	0.71	0.40	0.27	0.74	0.40	0.54
0.1	0.95	0.71	0.44	0.28	0.70	0.44	0.56
0.15	0.93	0.67	0.35	0.24	0.65	0.37	0.49
Performance variation across different runs for the same queries							
Runs	**Answer Relevancy**	**Answer Correctness**	**Context Relevancy**	**Context Precision**	**Context Recall**	**Faith fulness**	**Ragas Score**
Run 1	0.95	0.70	0.35	0.24	0.71	0.41	0.52
Run 2	0.94	0.72	0.44	0.29	0.71	0.44	0.57
Run 3	0.94	0.70	0.38	0.25	0.68	0.45	0.54

Table 7: Stability study of our proposed approach.

```
You are a researcher writing a research
    paper.
**Existing Synthesis**: {response}
**New Paragraph**: {paragraph}
**Query**: {query}
**Instructions**: Integrate the
    information from the new paragraph
    into the existing synthesis to
    create a cohesive and informative
    paragraph that addresses the query.
Ensure the synthesis uses the vocabulary
     and writing style of the original
    paragraphs to maintain a natural and
     consistent tone.
```

Figure 5: Prompt used to integrate new context into existing responses.

```
For a given synthesized result based on
    some source paragraphs, find the
    relevant source lines that are most
    relevant to each line of the
    synthesized result.
Synthesized result: {synthesized_result
    }.
Source Paragraphs:  {context}.
Just provide the source lines for each
    line of synthesized result, for
    example: Synthesized Line: ...
    Corresponding Source Line: ...   Do
    not add explanation and source lines
     if they are not exactly relevant.
```

Figure 6: Prompt for identifying the most relevant source lines for each line in a synthesized result.

and a consistent tone, while iteratively refining responses to address long context complexities and improve the tool's accuracy and cohesiveness.

Figure 6 shows a prompt directing the LLM to match each line of a synthesized result with the most relevant source lines from the provided paragraphs. The output lists only the precisely relevant source lines, enhancing the traceability and transparency of the synthesis process by clarifying the origins of each part of the synthesized result.

Figure 7 presents a prompt to generate questions by synthesizing information from at least two of three provided documents. The prompt requires formulating questions, including exact original context texts, and providing answers, all in a specified Python format. This ensures the integrity of the original contexts for evaluation. Questions are generated until a certain number of unique questions are produced, enhancing the tool's ability to synthesize information accurately across multiple documents.

A.6 Ragas Evaluation Metrics

The Ragas score is computed by calculating the harmonic mean of Faithfulness (FF), Answer Relevancy (AR), Context Precision (CP), and Context Recall (CR).

$$\text{Ragas Score} = \frac{4}{\frac{1}{\text{FF}} + \frac{1}{\text{AR}} + \frac{1}{\text{CP}} + \frac{1}{\text{CR}}} \quad (1)$$

In this equation, FF stands for Faithfulness, AR represents Answer Relevancy, CP is Context Precision, and CR denotes Context Recall. In the RAGs framework, Faithfulness and Answer Relevancy assess the accuracy of content generation, while Context Precision and Context Recall evaluate the effectiveness of information retrieval. Therefore, the Ragas score ensures a robust assessment of both generation and retrieval processes in RAGs.

```
You are an expert research scientist.
Instructions: Create a list of 150
    questions (max 5 at a time) that
    require using information from all
    three provided input documents (or
    at least two of the input documents)
    . For each question, please include
    the following details:

Question: Formulate a question that
    integrates information from multiple
     documents.
Original Context Texts: Provide the
    exact contexts from the documents
    that were used to create the
    question, without any alterations.
Answer: Provide an answer for a research
     article derived from the original
    context texts.
Ensure that each question requires the
    synthesis of information from
    multiple documents. Maintain the
    integrity of the original context
    texts as they will be used later for
     evaluation purposes.
Return the response in the following
    python format:
data = [
    {
        "question": "Question 1",
        "context": ["Context 11",
             Context   12   ],
        "ground_truth": "Answer 1"
    },
    {
        "question": "Question 2",
        "context": ["Context 21",
             Context   22   ],
        "ground_truth": "Answer 2"
    },]


Please keep generating only if it is
    possible to generate unique
    questions that you did not generate
    them before. Generate 5 questions at
     a time. I want a total 150
    questions.
```

Figure 7: Prompts for generating Question-Context-Answer pair from source documents.

Faithfulness (FF): The Faithfulness score measures how relevant the statements in an answer are to the provided context. Scores for this metric range from 0 to 1, with higher scores indicating better alignment and performance. The calculation process, as defined by the Ragas framework, involves three key steps: first, extracting statements from the generated answers; second, determining the contextual relevance of these statements using the LLM; and third, calculating the Faithfulness score by dividing the number of context-relevant statements by the total number of statements. This score provides a quantifiable measure of how faithfully the model's answers reflect the original context. It is calculated as:

$$FF = \frac{NCS}{TS} \tag{2}$$

Here, NCS refers to the Number of Context-Relevant Statements, and TS represents the Total Statements in the Answer.

Answer Relevancy (AR): The Answer Relevance metric evaluates how closely the answers generated by a Language Learning Model (LLM) align with the original questions posed. Answers that are incomplete or redundant receive lower scores, with scores ranging from 0 to 1, where higher scores indicate better performance. The Ragas framework calculates this metric through a three-step process: first, generating pseudo-questions from both the context and the generated answer; second, calculating the cosine similarity between the original question and each pseudo-question; and third, computing the average of these cosine similarities. This average provides a quantitative measure of how relevant the generated answers are to the original questions.

$$AR = \frac{\sum CS}{NPQ} \tag{3}$$

In this context, CS denotes Cosine Similarities between pseudo-questions and the original question, and NPQ stands for the Number of Pseudo-Questions.

Context Precision (CP): The Context Precision metric measures how effectively a Language Learning Model (LLM) retrieves the necessary contextual information required to accurately answer a question. Scores for this metric range from 0 to 1, with higher scores indicating better retrieval performance. According to the Ragas framework, Context Precision is calculated through a two-step process: first, determining the relationship between each retrieved-context and the original question using the LLM, where the context is marked as either relevant (Yes) or not (No); and second, computing the Mean Average Precision (mAP) across all retrieved contexts. This score indicates how accurately the model retrieves relevant information to support its answers.

$$CP = mAP \tag{4}$$

Context Recall (CR): The Context Recall metric evaluates how well the context retrieved by a Language Learning Model (LLM) matches the Ground Truth, indicating the completeness of the information retrieval. Scores range from 0 to 1, with higher scores reflecting better performance. The Ragas framework computes this metric through a three-step process: first, splitting the Ground Truth into individual sentences; second, determining the relationship between each sub-Ground Truth sentence and the retrieved context using the LLM, marking each as either relevant (Yes) or not (No); and third, calculating the Context Recall score by dividing the number of context-relevant Ground Truth sentences by the total number of Ground Truth sentences. This score helps in quantifying how thoroughly the model's retrieved-context covers the Ground Truth.

$$\mathrm{CR} = \frac{\mathrm{NGTS}}{\mathrm{TGS}} \tag{5}$$

Here, NGTS stands for the Number of Ground Truth Sentences inferred from the given contexts, and TGS represents the Total Ground Truth Sentences.

Context Relevance (CRL): The Context Relevance metric evaluates the quality of the retrieved contexts for a question. A highly relevant context is important because it reduces the risk of confusing the language model or prompting hallucinations, and it ensures the answer can be directly justified by the source text. It primarily assesses the retriever's performance in providing useful, focused evidence for the question-answering pipeline. The metric is defined as:

$$\mathrm{CRL} = \frac{\mathrm{NERS}}{\mathrm{TSC}} \tag{6}$$

Here, NERS stands for the number of extracted relevant sentences, and TSC represents the total number of sentences in the context. This ratio ranges from 0 to 1, with 1.0 indicating a perfectly focused context (every sentence in the provided context was deemed needed for the answer) and lower values indicating that some of the context was extraneous.

Answer Similarity: Answer Similarity quantifies how semantically close the generated answer (response) is to the ground truth answer (reference) by leveraging embedding-based representations. Both texts are converted into high-dimensional vector representations via an embedding model. These vectors are then normalized by dividing by their respective Euclidean norms, and the cosine similarity is computed as the dot product of the normalized vectors. This method ensures that the similarity measure directly reflects the semantic alignment between what is expected and what is generated.

Answer Correctness: Answer Correctness is a composite evaluation metric within retrieval-augmented generation frameworks designed to assess whether a generated answer accurately reflects the intended content and is factually supported by the reference or ground truth. This metric operates on two main dimensions. First, it evaluates factuality by decomposing the answer into discrete statements and then verifying each against the ground truth, categorizing them as true positives, false positives, or false negatives. These classifications are combined using an F-beta score, which balances precision and recall to quantify the degree of factual correctness. Second, Answer Correctness incorporates a semantic similarity component that measures the degree of semantic alignment between the generated answer and the reference answer using embedding-based methods. The overall score is derived from a weighted average of these two components, ensuring that the answer is not only factually accurate but also semantically coherent with the expected response.

A.7 Examples of Query-Answer Pairs

We present additional query-answer pairs with fine-grained references extracted from LLM-Ref when different LLMs are utilized. This demonstrates the compatibility of our tool across different LLMs.

Query: Pros and Cons of Discrete vs Cascaded Systems

Answer: Discrete units in direct speech-to-speech translation (S2ST) systems offer several advantages over cascaded systems [1-8]. Firstly, they have lower computational costs and inference latency because they require fewer decoding steps [1, 8, 10-12]. This makes them more efficient and faster in translating speech [1-3, 7, 8, 10-12]. Additionally, direct S2ST is a natural approach for languages that do not have a writing system, as it eliminates the need for intermediate text representation. Another benefit of direct S2ST systems with discrete units is their flexibility in training. They can be trained with or without text transcripts, making them suitable for both written and unwritten languages [1, 6, 7, 9, 13]. This versatility allows for a wider range of applications and language support. However, it is important to note that there is a performance gap between direct S2ST systems with discrete units and cascaded systems. Cascaded systems, which combine speech-to-text translation (S2T) and text-to-speech (TTS), may have an edge in terms of translation accuracy [1-3, 6, 7, 9, 13]. The paragraph does not provide specific details about the extent of this performance gap, but it suggests that cascaded systems may outperform direct S2ST systems in terms of translation quality. In summary, discrete units in direct S2ST systems offer advantages such as lower computational costs, support for languages without a writing system, and flexibility in training. However, there is a performance gap compared to cascaded systems, which may have better translation accuracy [1-3].

References:

[1] A. Lee et al., "Direct speech-to-speech translation with discrete units," arXiv, 2021.
[2] A. Lee et al., "Textless speech-to-speech translation on real data," arXiv, 2021.
[3] S. Popuri et al., "Enhanced Direct Speech-to-Speech Translation Using Self-supervised Pre-training and Data Augmentation," arXiv, 2022.
[4] Ye Jia et al., Direct speech-to-speech translation with a sequence-to-sequence model. Proc. Interspeech 2019.
[5] Ye Jia et al., Translatotron 2: Robust direct speech-to-speech translation. arXiv 2021.
[6] Lavie et al., "JANUS-III: Speech-to-speech translation in multiple languages." In 1997 IEEE International Conference on Acoustics, Speech, and Signal Processing.
[7] S. Nakamura, The ATR multilingual speech-to-speech translation system. IEEE Transactions on Audio, Speech, and Language Processing, 2006.
[8] W. Hsu, HuBERT: Self-supervised speech representation learning by masked prediction of hidden units. arXiv preprint arXiv:2106.07447.
[9] C. Zhang, X. Tan et al., "UWSpeech: Speech to speech translation for unwritten languages," arXiv:2006.07926, 2020.
[10] Q. T. Do, et al., "Toward expressive speech translation: A unified sequence-to-sequence LSTMs approach for translating words and emphasis." In INTERSPEECH, 2017.
[11] P. D. Aguero, et al., "Prosody generation for speech-to-speech translation." In 2006 IEEE International Conference on Acoustics Speech and Signal Processing Proceedings, volume 1, pages I–I, 2006.
[12] G. K. Anumanchipalli et al., "Intent transfer in speech-to-speech machine translation." In 2012 IEEE Spoken Language Technology Workshop (SLT), 2012.
[13] A. Tjandra et al., "Speech-to-speech translation between untranscribed unknown languages." In 2019 IEEE Automatic Speech Recognition and Understanding Workshop (ASRU), 2019.

Figure 8: Fine-grained reference samples generated by LLM-Ref when GPT-3.5 is used as the LLM.

Query: How does data augmentation improve speech-to-speech translation?

Answer: Data augmentation plays a crucial role in enhancing speech-to-speech translation (S2ST) systems by addressing the challenges posed by data scarcity [1-8]. In the context of end-to-end S2T models, effective training strategies such as multitask learning and model pre-training are complemented by data augmentation techniques, which help to enrich the training dataset[1-4, 9-13]. This enrichment allows the models to better generalize and improve their performance, particularly in mitigating the error propagation issues commonly encountered in traditional ASR+MT systems[1-3, 14-16]. By diversifying the training data, data augmentation facilitates the synthesis of para-linguistic information, such as prosody and word-level emphasis, which are essential for producing more natural and contextually appropriate translations[1-3, 17-19]. Recent empirical findings further demonstrate that augmenting supervised S2ST data with directly mined S2ST data, particularly when combined with speech normalization techniques, can significantly enhance the performance of textless S2ST systems[2-3]. For instance, the integration of mined data has shown to yield substantial improvements in BLEU scores across various language pairs, indicating that the additional data not only enriches the training set but also aids in capturing variations in speech that are difficult to model without such information.Moreover, the application of data augmentation techniques, such as utilizing ASR, MT, and TTS models to create weakly supervised data, has been shown to effectively tackle the data scarcity issue in S2ST. By leveraging the S2UT framework, researchers have demonstrated that large-scale self-supervised pre-training with monolingual speech and text data, combined with data augmentation, can significantly enhance S2ST model training [3,12]. Specifically, the use of a wav2vec 2.0 speech encoder and an mBART text decoder trained with discrete units extracted from unlabeled speech data has proven effective [2-3, 14-16]. The incorporation of weakly supervised data from ASR speech further increases the size of the parallel S2ST training data, leading to improved model performance[2,3]. For example, experiments have shown that with a pre-trained wav2vec 2.0 encoder and a randomly initialized decoder, substantial BLEU gains can be achieved, indicating that data augmentation not only enriches the training dataset but also enhances the model's ability to generalize across different setups, including low-resource scenarios[2-3]. Thus, through these enhancements, data augmentation significantly contributes to the robustness and efficacy of S2ST systems, enabling them to leverage both supervised and mined data for improved translation quality[1-2].

References:

[1] A. Lee et al., "Direct speech-to-speech translation with discrete units," arXiv, 2021.
[2] A. Lee et al., "Textless speech-to-speech translation on real data," arXiv, 2021.
[3] S. Popuri et al., "Enhanced Direct Speech-to-Speech Translation Using Self-supervised Pre-training and Data Augmentation," arXiv, 2022.
[4] Ye Jia et al., Direct speech-to-speech translation with a sequence-to-sequence model. Proc. Interspeech 2019.
[5] R. J. Weiss, J. Chorowski et al., "Sequence-to-sequence models can directly translate foreign speech," Proc. Interspeech, 2017.
[6] J. Pino, Q. Xu et al., "Self-training for end-to-end speech translation," Proc. Interspeech, 2020.
[7] J. Kahn, A. Lee et al., "Self-training for end-to-end speech recognition," in ICASSP, 2020.
[8] T. Hayashi, S. Watanabe et al., "Back-translation-style data augmentation for end-to-end asr," in SLT, 2018.
[9] Alexandre Bérard, Olivier Pietquin, Christophe Servan, and Laurent Besacier. 2016. Listen and translate: A proof of concept for end-to-end speech-to-text translation. arXiv preprint arXiv:1612.01744.
[10] Parnia Bahar, Tobias Bieschke, and Hermann Ney. 2019. A comparative study on end-to-end speech to text translation. In 2019 IEEE Automatic Speech Recognition and Understanding Workshop (ASRU), pages 792–799. IEEE.
[11] Xian Li, Changhan Wang, Yun Tang, Chau Tran, Yuqing Tang, Juan Pino, Alexei Baevski, Alexis Conneau, and Michael Auli. 2021. Multilingual speech translation from efficient finetuning of pretrained models. In Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers), pages 827–838.
[12] C. Zhang, X. Tan et al., "UWSpeech: Speech to speech translation for unwritten languages," arXiv:2006.07926, 2020.
[13] A. Polyak, Y. Adi et al., "Speech resynthesis from discrete disentangled self-supervised representations," arXiv:2104.00355, 2021.
[14] M. C. Stoian, S. Bansal et al., "Analyzing asr pretraining for lowresource speech-to-text translation," in ICASSP, 2020.
[15] A. Baevski, Y. Zhou et al., "wav2vec 2.0: A framework for selfsupervised learning of speech representations," Neurips, vol. 33, pp. 12 449–12 460, 2020.
[16] Y. Liu, J. Gu et al., "Multilingual denoising pre-training for neural machine translation," Transactions of the Association for Computational Linguistics, vol. 8, pp. 726–742, 2020.
[17] Q. T. Do, et al., "Toward expressive speech translation: A unified sequence-to-sequence LSTMs approach for translating words and emphasis." In INTERSPEECH, 2017.
[18] P. D. Aguero, et al., "Prosody generation for speech-to-speech translation." In 2006 IEEE International Conference on Acoustics Speech and Signal Processing Proceedings, volume 1, pages I–I, 2006.
[19] G. K. Anumanchipalli et al., "Intent transfer in speech-to-speech machine translation." In 2012 IEEE Spoken Language Technology Workshop (SLT), 2012.

Figure 9: Fine-grained reference samples generated by LLM-Ref when GPT-4o-mini is used as the LLM.

Query: Literature review on Direct speech-to-speech translation?

Answer: Direct speech-to-speech translation (S2ST) aims at translating speech from one language into speech in another language without relying on text generation as an intermediate step. Compared to conventional cascaded approaches, which take advantage of automatic speech recognition (ASR), machine translation (MT) or end-to-end speech-to-text translation (S2T) followed by text-to-speech synthesis (TTS), direct S2ST has the advantage of faster inference and can support translation between languages without text writing systems [1-12]. Recent studies have proposed various approaches to direct S2ST, including self-supervised speech encoder pre-training with unlabeled speech data, and data augmentation techniques that benefit S2T model training [1-3, 13-20,.]. For instance, [4] proposes to apply a self-supervised speech encoder pre-trained on unlabeled speech to convert target speech into discrete units and build a speech-to-unit translation (S2UT) model for direct S2ST [1-3, 5, 9-11, 15, 17, 20,.]. Self-supervised discrete targets can disentangle linguistic content from speaker identity and prosodic information in speech, and enable opportunities for applying techniques from speech-to-text model training, such as ASR and S2T, to direct S2ST[1-5, 10, 13 20-23]. Moreover, [6] shows that incorporating weakly supervised training data from ASR speech can bring significant improvements to direct S2ST models[1-5, 13, 15, 20-22]. Our work builds upon these advances and explores the effectiveness of self-supervised pre-training and data augmentation for direct S2ST models.

References:

[1] A. Lee et al., "Direct speech-to-speech translation with discrete units," arXiv, 2021.
[2] A. Lee et al., "Textless speech-to-speech translation on real data," arXiv, 2021.
[3] S. Popuri et al., "Enhanced Direct Speech-to-Speech Translation Using Self-supervised Pre-training and Data Augmentation," arXiv, 2022.
[4] A. Tjandra et al., "Speech-to-speech translation between untranscribed unknown languages." In 2019 IEEE Automatic Speech Recognition and Understanding Workshop (ASRU), 2019.
[5] C. Zhang, X. Tan et al., "UWSpeech: Speech to speech translation for unwritten languages," arXiv:2006.07926, 2020.
[6] Lavie et al., "JANUS-III: Speech-to-speech translation in multiple languages." In 1997 IEEE International Conference on Acoustics, Speech, and Signal Processing.
[7] S. Nakamura, The ATR multilingual speech-to-speech translation system. IEEE Transactions on Audio, Speech, and Language Processing, 2006.
[8] Alexandre Bérard et. al.. 2016. Listen and translate: A proof of concept for end-to-end speech-to-text translation. arXiv preprint arXiv:1612.01744.
[9] Ye Jia et al., Direct speech-to-speech translation with a sequence-to-sequence model. Proc. Interspeech 2019.
[10] Ye Jia et al., Translatotron 2: Robust direct speech-to-speech translation. arXiv 2021.
[11] Takatomo Kano et. al. 2021. Transformer-based direct speech-to-speech translation with transcoder. In 2021 IEEE Spoken Language Technology Workshop (SLT), pages 958–965. IEEE.
[12] Aaron van den Oord et. al., 2017. Neural discrete representation learning. In Proceedings of the 31st International Conference on Neural Information Processing Systems, pages 6309–6318.
[13] Shu-wen Yang et al. 2021. SUPERB: Speech processing universal performance benchmark. arXiv preprint arXiv:2105.01051.
[14] A. Baevski, Y. Zhou et al., "wav2vec 2.0: A framework for selfsupervised learning of speech representations," Neurips, vol. 33, pp. 12 449–12 460, 2020.
[15] W. Hsu, HuBERT: Self-supervised speech representation learning by masked prediction of hidden units. arXiv preprint arXiv:2106.07447.
[16] Zhiyun Fan et. al. 2020. Exploring wav2vec 2.0 on speaker verification and language identification. arXiv preprint arXiv:2012.06185.
[17] R. J. Weiss, J. Chorowski et al., "Sequence-to-sequence models can directly translate foreign speech," Proc. Interspeech, 2017.
[18] Parnia Bahar et al. 2019. A comparative study on end-to-end speech to text translation. In 2019 IEEE Automatic Speech Recognition and Understanding Workshop (ASRU), pages 792–799. IEEE.
[19] Xian Li et. al. 2021. Multilingual speech translation from efficient finetuning of pretrained models. In Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers), pages 827–838.
[20] Kushal Lakhotia et al. 2021. Generative spoken language modeling from raw audio. arXiv preprint arXiv:2102.01192.
[21] A. Polyak, Y. Adi et al., "Speech resynthesis from discrete disentangled self-supervised representations," arXiv:2104.00355, 2021.
[22] E. Kharitonov, A. Lee et al., "Text-free prosody-aware generative spoken language modeling," arXiv:2109.03264, 2021.
[23] F. Kreuk et al., "Textless speech emotion conversion using decomposed and discrete representations," arXiv:2111.07402, 2021.

Figure 10: Fine-grained reference samples generated by LLM-Ref when Llama is used as the LLM.