JOURNAL OF SCIENCE
OF LAC HONG UNIVERSITY
ISSN: 2525 - 2186
Tạp chí Khoa học Lạc Hồng, **2025**, 20, 036-043

TĂNG CƯỜNG TRÍCH XUẤT DỮ LIỆU VỚI LLMS VÀ HỆ THỐNG RAG

Trần Trọng Kiên[1*], Khâu Văn Bích[2]
[1]*Trung tâm Chuyển đổi số tỉnh Bình Dương, số 36 Trịnh Hoài Đức, phường Phú Lợi, Thành phố Thủ Dầu Một, tỉnh Bình Dương, Việt Nam.*
[2]*Trường Đại học Trần Đại Nghĩa, số 189 Nguyễn Oanh, phường 10, quận Gò Vấp, Thành phố Hồ Chí Minh, Việt Nam*
Tác giả liên hệ: *kienttsbc@gmail.com*

THÔNG TIN BÀI BÁO

Ngày nhận: 13/2/2025
Ngày hoàn thiện: 2/1/2025
Ngày chấp nhận: 16/1/2025
Ngày đăng: 15/3/2025

TỪ KHÓA

Mô hình ngôn ngữ lớn;
Thu thập dữ liệu tăng cường;
Tối ưu hóa phân đoạn;
Thu thập dữ liệu.

TÓM TẮT

Các Mô Hình Ngôn Ngữ Lớn (LLMs) có khả năng mô phỏng tốt các nhiệm vụ của con người nhưng hạn chế trong trích xuất dữ liệu chính xác. Để khắc phục, LLMs được tích hợp với Hệ Thống Tạo Dữ Liệu Kết Hợp Truy Xuất (RAG), sử dụng các thuật toán phân đoạn, tìm kiếm, và xếp hạng để cải thiện truy xuất thông tin từ văn bản không có cấu trúc. Nghiên cứu thực nghiệm đã tối ưu hóa các phương pháp phân đoạn và đề xuất khung lý thuyết, mở đường cho sự phát triển và mở rộng ứng dụng của các hệ thống RAG trong tương lai.

INTEGRATING PRE-TRAINED LLMS WITH RAG FOR EFFICIENT CONTENT RETRIEVAL

Tran Trong Kien[1*], Khau Van Bich[2]
[1]*Binh Duong Province Digital Transformation Center, No. 36 Trinh Hoai Duc Street, Phu Loi Ward, Thu Dau Mot City, Binh Duong Province, Vietnam.*
[2]*Tran Dai Nghia University, No. 189 Nguyen Oanh Street, Ward 10, Go Vap District, Ho Chi Minh City, Vietnam*
*Corresponding Author: *kienttsbc@gmail.com*

ARTICLE INFO

Received: Feb 13rd, 2025
Revised: Jan 2nt, 2025
Accepted: Jan 16th, 2025
Published: Mar 15th, 2025

KEYWORDS

Large Language Models;
Retrieval-Augmented Generation;
Optimizing chunking;
Data retrieval.

ABSTRACT

Large Language Models (LLMs) are highly effective at replicating human tasks and boosting productivity but face challenges in accurate data extraction due to prioritizing fluency over factual precision. Researchers are addressing these limitations by combining LLMs with Retrieval-Augmented Generation (RAG) models. This approach utilizes chunking, searching, and ranking algorithms to streamline data retrieval from unstructured text, improving LLMs' precision and processing. The findings provide key insights into optimizing chunking strategies and set the stage for the advancement and broader application of RAG-enhanced systems.

Doi: https://doi.org/10.61591/jslhu.20.633
Available online at: https://js.lhu.edu.vn/index.php/lachong

1. INTRODUCTION

Data mining heavily relies on retrieval as a methodology that involves understanding input queries and extracting data from external sources [1, 2]. This method is widely applied in domains such as search engines and recommendation systems to effectively sift through information and provide users with results. For example, popular search engines like Google, Bing, and Baidu serve as illustrations of retrieval tools in practice by efficiently sorting and presenting web pages or documents that match user searches. Retrieval models play a role in handling databases by providing dependable and timely information essential for knowledge-centric activities. The combination of retrieval models with language models has led to the development of Retrieval Augmented Generation (RAG) in light of the increasing prevalence of AI-generated content (AIGC) in [3,4]. This advancement boosts models by integrating data to enhance the overall quality of generated content [5]. RAG has shown effectiveness in tasks that require a lot of knowledge like Open-domain Question Answering (OpenQA) as well as in general language tasks and different practical uses. This highlights its potential, in the field of AI [6, 7].

Regular methods of browsing websites face challenges, like dealing with changing content, grasping meanings, and evolving language styles effectively. The growing reliance on JavaScript frameworks and the intricate nature of context understanding and semantics alongside the changing language landscape hinder the effectiveness of techniques. Even though web crawling offers information for research purposes and business insights in data analysis fields it's essential to use these methods, with care and ethical considerations in mind [8]. Make sure to follow the rules set by the website like the terms of service and privacy policies to respect the rights of website owners and users according to [9, 10].

Large language models (LLMs) that have been trained on amounts of text and code demonstrate abilities in grasping the subtleties and context of language usage. When used alongside customization techniques, LLMs can perform tasks like extracting data with a deeper semantic understanding, identifying relevant links intelligently, and adapting to changing content. This leads to thorough information retrieval compared to conventional approaches. Research suggests that employing LLM-based web crawling significantly reduces data collection time while improving the accuracy of data. According to market studies, LLM-driven web scraping can cut data gathering time by 25% while enhancing data precision by 10%. In e-commerce, LLMs attained a 92% accuracy rate, for pulling product details compared to the 85 % achieved using techniques [11].

Recent advancements in technology have introduced LLMs which play a role in enhancing the precision and reliability of data extraction and validation processes in the context of RAG. For instance; GPT 4 showcases skills in comprehending and crafting text that mimics writing by utilizing vast amounts of training data effectively. RAG optimizes these capabilities by merging retrieval-based techniques with models to extricate details from extensive datasets and formulate coherent responses. By incorporating retrieval methods, LLMs can tap into an array of details that improve the accuracy and significance of the information gathered. Additionally, LLMs play a role in maintaining consistency by standardizing language and formatting in the content obtained. Their ability to deliver detailed responses also ensures the entirety of the data is captured. Regarding reliability, LLMs offer performance in retrieving and verifying data thus minimizing errors and discrepancies effectively. The collaboration between LLMs and RAG techniques enhances the effectiveness of data extraction and verification processes while boosting trust in the reliability and accuracy of the extracted data.

RAG offers significant benefits for effective data extraction. Firstly, it boosts accuracy by utilising external sources of information, ensuring that the gathered data is both precise and comprehensive. This integration addresses the limitations of models that rely solely on their internal memory [12]. Additionally, RAG enhances comprehension by incorporating retrieval methods, enabling the model to better understand contextual information. This capability enriches the data extraction process, allowing the model to capture details that might otherwise be overlooked. Furthermore, RAG excels at managing dynamic content, effectively handling information that frequently changes. Unlike traditional approaches which may struggle with evolving data, RAG's ability to retrieve real-time information ensures that relevant data is extracted, which may be missed by conventional methods [13].

However, RAG also presents certain challenges. The inclusion of retrieval mechanisms introduces complexity into the data extraction process, as coordinating and aligning retrieved data with generated responses can be challenging and may require additional computational resources and sophisticated algorithms. Moreover, the success of RAG is highly dependent on the quality and reliability of the information sources it uses. If the retrieved data is inaccurate or outdated, this can adversely affect the precision and trustworthiness of the results. Finally, the process of collecting and merging data may cause delays, potentially reducing the efficiency of data retrieval in time-sensitive scenarios where rapid responses are required.

Challenges involved in data extraction using LLMs

- Challenges with Implicit Data: Linguistic models often face difficulties in handling details like connections between ideas or specialized field expertise. limitations, like these can impede their capacity to fully grasp the significance embedded in content [5].

- Limited Knowledge Scope: The AI text detector relies heavily upon the heuristics and perplexity metrics to determine whether a given piece of content has been produced by a machine learning model rather than, by a human being based upon its structures and distinctive linguistic features [11].

- Restricted Dynamic Content: Websites are using content created by JavaScript nowadays; this makes it difficult for traditional LLM based crawlers to access and extract the information because it might not be visible to

them easily which limits their capability to gather important data [13].

- Hallucination: occurs when LLMs produce information that appears believable but could be entirely false or inaccurate. A concept referred to as hallucination, in the field of AI research [14].

This dynamic environment offers both significant opportunities and challenges for data extraction. Traditional web crawling methods, which rely on predefined rules, often struggle to keep up with the rapid changes of the web, frequently overlooking relevant content. In this paper, we investigate how the integration of LLMs with prompt engineering can transform web crawling, enabling more straightforward, faster, and accurate data extraction.

Several cutting-edge pipelines have been developed that utilize LLMs to create paths for reasoning and plans in conjunction with the information retrieval (IR) component. According to Zhu et al., if the outputs from the information retrieval or the results from the LLM are subpar, in quality, the entire process of retrieving and generating information can face challenges with each stage affecting the other, they suggest an approach to tackle this problem by enhancing query quality and aligning retrieved information, with generation models through retrieval tactics [15]. The current challenge with LLMs lies in their input length, which hinders the integration of long-retrieved documents into established RAG systems. Xu et al., aim to overcome this obstacle by introducing the Retrieve, Compress, Prepend (RECOMP) method in [16]. Retrieve Compress Prepend. This involves summarizing retrieved documents before incorporating them into the generation process is an aspect discussed here by Hofstätter et al. in [17]. They address the issue of prolonged inference latency that can arise from retrieved passages during auto-decoding by suggesting a more efficient version of the FiD model that compresses encoded vectors of each passage and utilizes a re-ranker to refine the outcomes before sending them to the decoder.

A. Text Chunking in RAG

LLM-based RAG architectures face inherent operational constraints due to their contextual processing limitations. The simultaneous processing of extensive textual corpora within these models can potentially compromise their information retrieval efficacy. To mitigate these limitations, the implementation of systematic text segmentation strategies becomes imperative, wherein documents are partitioned into cognitively manageable units that preserve semantic coherence. This granular approach to information presentation enhances the model's retrieval precision, subsequently optimizing the response generation process.

The segmentation methodology serves dual purposes: it effectively moderates the computational demands on the LLM by implementing controlled information processing parameters, while simultaneously expediting the response generation mechanisms. This strategic approach minimizes cognitive noise from peripheral information, thereby enhancing both the precision and relevance of generated responses. The segmentation protocol is specifically engineered to maintain the structural integrity and logical progression of the source material, a critical consideration for advanced natural language processing tasks such as comprehensive document synthesis and query response generation, where contextual preservation is fundamental to producing coherent and substantive outputs.

To achieve optimal text segmentation, the framework incorporates the Recursive Character Text Splitting (RCTS) methodology within its LLM-based information extraction pipeline. RCTS enables efficient information processing and extraction through its iterative decomposition of web-based content into computationally manageable segments. Subsequent sections of this document elaborate on the technical specifications of the RCTS algorithm, delineating its operational mechanisms and strategic significance within the proposed information extraction framework.

B. Algorithm Breakdown

The RCTS algorithm works by methodically dividing text content using predefined character sequences until the resulting segments meet specific dimensional criteria. This systematic approach focuses on maintaining the meaning between text elements and is considered the preferred starting point for text segmentation procedures.

The algorithm starts with three inputs: a raw text string, as input a structured delimiter sequence moving from large separators (such as double line breaks) to smaller divisions (e.g., like single spaces), and a set maximum segment size. The algorithm then goes through each level of the delimiter hierarchy splitting the text at each delimiter to create text segments. The components are assessed based on their size and fit; if they meet the requirements, they become part of the group of segments but if they are too large they go through a series of steps using specific markers until all oversized parts have been dealt with appropriately before being added to the final batch as allowed units. The outcome is a set of text sections that meet standards.

The RCTS approach offers benefits for working with text effectively! One major advantage is how it keeps the context intact by focusing on punctuation marks to maintain meanings within the text – which is crucial for tasks that involve extracting information while keeping the context in mind. Additionally, the RCTS method is quite adaptable thanks to its character sequence settings (char_list) allowing for adjustments to how content is divided to meet specific needs. For example, using punctuation symbols helps create sections that work well with tokenization procedures. The recursive method improves efficiency by performing partitioning tasks needed in terms of dimensions.

However, there are a few things to think about when it comes to carrying out RCTS. One thing to consider is how much computational resources are being used up. Doing a lot of operations on documents might slow the system down overall because it needs more computing power for that kind of work. Another thing to be aware of is the risk of splitting things up setting certain configurations might make the pieces too small for some types of text. This could make it harder to do tasks that need context and may affect how well data is extracted and processed.

C. Vector Stores

Vector embeddings are a building block in architecture that helps convert text into forms in a high-dimensional vector space efficiently. This conversion ensures that text elements with meanings are represented closely in vectors making it easier to retrieve information based on similarity.

The successful implementation of data extraction methods depends on elements. Vector indexes play a role in organizing and managing data within storage systems [18]. This setup helps fine-tune index configurations to improve efficiency in both indexing and retrieval processes [19]. When dealing with documents in this context it goes beyond the text itself; it includes the text data along with their vector representations that establish a clear link, between semantic content and vector space [20]. Text embedding model interfaces play a role in converting text content into vector representations. This enables integration with embedding methods tailored to specific project needs. The incorporation of similarity search features is key to pinpointing documents that closely match query vectors and enhancing the identification of data points. Furthermore, text embedding model interfaces are essential as they convert textual data from documents into vector representations, facilitating integration with various embedding techniques tailored to specific project requirements [21]. Lastly, similarity search functionality is critical for retrieving documents whose embeddings are closest to the query vector, thereby enabling the identification of the most relevant data points for extraction [22].

The use of vector library structures tackles the restrictions of Language Models in extracting data, as depicted in Figure 1. This methodological strategy improves information retrieval by optimizing semantic similarity, resulting in improved accuracy and efficiency in data extraction methods [5, 20]. The organized process for extracting data based on vector storage includes:

Data Preprocessing Phase:

- Implementation of preliminary text processing protocols.
- Execution of data-cleaning algorithms.
- Application of linguistic processing techniques including: tokenization procedures. stop word elimination, and morphological normalization (stemming/lemmatization).

Vectorial Transformation Phase:

- Integration of preprocessed text into selected embedding frameworks.
- Generation of dense vector representations for documentary units.

Vector Indexation Phase:

- Systematic indexation of document objects.
- Integration of textual and vectorial components.
- Utilization of specialized indexing frameworks.

Query Processing Phase:

- Application of preprocessing protocols to query inputs.
- Vectorial transformation using a consistent embedding methodology.

Similarity Analysis Phase:

- Implementation of vector-based similarity search protocols.
- Retrieval of proximal vector representations.
- Application of predetermined similarity metrics.

LLM Integration Phase:

- Processing of semantically relevant retrieved documents.
- Enhanced information extraction through optimized LLM processing.
- Implementation of accuracy and efficiency optimization protocols.

In addition, the transformation pipeline within RAG architecture encompasses the decomposition of textual documents into constituent units, followed by their encoding into vector embeddings via extensively trained bi-encoder frameworks, with subsequent storage in dedicated vector databases.

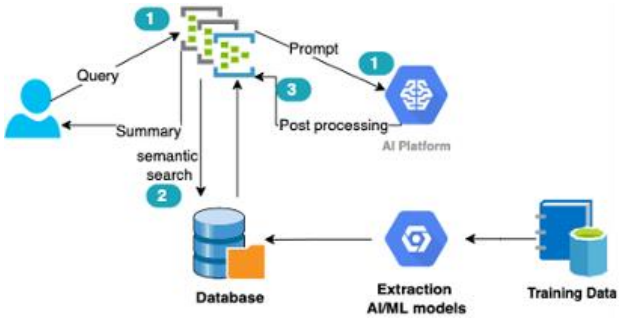

Figure 1. *The workflow for data extraction.*

2. METHODOLOGY

The process of collecting data begins with a phase of preparing the data as shown in Figure 2. This first step involves gathering website URLs to acquire primary data sources. To enhance the processing abilities of LLMs the URLs are converted into HTML content using browser rendering methods. This conversion process allows for obtaining HTML code structures containing both unorganized data elements, hence forming the basic dataset for LLM-driven analyses and information extraction tasks.

To enhance retrieval efficiency and optimize search functionality, the acquired HTML content undergoes systematic segmentation into computationally manageable units. This granular decomposition protocol facilitates more efficient LLM processing of web page components. Each resultant segment represents a discrete unit of target data designated for subsequent extraction procedures.

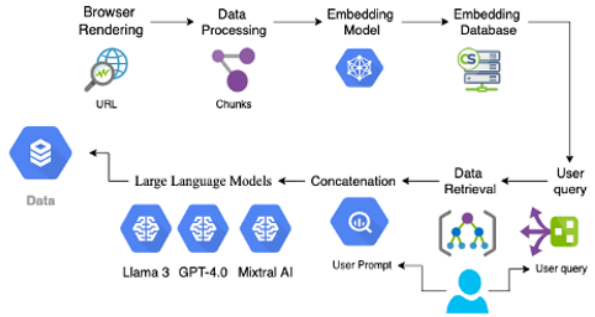

Figure 2. *The LLMs search process*

Following the segmentation phase, these data units undergo vectorial transformation through sophisticated embedding protocols. The embedding methodologies facilitate the conversion of textual data within each segment into high-dimensional numerical representations. This dimensional transformation enables enhanced efficiency in similarity computations between segmented data and user-specified extraction queries. Various embedding frameworks implement contextual mapping algorithms, transforming lexical units into vectorial representations based on their semantic relationships and distributional patterns within the corpus. The embedding of segmented data facilitates the construction of searchable indices, enabling LLM-based identification of relevant information-containing segments.

The management of unstructured data predominantly employs vectorial transformation and storage methodologies. During query processing, the search query undergoes parallel vectorial transformation, facilitating the retrieval of maximally similar embedding vectors. Vector storage architectures orchestrate both the archival of embedded data and the execution of vector-based search operations. The generation of vectorial representations confers significant advantages in conceptualizing textual relationships within vector spaces, enabling advanced information retrieval operations, particularly in semantic search applications where textual similarity is evaluated through vectorial proximity metrics.

The embedding vectors are saved in a data structure called the Facebook AI Similarity Search (FAISS) vector store for retrieval based on vector similarity efficiency. When a request is made to extract a field, it is also transformed into an embedding vector. Subsequently, the system conducts a similarity search within the FAISS vector store to locate the k segments (where k represents a predetermined number), showing the semantic resemblance to the query vector. After gathering the k chunks that closely match the field of interest in terms of meaning within the HTML content, combining them into a unit helps the LLM analyze these relevant chunks more efficiently as a whole.

With this approach, customized prompts are created to help the LLMs extract information from the HTML by breaking down data points into chunks. These prompts offer guidance and details that outline the required details within the linked segments. By utilizing its natural language comprehension skills, the LLM can decode these prompts effectively. Explore the HTML content allowing for reliable data extraction.

To prevent hallucinations that may arise from the LLMs outputting process we adopted a method that combines three different LLMs to leverage each model's advantages and minimize the chances of introducing biases or errors.

- MixtralAI was chosen for its established success in tasks that require retrieving information and delivering performance at a cost-effectiveness ratio.

- OpenAIs GPT 3 has proven to be highly proficient, in natural language processing tasks, like information extraction and generating data outputs that can easily integrate into databases using prompt template limitations.

- Llama 3 offers a mix of retrieval and generation abilities that prove useful in situations that demand the integration of data and the creation of informative text.

To boost accuracy levels effectively in the process. Ensure precision enhancement was achieved successfully for each piece of data processed by the LLM model by employing an ensemble voting technique post-analysis stage completion of each LLM's task independently using a system that considered various factors, like the credibility of the extracted information and its occurrence frequency across multiple LLM outputs and the overall quality metrics including completeness and internal consistency validation, against a benchmark dataset to determine accuracy levels precisely. The ultimate choice was made by picking the option with the number of votes from the ensemble.

Analyzing sentence embeddings and determining similarities between sentences is enhanced through chunking which group sentences together based on their closest embeddings. A method that significantly boosts the quality of retrieval by ensuring semantic coherence in texts and proving particularly beneficial, in scenarios where maintaining meaning and context is crucial. The semantic chunking core idea revolves around utilizing sentence embeddings to form segments of information. The process includes:

1. Splitting the document into sentences based on punctuation marks like periods, question marks, and exclamation points.

2. Indexing each sentence according to its position in the text.

3. Grouping sentences by selecting a certain number of sentences around a target sentence, with additional buffer sentences on either side.

4. Calculating the similarity between these groups of sentences.

5. Merging similar groups while separating those that are not, to ensure coherence within each chunk.

The development of agentic systems within RAG pipelines represents a significant advancement, marked by their proactive and self-regulating features. These agentic RAG systems incorporate sophisticated artificial intelligence, allowing them to grasp the intent behind user queries, comprehend the semantic organization of data, and determine the necessary degree of data masking. Such systems reflect a leap forward in creating intelligent,

context-aware mechanisms that enhance the efficiency and accuracy of data retrieval and processing.

In essence, the upcoming advancements in data preprocessing rely on blending comprehension of meaning, manipulation or enhancement of data flexible but uniform rule-based authorization control. This mix lays the groundwork for a breed of systems known as RAG systems set to broaden the horizons of data handling and information retrieval. Moreover, incorporating constitutional access control, in these setups guarantees that retrieving and synthesizing information strictly follows governance guidelines steering clear of the constraints posed by attribute scalable-centric approaches. Agentic RAG systems, with this type of access control can independently handle data flow to provide responses that are contextually precise and linguistically clear while also being mindful of privacy and adhering to policy guidelines.

Table 2. *Semantic Chunking for a document in [23]*

Question	Answer	Contexts	Ground_truth	Context_precision	Faithfulness	Answer_relevancy	Context_recall
What special symbols are added to...	The special symbols added to the input example...	[BERT BERT InE[CLS] E1 ESEP ... ENE1 ... EM...	The special symbols added to the input example...	1	1	0.91007	1
According to the passage, what is o...	One example of a supervised task with a large ...	2018; Radford et al., 2018; Dai and Le, 2015)...	The passage mentions that transfer learning fr...	1	1	0.93277	1
Based on the provided context, how ...	The BERT model architecture has a number of la...	[arate fine-tuned models, even though they are ...	Based on the provided context, the BERT model ...	1	1	0.87787	1
Why did the developers choose BERT ...	The developers chose BERT BASE to have the sam...	TH, and the number of self-attention heads as ...	The developers chose BERT BASE to have the sam...	1	1	0.91932	1
How does the input representation of..	The input representation of BERT unambiguously..	[Input/Output Representations To make BERTinha...	The input representation of BERT unambiguously...	1	1	0.91839	1
According to the passage, what are t...	The two unsupervised tasks used to pre-train B...	[ding to every token indicating whether it bel...	The two unsupervised tasks used to pre-train B...	1	1	0.90531	1
How can a deep bidirectional repres...	A deep bidirectional representation can be tra...	[standard conditional language models can only...	A deep bidirectional representation can be tra...	1	1	0.88818	1

3. RESULTS

Based on the literature presented, several key questions must be addressed to ensure the success of web scraping. Figure 3 illustrates the operational process based on graph-based pipelines. Each pipeline comprises a sequence of nodes, with each node performing a specific function in the data collection and processing workflow.

Figure 3. *The process based on graph-based pipelines.*

Firstly, create a data scraper by setting up prompts to request specific details from JSON data, such as product names, average ratings, and prices in Figure 4. The output will display with the user prompt as “List all product’s name along with its average rating and price based on the reviews” in Figure 5.

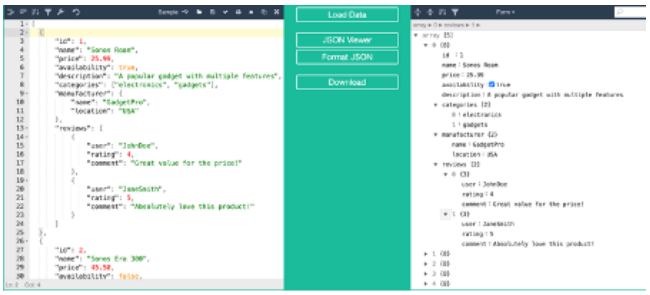

Figure 4. *Demonstrate file JSON data.*

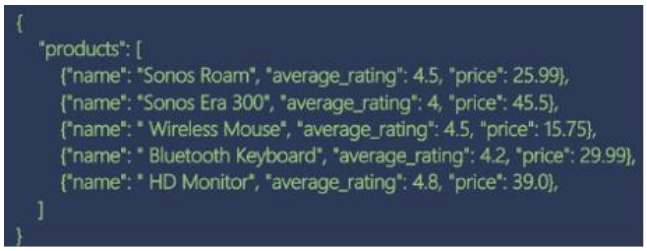

Figure 5. *The output product's name with rating and price on local file*

In addition, semantic chunking carefully considers the internal relationships within a text, dividing it into semantically complete and meaningful segments. This approach ensures that the integrity of the information is maintained during retrieval, leading to accurate and contextually appropriate results. The hypothesis behind this method is that it closely mimics the way humans naturally process documents.

In our study, we implemented semantic chunking and recursive retrieval on the document available in [23]. The findings, presented in Table 2, demonstrate that Semantic Chunking effectively organizes contextually similar information into distinct and coherent segments. This technique significantly boosts the efficiency and effectiveness of large language models by supplying them with focused and relevant inputs, thereby enhancing their ability to understand and process natural language data. The emergence of RA-LLMs has significantly impacted various areas of software engineering. For instance, some research has introduced the retrieval-augmented generation approach for tasks like code generation and program repair. Moreover, RA-LLMs have shown promise in processing tabular data and improving text-to-SQL semantic parsing.

Table 1. *Scraper Graph*

node_ name	total_t okens	prompt_ tokens	successful_ requests	total_ cost	exec_ time
Fetch	0	0	0	0	6.385
Parse	0	0	0	0	0.069
RAG	0	0	0	0	2.502
Generate Answer	0	0	1	1	28.340
Total	0	0	1	1	37.296

4. FUTURE CHALLENGE

RA-LMMs are transforming the finance industry by integrating external data sources like Bloomberg, Twitter, and Reddit to improve sentiment analysis and decision-making [24]. They enhance financial question answering through PDF parsing and structured document chunking [25], [26]. While Wikipedia aids RAG systems, its variable reliability highlights the need for improved data quality. Future advancements will focus on semantic understanding, intelligent data augmentation, and adaptive access control, paving the way for next-generation RAG systems. With its articles covering subjects, Wikipedia offers comprehensive information [27]. However, the dependability and correctness of articles may differ greatly. The presence of incorrect or deceptive details could impact the model's results quality. To tackle this issue effectively and make progress in refining the knowledge base while reducing the effects of poor-quality data on LLM performance is crucial for improvement in RA-based LLM systems output accuracy and reliability across different practical uses. Ultimately, the future of data preprocessing lies in the integration of semantic understanding, intelligent data masking or augmentation, and adaptive policy-driven access control. This combination forms the foundation of a new generation of RAG systems poised to transform the landscape of data processing and knowledge retrieval.

5. CONCLUSION

This research enhances LLM-based data extraction by fine-tuning models with domain-specific knowledge, enabling precise and efficient web scraping. It introduces constitutional access control for contextually accurate and privacy-aware responses. Future advancements will focus on semantic understanding, intelligent data masking, and policy-driven access control to improve flexibility, real-time processing, and personalized retrieval in RAG systems, driving innovation in automated information retrieval.

6. REFERENCES

[1] Mei Kobayashi and Koichi Takeda, "Information retrieval on the web", *ACM Comput. Surv*, 32(2), pp.144–173. **2010**.

[2] Yujuan D. *et al.*, "Discriminative dual-stream deep hashing for large-scale image retrieval", *Information Processing & Management*, 57(6), **2020**.

[3] Jiatong L. et al, "Empowering Molecule Discovery for Molecule-Caption Translation with Large Language Models: A ChatGPT Perspective", *IEEE Transactions on Knowledge and Data Engineering*, pp. 1-13, **2024**.

[4] Junda Wu, *et al.*, "CoRAL: Collaborative Retrieval-Augmented Large Language Models Improve Long-tail Recommendation", *arXiv preprint arXiv:2403.06447*, **2024**.

[5] Lewis P. *et al.*, "Retrieval-augmented generation for knowledge-intensive NLP tasks", *In Proceedings of the 34th International Conference on Neural Information Processing Systems (NIPS '20)*. Curran Associates Inc., Red Hook, NY, USA, Article 793, 9459–9474.

[6] Hongjin S. *et al.*, "Selective Annotation Makes Language Models Better Few-Shot Learners", *arXiv preprint: 2209.01975*, **2022**.

[7] Liu, S. et al, "Multi-modal molecule structure–text model for text-based retrieval and editing", *Nature Machine Intelligence*, 5(12), pp. 1447–1457, **2023**.

[8] Vlad Krotov, Leigh Johnson, "Big web data: Challenges related to data, technology, legality, and ethics", *Business Horizons*, 66(4), pp. 481-491, **2023**.

[9] Han S. et al, "Web Scraping for Hospitality Research: Overview, Opportunities, and Implications", *Cornell Hospitality Quarterly*, 62(1), pp. 89-10.4, **2021**.

[10] Andrews, J. T. et al, "Ethical considerations for collecting

human-centric image datasets". *arXiv preprintarXiv:2302.03629*, **2023**.

[11] Brown, T. et al, "Language Models are Few-Shot Learners", *arXiv*, **2020**.

[12] Lewis, P. et al. "Retrieval-Augmented Generation for Knowledge-Intensive NLP Tasks", *arXiv*, **2020**.

[13] Zhang Y., et al, "A Survey on Retrieval-Augmented Language Models", *Journal of Machine Learning Research*, 1, pp. 1-35, **2020**.

[14] Radford A., et al, "Language Models are Unsupervised Multitask Learners", *OpenAI*, **2019**.

[15] Yin Z. et al, "Furthest Reasoning with Plan Assessment: Stable Reasoning Path with Retrieval-Augmented Large Language Models", *arXiv preprint arXiv:2309.12767*, **2023**.

[16] Fangyuan X. et al, "RECOMP: Improving retrieval-augmented LMs with context compression and selective augmentation", *arXiv:2310.04408*, **2023**.

[17] Sebastian H. et al, "FiD-Light: Efficient and Effective Retrieval-Augmented Text Generation". *In SIGIR. ACM*, pp. 1437–1447, **2023**.

[18] Devlin J. et al, "BERT: Pre-training of deep bidirectional transformers for language understanding", *arXiv*, **2019**.

[19] Johnson J. et al. "Billion-Scale Similarity Search with GPUs", *IEEE Transactions on Big Data*, 7(3), pp. 535-547, **2021**.

[20] Lewis P. et al. "MLQA: Evaluating cross-lingual extractive question answering", *in Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pp. 7315-7330, **2020**.

[21] Reimers N. et al, "Sentence-BERT: Sentence embeddings using Siamese BERT-networks", *in Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing*, pp. 3980-3990, **2019**.

[22] Manning C. D. et al, "Introduction to Information Retrieval", *Cambridge University Press*, **2008**.

[23] Jacob D. et al, "BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding", *in Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, NAACL-HLT 2019*, Minneapolis, MN, USA, **2019**.

[24] Boyu Z. "Enhancing financial sentiment analysis via retrieval augmented large language model", *in Proceedings of the Fourth ACM International Conference on AI in Finance*, pp.349–35, **2023**.

[25] Demiao L, et al, "Revolutionizing Retrieval-Augmented Generation with Enhanced PDF Structure Recognition*", arXiv preprint arXiv:2401.12599*, **2024**.

[26]Antonio J. Y. et al, "Financial Report Chunking for Effective Retrieval Augmented Generation", *arXiv preprint arXiv:2402.05131*, **2024**.

[27] Yinghao Z. et al, "REALM: RAG-Driven Enhancement of Multimodal Electronic Health Records Analysis via Large Language Models", *arXiv preprint arXiv:2402.07016*, **2024**.