# Learning Quantum Computing

*Ka Chun Ng*


Master of Science

School of Informatics

University of Edinburgh

2024

# Abstract

Quantum computing represents a significant and promising technology with the potential to significantly enhance computational performance in a range of fields, including finance, biology and artificial intelligence. The growing interest in this emerging field has led to an increase in the complexity and number of concepts and terminologies, which can be challenging for both novice and experienced researchers to navigate. Consequently, a series of RAG systems have been constructed which utilise a multitude of cutting-edge techniques, including auto-merging, knowledge graph and multi-query, with the objective of synthesising knowledge pertaining to quantum computing for the purpose of context retrieval in response to user queries, facilitated by the use of a Large Language Model (LLM). The systems are evaluated using the RAGAs framework and the expertise of a quantum computing specialist. In the course of the experiments, it became evident that the traditional reranking strategy of reciprocal rank fusion is unsuitable in the context of dense retriever. Conversely, the knowledge graph was demonstrated to facilitate the existing system in context retrieval. In general, the straightforward vector search system (SV) demonstrates satisfactory performance in the LLM-generated test set, whereas the system that employs both vector search and knowledge graph (KV) excels in the user study. Overall, 40% of the responses to the user query are rated 8 or above out of 10 across the systems.

# Research Ethics Approval

This project was planned in accordance with the Informatics Research Ethics policy. It did not involve any aspects that required approval from the Informatics Research Ethics committee.

# Declaration

I declare that this thesis was composed by myself, that the work contained herein is my own except where explicitly stated otherwise in the text, and that this work has not been submitted for any other degree or professional qualification except as specified.

(*Ka Chun Ng*)

# Acknowledgements

I would like to thank the University of Edinburgh and my supervisor Dr Chris Heunen for their guidance and support in completing this dissertation.

# Table of Contents

# Chapter 1

# Introduction

## 1.1 Motivation

In the context of classical computers, data is represented by a binary system comprising two states: 0 and 1. As the search space increases, the data and time complexities grow exponentially. In the 1980s, Benioff and Feynman provided a theoretical definition of quantum computers. [5][16] In 1985, Deutsch put forth a rigorous foundation for quantum computing and algorithms. [10] The capacity for information encoding as qubits (0 and 1) represents a significant advantage of quantum computers, which leverage quantum physics concepts to reduce time complexity significantly in comparison to classical computers. [28] In the 1990s, Shor's prime factorisation algorithm and Grover's search algorithm demonstrated the superiority of quantum computers over classical computers in the field of cryptography and database. [30] The research community was prompted to investigate further potential applications of quantum computing in a wide range of fields, including material science, engineering, logistics [4] and machine learning[65].

In recent times, artificial intelligence has come to dominate the market and research interests in the technology sector, as a consequence of its proven success in a number of different areas, including natural language processing (NLP) and image processing. In particular, there has been a notable increase in research interest surrounding large language models (LLMs) and their applications. [42] The Retrieval-Augmented Generation (RAG) approach was introduced with the objective of leveraging information retrieval techniques on a massive scale to address a range of domain-independent Natural Language Processing (NLP) tasks. [29] At the same time, the intricate and vast quantity of data inherent to quantum computing renders it challenging for both novice users and researchers to attain a comprehensive grasp of the subtleties inherent

to its concepts and protocols. In response to this challenge, this research is dedicated to developing a structured system to facilitate the resolution of questions pertaining to quantum computing.

## 1.2 Problem Statement and objective

As the rate of publications for quantum computing grows rapidly over the past two decades from less than 1000 per year to more than 4000 per year in 2021 [13], the depth and scope of research interests keep expanding. This explosion of information has led to challenges such as semantic overlap, synonymous terminology, and content duplication, making it increasingly difficult for researchers to stay updated or gain a deep understanding of the field quickly.

For those new to the field, the highly technical research papers on quantum computing are frequently challenging to grasp without a firm grasp of the terminology and concepts that form the foundation of the subject. The vast quantity of literature on the subject can make it difficult to identify pertinent and fundamental information that would enable one to comprehend the latest research. This creates a significant learning curve, particularly for those seeking to stay up-to-date on current developments, and it greatly diminishes their interest and accessibility to the subject.

To address these challenges, this research aims to develop a structured RAG system to build a structured system to synthesise high-level details and abstractions for current or future literature of quantum computing in order to bridge the knowledge gap between system and target users. This will entail the synthesis of data and the construction of a retrieval system, which will then undergo evaluation. Furthermore, research and analysis on RAG techniques will be conducted to provide insights on the creation of an improved knowledge retrieval system for quantum computing papers.

## 1.3 Contributions

This project has developed several RAG pipelines for retrieving contexts from query in order to answer user's query of quantum computing knowledge. At last, 3 superior systems are selected and give good results in real user study. Combinations of techniques were used in the systems for evaluating its effectiveness as shown in table 1.1.

| Category | Techniques |
|---|---|
| Preprocessing | Structured text parsing, Non-structured text parsing |
| Indexing | Knowledge Graph, Auto-merging, Vector Index |
| Post-processing | Re-ranking, Fusion |

Table 1.1: Techniques used in the RAG pipelines for retrieving contexts from queries.

This study employs a diverse range of RAG system configurations to evaluate the effectiveness of knowledge retrieval and the quality of responses based on metrics judged by LLM, namely faithfulness, answer relevancy, context precision, context recall and answer correctness. The generated datasets have been subjected to a comprehensive analysis in conjunction with the human datasets, with the objective of elucidating the characteristics and limitations of each system, filling the research gap of the latest advancement of RAG techniques. The findings of this analysis will inform the selection of the pipeline structure and future research directions, particularly with regard to datasets utilised in academic research.

## 1.4 Structure

The remaining content of the paper is described as 4 parts:

1. Background and related study for better understanding in reading this paper
2. Methodology adopted in this research, which includes implementation details in the stage of dataset preparation, prepossessing and RAG system.
3. Results and discussion for the current research in the RAG systems implemented with the outcomes delivered as well as the limitations
4. Conclusion on the dissertation, which gives summary, limitations and further research directions to work on

# Chapter 2

# Background

## 2.1 Information retrieval

In the initial stages of information retrieval, Boolean systems utilising an exact match of words were employed to generate non-ranked results of a relatively low quality. In contemporary information retrieval systems, two models are frequently employed: probabilistic models and vector space models. [67] The probabilistic model was trained to model the probability of relevance for documents according to the query [63], resulting in a highly biased and inconsistent approach. In contrast to other search models, the vector space model represents text as a vector of terms in a multidimensional space, offering flexible structure and compromise results. [49] In the present era, transformer-based models such as BERT[11] have demonstrated superior retrieval performance to that of state-of-the-art traditional sparse retriever BM25[24] through the formation of information-rich dense vectors. [27] The model effectively encoded both the semantic and syntactic information of words into word embeddings [1] and captured the contextual relevance between the query and the documents through the angle difference of the respective vectors. [2].

## 2.2 Retrieval-Augmented Generation

Large language models represent a significant advancement in the field of natural language processing and artificial intelligence, exhibiting remarkable proficiency in language-oriented tasks. Nevertheless, it exhibits a tendency to generate implausible and inaccurate content, particularly in the context of domain-specific knowledge, which is often underrepresented in the training datasets. [25] This phenomenon is commonly

referred to as ”hallucination” [20, 45]. In 2020, retrieval-augmented generation (RAG) was proposed as a means of mitigating the issue of ”hallucination” and providing answers that are more factual and relevant, drawing on updated world knowledge from external document indexes. [29] The findings of the research indicate that RAG was a more resilient and effective method for knowledge injection in comparison to another prominent approach, which involves fine-tuning the pre-trained model. [48] At present, the most advanced RAG methods include naive RAG, advanced RAG and modular RAG. In the case of advanced RAG, the pre-processing and post-processing steps are incorporated into the pipeline in order to enhance the quality of the index and the retrieved context. In modular RAG, a flexible and multifunctional pipeline architecture is provided by the incorporation of combinations of functional modules, resulting in refined outcomes. [18] To illustrate, the RAG Fusion approach[62] incorporates query rewriting modules[40] that prompt LLM to generate a new query to facilitate comprehension of the query inputs and re-ranking algorithms, including reciprocal rank fusion(RFF)[9] to optimise the ordering by relevance between the query and retrieved documents.

## 2.3 Knowledge graph

Knowledge Graph(KG) is a semantic representation of data, comprising nodes and edges. The edges indicate the relationships between the nodes, which typically represent concepts or entities. [77] This structured graph can enhance the retrieval of information and reasoning by tracing paths with related nodes. The retrieval of accurately contextualised data can mitigate ”hallucination”, thus improving the quality of responses for LLMs in specific domains. [72, 74, 45]

## 2.4 Retrieval Augmented Generation Assessment

Retrieval-Augmented Generation Assessment (RAGAs) framework provides a systematic method for evaluating retrieval or answer generation modules without the need for reference data sets and achieve close alignment with human annotated results. By generating synthetic datasets based on specific characteristics of designated questions, such as multi-context or reasoning, the workload involved in manual test-set creation and collections can be significantly reduced. To assess the coherence and relevance of the generated answers, provided contexts and ground truths(the model answers generated

by the selected contexts), metrics such as Faithfulness, Answer Relevance and Context Relevance are employed. [14, 15]

## 2.5 Related work

In the preceding study of RAG systems, researchers have compared relatively simple and traditional pipelines as baselines. These pipelines typically employ simple chunking with a direct match for dense retrievers. The majority of papers focus on a limited number of target subsets, such as the fusion process[17], or the pre-retrieval process[40, 50]. The latest developments in indexing techniques and details are not addressed in the formal research. In the case of knowledge graph-assisted pipelines, there have been notable advances in testing datasets[19] and downstream tasks[76, 75, 26]. However, the construction of knowledge graphs is typically only briefly described, depending on the research domain, or even disregarded. [51] In the research on RAG pipelines, combinations of chunking methods, indexing techniques and re-rankers are evaluated, but the results vary depending on the datasets used. To illustrate, in the dataset HotpotQA[78], SQUAD[61] and QuAC[8], simple sentence parsing performs the best out of all chunking methods in recall. [68] In constrast, in the dataset of arXiv[3] collections, sentence window (A technique to embed chunks of text as nodes with the surrounding text as window. The content within the window is combined with that of the retrieved node, thereby providing more complete context. [66]) chunker achieves highest score in precision. [12] The selection of modules in each stage of the RAG pipeline is contingent upon the nature and structure of the documents in question. To evaluate the practical performance of RAG pipeline variations in the context of quantum computing research papers, we are utilising LlamaIndex[35], a sophisticated framework for building document indexes for LLM applications, throughout the project.

## 2.6 Hardware and software setup

The hardware setup for this project composed of 1 linux server and 2 windows computer.

| **Computer** | **Specification and Usage** |
|---|---|
| Win10 Desktop | 1TB HDD, 1TB SSD, 24GB RAM, GTX1060, i7-7700<br>**Usage:** MongoDB[41] database, Neo4j[44] database, Milvus[71] database |
| Linux server 3 | 3TB SSD, 125GB RAM, RTX3090 X2, i9-7900X<br>**Usage:** Hosting server for LLM services |
| Win11 Laptop | 2TB SSD, 36GB RAM, RTX3070ti, i7-12700H<br>**Usage:** Embedding model, project coding and execution |

Table 2.1: Comparison of computer specifications and their usage for data parsing.

In consideration of the constraints imposed by the hardware and budgetary limitations, the selection of the embedding model and LLM will be based on open-source models that demonstrate memory efficiency, as well as their position on the MTEB leaderboard [43], LMSYS Chatbot Arena [7] and the time of their release. n accordance with the aforementioned criteria, the Qwen2-7B model [54] and the gte-large-en-v1.5 model [79, 31] were selected as the default LLM and text embedding model respectively. Both models demonstrate the most cost-effective performance at the time of implementation, outperforming other homogeneous models evaluated on various public datasets in the majority of tasks. With regard to rerankers, we will assess the algorithmic RFF and jina-reranker-v2-base-multilingual as Jina provides a limited free API service of a 137B model with excellent performance in a range of benchmarks [23]

In order to facilitate the implementation of the project, the main tool employed is Llamaindex[35], an open-source package that provides a range of tools and integrations for the construction of RAG systems.

# Chapter 3

# Methodology

This chapter will present the details of the implementation and experiments conducted for the RAG systems in order to respond to queries pertaining to quantum computing. The overall data ingestion and indexing flow is described below:

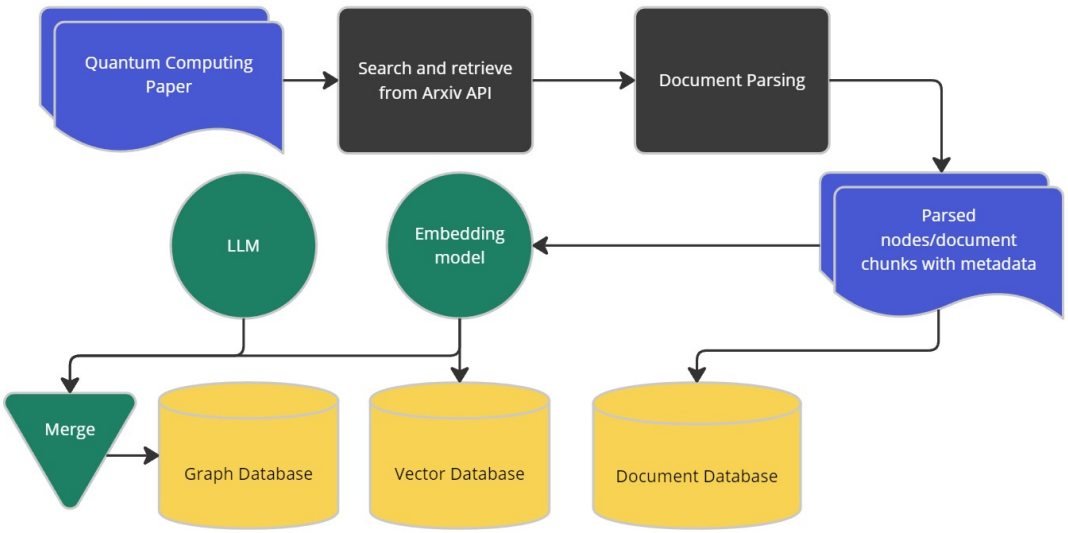


Figure 3.1: Data ingestion and indexing pipeline

## 3.1 Dataset Preparation

In order to retrieve academic researches of quantum computing, we utilise the arXiv api to collect pre-print scholars based on the relevant search phrases. Initially, 10 search phrases(quantum computing, quantum algorithm, quantum error correction, quantum cryptography, quantum communication, quantum network, quantum internet, quantum

machine learning, quantum programming and quantum optimization) for top 1000 relevant papers were collected. Due to time constraint in processing documents, 996 papers (4 of the archive are deleted from the site) searched by ”quantum computing” were adopted as ingestion data with metadata. Metadata includes url, updated/published date, title, authors, summary, comment, journal reference, doi, primary category and categories. The downloaded pdf of documents have high discrepancy in terms of text length as the paper content (incomplete content) and layout varies (incorrect parsing by pymupdf4llm[53] library). The statistics for character length parsed is described below.

| **Statistics** | **PDF character count** | **metadata character count** |
|---|---|---|
| mean | 74835.8 | 1421.68 |
| median | 45682.0 | 1359.5 |
| min | 456.0 | 516.0 |
| max | 5257322.0 | 2610.0 |
| std | 192432.09 | 465.71 |

Table 3.1: Statistics for character count of collected data

The noise data are kept in the dataset to ensure robustness retrieval, such as the document that only produces 456 characters with majority symbols.

| **Properly parsed markdown** | **Badly parsed markdown** |
|---|---|
| —— | —Finite String of Rules—Current State—current symbol scanned— |
| ##### 2.48 Grover’s Algorithm: Initialization | —e—Col2—Col3—Col4—Col5—e e— ———————— ———————— ———— ——— ——— ———————— ————————p— —e————————e— |
| State preparation: At the beginning of the protocol, there is no information about where the marked item is. Thus, it is convenient to start with an equal superposition state | |

Table 3.2: Sample of parsed markdown from quantum computing research

## 3.2 Prepossessing

There are two principal categories of software for parsing PDF files into text: vision model parsing (VMP) and heuristic parsing (HP). In the context of machine learning vision models, the AI vision model is employed to perform a range of tasks, including the detection of layouts, the parsing of equations and tables. The software is capable of generating satisfactory results with images, tables and equations as markdown, but the inferencing process for a single PDF document can take several minutes, and not all items are recognised or formatted correctly due to the inconsistent formatting of PDFs. In heuristic parsing, the layout is identified through the application of algorithms that utilise rules to determine the margins and recognise the texts and tables. Images and equations necessitate the implementation of customised handling procedures to facilitate the incorporation of the parsed content. The parsing process is highly expeditious, with the capacity to complete the parsing of an entire PDF within a single second. [53] While it can still achieve satisfactory results for text extraction, it is unable to parse images, tables, and equations on its own. There are some closed-source services that provide high-quality, limited free parsing services, which are not within the scope of this study. Given the limitations of time and cost, pymupdf4llm (Based on pymupdf package, which has been demonstrated to have the fastest extraction speed and the second-best text extraction quality among the most commonly used Python PDF parsing libraries.[53]) was selected as a compromise between speed and effectiveness, offering markdown conversion for PDFs with table support. The following table presents an overview of the parsing methods that were investigated.

| Name | Type | Speed | Quality | Comment |
|---|---|---|---|---|
| nougat[6] | VMB | 2.6 seconds per page of arXiv papers in A6000 ada machine[70] | Excellent | Support parsing of images, tables and equations into markdown |
| marker-pdf[70] | VMB | 0.63 seconds per page of arXiv papers in A6000 ada machine[70] | Excellent | Same support as nougat |
| pymupdf/ pymupdf4llm[53] | HP | 0.1 seconds per page[52] | Good | Support text parsing with moderate level table parsing quality |
| LlmSherpa[46] | HP | slightly slower than pymupdf | Good | More unstable than pymupdf in parsing content |
| LlamaParse[36] | VMB | similar to maker-pdf | Excellent | Same support as nougat |

Table 3.3: Overview of PDF parsing methods

## 3.3 Retrieval System

This section will examine the techniques employed in the modules of a retrieval system, categorised into four distinct parts. **Text segmentation**, **Indexing**, **Retrieval**, **Systems Overview** The following figure shows an overview for the whole pipeline of the system:

### 3.3.1 Texts segmentation

#### 3.3.1.1 SentenceSplitter

In order to circumvent the division of words into arbitrary characters and to maintain the structural integrity of paragraphs, SentenceSplitter[39] is employed to partition the texts into discrete nodes, according to the regular expression (regex) of punctuation, with a specified maximum number of tokens.

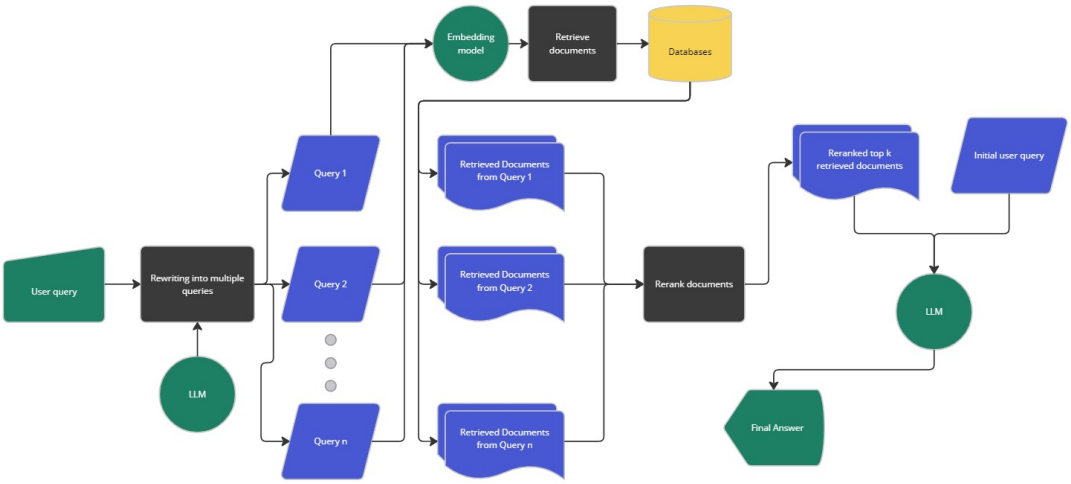


Figure 3.2: System pipeline for QA system

#### 3.3.1.2 MarkdownNodeParser

The parsed markdown file was employed to divide the nodes while retaining the structural information of the markdown file through the annotations of headings, tables, and lists. [37]

#### 3.3.1.3 HierarchicalNodeParser

The hierarchical node parser parsed the texts into levels of nodes with a defined number of overlapping nodes to avoid information loss. The nodes at the top of the hierarchy will have the highest token coverage, while each of them consists of a number of low-level nodes which have smaller text segments within the scope of the parent node. [34]

### 3.3.2 Indexing

#### 3.3.2.1 Vector Index

The process of embedding nodes into vectors will be conducted through the utilisation of the embedding model. Following this, the embedded vectors will be stored in the vector database for subsequent retrieval. In order to ascertain the degree of similarity between the vectors, cosine similarity will be employed. This methodology facilitates the identification of documents exhibiting a similar content to the given vectors.

#### 3.3.2.2 Property Graph Index

Property Graph [38] has been developed with the objective of providing support for graph stores with embedding capabilities, such as Neo4j. The entities created can be retrieved through a similarity search of vectors with tracing paths. In this project, the entities and relations are created and connected with the assistance of LLM and a predefined schema, one by one for each node using the utilities of Llamaindex. Furthermore, the original path hierarchy is preserved through the construction of relationships between nodes, including those of the parent, child, and sibling types. The construction process is dependent on a considerable number of time-consuming LLM calls, with the entire process taking in excess of 200 hours to complete the synthesis of 996 academic papers.


Figure 3.3: Part of the knowledge graph built from quantum computing papers

### 3.3.3 Retrieval

#### 3.3.3.1 Base vector retrieval

In the context of base vector retrieval, vector index of all nodes within the vector database are loaded into memory in order to enable a comparison to be made between the question under scrutiny and the database as a whole. Those nodes which are deemed

to exhibit the greatest degree of resemblance to the question are subsequently chosen for inclusion in the retrieved contexts.

#### 3.3.3.2 Auto-merging retrieval

In regard to the auto-merging retrieval [33] process, the HierarchicalNodeParser is utilised to generate a tree-like structure with a top-to-bottom organisation of nodes. The root level represents the uppermost level of the hierarchy, encompassing the collection of nodes with the greatest size of chunk. In contrast, the nodes at the leaf level exhibit the smallest size of chunk. The leaf nodes are embedded in the vector database for searching. When the nodes are retrieved, nodes will try to group with sibling nodes. A group of nodes will be merged into a single parent node recursively if the defined threshold is reached. An example figure is shown below:

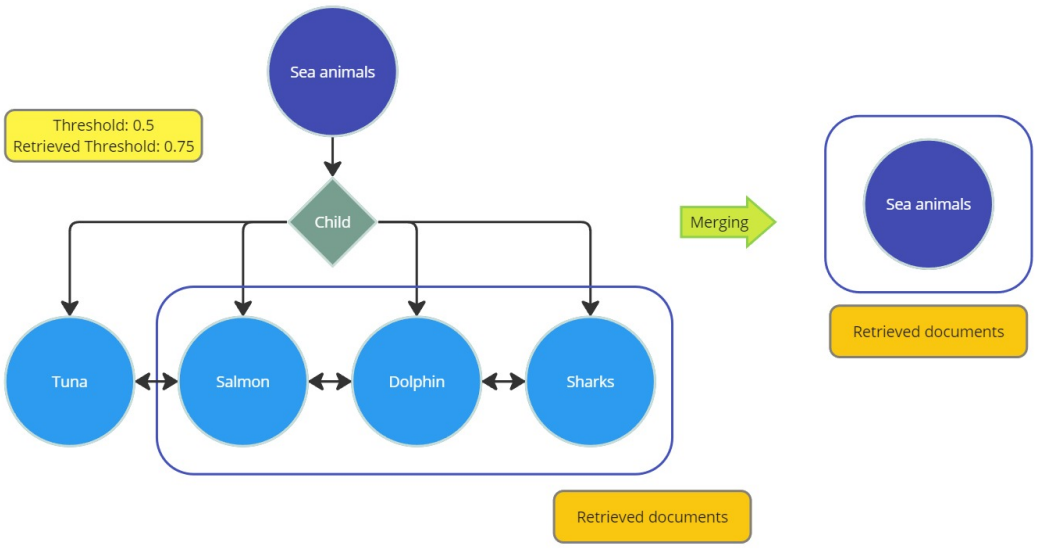


Figure 3.4: Data ingestion and indexing pipeline

This approach enables the merging nodes to construct more comprehensive and consistent contexts, rather than unstructured and fragmented text segments. Its fundamental premise is analogous to that of the Sentence Window retriever, however it does not necessitate the provision of the contextual surroundings of retrieved chunks, thereby minimising the influence of extraneous information with a concentrated context.

#### 3.3.3.3 Pre-retrieval and post-retrieval

In order to enhance search convergence and query quality, LLM is employed to generate multiple queries. However, due to the phenomenon of ”lost in the middle”[32], which

causes the performance of LLM to degrade with the increase of context position of relevant documents, a solution is required. Consequently, the retrieved results from the system are then reranked by ranker or reciprocal rank fusion in order to obtain the top k relevant documents from the query. An attempt is made to rerank using LongLLMLingua[21], a tool that utilises well-trained LLM for prompt compression and reranking to improve RAG performance. The LLM LLaMA-7B, which had been specifically trained for the framework in the research, required 7.8 GB of VRAM for an 8-bit quantised model to process the retrieved contexts. The average processing time for a single document was 5 minutes. The QA result was not as effective as that of the reranker in a small testing case, due to limitations in hardware, time, and performance. Therefore, it is no longer recommended as an option for this task.

### 3.3.4 Systems Overview

As the pre-pretrieval and post-retrieval steps are not contingent on the indexing and retrieval structure, the following table illustrates the parsers, indexes, and retrieval employed for a system designed for a single query. In total, three systems have been constructed for this purpose.

| **System** | **Parser** | **Index** | **Retrieve** |
|---|---|---|---|
| SimpleVector (SV) | Sentence | VectorIndex | Vector search |
| Automerge (AM) | Hierarchical | VectorIndex | Vector search<br>Auto-merging |
| KnowledgeVector<br>(KV) | Markdown<br>Hierarchical | PropertyGraph<br>Index<br>VectorIndex | Vector search<br>Automerging<br>context search |

Table 3.4: Overview of PDF parsing methods

For AM system, the hierarchical parser divides the documents into three levels, with each level encompassing a chunk size of 128, 512, and 2048, respectively. For Sentence and Markdown parser, the texts are divided of size 512. For SV and AM system, each query will result in the retrieval of 10 relevant nodes. In the case of the KV system, the auto-merge retriever is employed to retrieve document chunks. These chunks are then subjected to a search of the vector context nodes, utilising metadata filtering and embeddings similarity. The retrieved vector context nodes from the knowledge graph

contain paths to the neighbouring nodes of depth 1, with the relationships between them serving as the content. In the proposed system, a maximum of 10 vector context nodes will be retrieved if the similarity exceeds 0.5. Consequently, a maximum of 20 distinct nodes will be retrieved from the KV system in total.

#### 3.3.4.1 RAG fusion for systems

Previous researches adopt multi-query as an isolated retrieval techniques itself. In the proposed system, fusion techniques would be conducted in each of the system as well as the non-fusion version. In the pre-retrieval stage, 3 rewritten queries would be generated by prompting the Qwen2-7B model. The gathered results would be re-ranked by 1) reciprocal rank fusion and 2) jina-reranker-v2-base-multilingual(JRR), and the top five sorted retrieval results would be used as context for answering.

#### 3.3.4.2 Reciprocal Rank Fusion

RFF is a widely utilised rank fusion algorithm for the computation of the relevance score of hybrid search results derived from multiple queries. It is a popular and effective choice for information retrieval systems to combine search results from different retrievers or queries for more robust and reliable ranked search results. The underlying rationale is that the documents which are ranked higher in the retrieved results will be more relevant to the query. Given a set of ranked documents (denoted D) and a set of rankings (denoted R), the scores of documents could be computed by

$$RFFscore(d \in D) = \sum_{r \in R} \frac{1}{k + r(d)} [9] \tag{3.1}$$

The constant k serves to regulate the impact of documents of a lower rank, with the value increasing in accordance with this regulation. The optimal performance of this regulation is observed when k is equal to 60 through pilot investigation. [9]

#### 3.3.4.3 Jina Reranker

Unlike algorithmic fusion strategies such as RFF, the reranker JRR is fine-tuned from a transformer model and well-trained in query-documents datasets to optimise text ranking results.[22] The reranker approach is an effective method for comparing the relevance of a query and the results returned by a search engine. This approach allows for the reduction of false rankings and the elimination of irrelevant results. It is hypothesised that the reranker approach has a superior understanding of the search query

in comparison to RFF, due to its consideration of contextual meaning. Consequently, it is employed in order to conduct experiments which serve to evaluate the efficacy of the RFF approach within the aforementioned systems.

# Chapter 4

# Results and Discussion

## 4.1 Testset Collection

In order to assess the proficiency of RAG systems, a synthetic evaluation set is constructed using RAGAs. Subsequently, a distinct small test set, created by expert in quantum computing(Supervisor of this Dissertation project), is obtained for the final assessment and analysis of the selected set of systems that demonstrate superior performance in the preceding stage of evaluation.

### 4.1.1 RAGAs testset generation

The construction of a dataset for the evaluation of an RAG system is typically challenging, as the reference answer should be based on the context contained in the knowledge of the system. Furthermore, the generation of representative QA pairs for information-rich systems with cross-referencing and reasoning introduces additional complexity. In RAGAs framework, it takes the inspiration from Evol-Instruct[73] to evolve seed questions to the desired question with different characteristics. A set of documents can then be used to systematically construct questions from contexts that require the application of reasoning and consideration of multiple contextual factors in order to be answered correctly. The generated dataset comprises 4 columns: 1) question 2) contexts 3) answer 4) ground_truth (the facts selected from the datasets). [60]

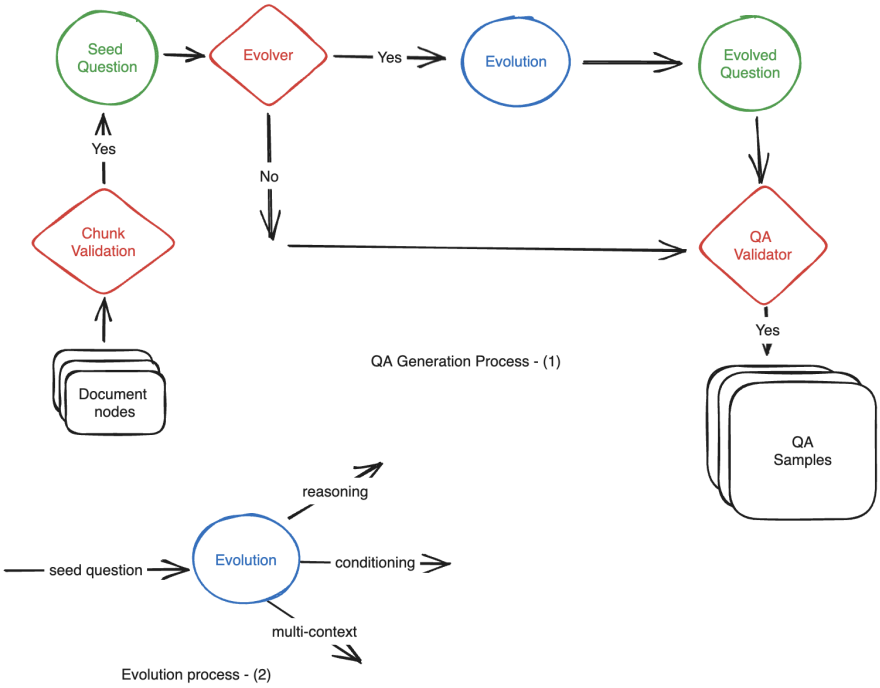


Figure 4.1: Process of synthetic dataset generation[60]

## 4.2 Metrics

In the case of human evaluation, a rating on a scale of 0 to 10 is employed as a metric. With regard to automated evaluation, the quality of the retrieved context and the generated response will be evaluated through LLM, utilising the RAGAs framework. The following metrics will be employed and assessed by LLM throughout the evaluation process.

### 4.2.1 Faithfulness [59]

Faithfulness is defined as the logical consistency of the generated response in relation to the provided context. A fully faithful answer is one that can be inferred from the given context alone; this ensures that the LLM does not use any external knowledge for the generation of the response. The score is calculated as a percentage of claims in the generated answer that can be inferred from the given context.

### 4.2.2 Answer Relevancy [56]

Answer Relevancy is a measure of the relevance between the generated answer and the query, calculated using the cosine similarity of an embedding model. Initially, a number of question variants are generated from the answer, and then the mean similarity between those variants and the original question is calculated. This provides an approximate calculation of relevance, although the quality of the result depends on the capability of the LLM and the embedding model.

### 4.2.3 Context Precision [57]

Context Precision rates the retrieved contexts based on the ranking of the ground truth-related chunks. The score is elevated when the more pertinent items are positioned higher. If we define $v_k \in \{0,1\}$ as the relevance indicator, and $K$ as the total number of items retrieved in the context, then the context precision can be calculated as follows: [57]:

$$Precision@k = \frac{true\ positives@k}{true\ positive@k + false\ positives@k} \tag{4.1}$$

$$Context\ Precision@K = \frac{\sum_{k=1}^{K}(Precision@k \times v_k)}{R} \tag{4.2}$$

### 4.2.4 Context Recall [58]

Context Recall gauges the proportion of claims substantiated by ground truth that can be inferred from retrieved contexts. Initially, the ground truth is segmented into sentences, and each sentence is then evaluated for alignment with the contexts by LLM, using a prompt that contains the question, the contexts and the segmented truth.

### 4.2.5 Answer Correctness [55]

Answer Correctness evaluates the semantic and factual similarity between the generated answer and the ground truth. The semantic similarity is determined by cosine similarity from the embedding model, while the factual similarity is computed using the F1 score. In the context of factual similarity, the precision (P) represents the proportion of factual elements present in the generated answer that align with the ground truth. The recall (R), on the other hand, denotes the portion of factual elements in the ground truth that are supported by the generated answer. The determination of both is contingent upon

prompting LLM. By default, the weighting of factual and semantic similarity is set at 0.75 and 0.25, respectively, which yields the following equation:

$$Score = 0.75 \times \frac{2PR}{P+R} + 0.25 \times semantic\,similarity \tag{4.3}$$

## 4.3 Results and Discussion

In order to guarantee impartiality with regard to the generation process, a distinct 4-bit quantised LLM Gemma27B[69], is employed for the purpose of answer generation within the RAG system. With respect to the generation of datasets and the assessment of associated metrics, GPT-4o mini[47] is utilised, as it is regarded as the most proficient model in a multitude of language-related tasks.

### 4.3.1 RAGAs generated dataset

In total, 100 sets of testing data are generated from 10% of randomly selected documents. The questions are of type simple, reasoning and multi-context in the ratio of 2:1:1. In the generated dataset, 16 of the tuples get ”The answer to given question is not present in context”, which means that the randomly selected chunk of context could not produce a meaning question that is related to it. In the dataset, all of them are references in the paper. The illegal questions are pruned out, resulting in 41 ”simple”, 20 ”reasoning” and 23 ”multi-context” questions selected for analysis.

In the first experiment, it utilises the all retrieved contexts to answer the question even though that is unrelated. So a little adjustment was made in the default prompt from llamaindex [64] for asking response from LLM. The modified prompt is as follows:

```
Context information is below.
---------------------
{context_str}
---------------------
Given the context information and not prior knowledge,
identify the related context to
answer the query.\nYou do not need to use every piece of
information provided.
Query: {query_str}
Answer:
```

#### 4.3.1.1 Evaluation Result

| System | Task | Reranker | F | AR | CP | CR |
|---|---|---|---|---|---|---|
| SV | S | None | 0.908 | 0.837 | 0.909 | 0.717 |
| SV* | S | RFF | 0.888 | 0.776 | 0.87 | 0.707 |
| SV* | S | JRR | **0.914** | **0.846** | **0.938** | **0.791** |
| AM | S | None | **0.888** | 0.829 | 0.906 | **0.799** |
| AM* | S | RFF | 0.887 | 0.803 | 0.811 | 0.579 |
| AM* | S | JRR | 0.874 | **0.847** | **0.939** | 0.722 |
| KV* | S | RFF | **0.889** | 0.837 | 0.803 | 0.632 |
| KV* | S | JRR | 0.859 | **0.875** | **0.908** | **0.709** |

Table 4.1: Averaged metric scores of system in simple task, * denotes that the system applies multi-query and fusion approach, JRR represents jina-reranker-v2-base-multilingual

| System | Task | Reranker | F | AR | CP | CR |
|---|---|---|---|---|---|---|
| SV | R | None | 0.867 | **0.808** | **0.85** | **0.75** |
| SV* | R | RFF | **0.892** | 0.736 | **0.85** | 0.696 |
| SV* | R | JRR | 0.862 | 0.768 | 0.84 | 0.738 |
| AM | R | None | 0.85 | 0.715 | **0.841** | 0.675 |
| AM* | R | RFF | **0.901** | 0.717 | **0.841** | 0.695 |
| AM* | R | JRR | 0.859 | **0.788** | **0.841** | **0.718** |
| KV* | R | RFF | **0.9** | 0.682 | 0.82 | 0.72 |
| KV* | R | JRR | 0.862 | **0.768** | **0.84** | **0.738** |

Table 4.2: Averaged metric scores of system in reasoning task, * denotes that the system applies multi-query and fusion approach, JRR represents jina-reranker-v2-base-multilingual

For simple and reasoning task, it is challenging to identify a system that consistently demonstrates superior performance. Nevertheless, it can be observed that the context precision and answer relevance exceed 80% across all systems. This demonstrates that, in the context of these information retrieval task, the performance of the task is not significantly influenced by the choice of system, given that the searching is a similarity search that is highly dependent on the embeddings model without complicated

processing for the original query. The discrepancies are primarily attributable to variations in chunking size and the inclusion of supplementary materials in the retrieved data set.

| **System** | **Task** | **Reranker** | **F** | **AR** | **CP** | **CR** |
|---|---|---|---|---|---|---|
| SV | MC | None | **0.917** | 0.783 | **0.989** | 0.804 |
| SV* | MC | RFF | 0.677 | 0.759 | 0.875 | 0.656 |
| SV* | MC | JRR | 0.788 | **0.851** | 0.955 | **0.808** |
| AM | MC | None | 0.815 | **0.812** | 0.939 | 0.692 |
| AM* | MC | RFF | 0.705 | 0.754 | 0.888 | 0.653 |
| AM* | MC | JRR | **0.912** | 0.767 | **0.95** | **0.75** |
| KV* | MC | RFF | 0.677 | 0.759 | 0.875 | 0.656 |
| KV* | MC | JRR | **0.878** | **0.772** | **0.948** | **0.775** |

Table 4.3: Averaged metric scores of system in multi-context task, * denotes that the system applies multi-query and fusion approach, JRR represents jina-reranker-v2-base-multilingual

In the multi-context task, the SV system demonstrated optimal performance, attaining high scores in the following categories: faithfulness, answer relevancy, context precision and context recall. It demonstrated the highest rank in the majority of these categories across the variations. The indexed chunks of the AM system are too fine-grained for the multi-context task, which results in its inability to achieve matching performance to that of the SV system for the same type of variations. It is evident that the Fusion KV system with JRR reranker attains the highest score in answer relevancy. The fusion of the knowledge graph with the AM system has been demonstrated to enhance the retrieval and quality of the answers produced, given that the lower-quality index from the AM system is used. This illustrates the potential of incorporating a knowledge graph into an RAG system, which could increase the capability of questions that rely on multiple contexts for answering.

| System | Task | Reranker | F | AR | CP | CR |
|---|---|---|---|---|---|---|
| SV | ALL | None | **0.897** | 0.809 | **0.916** | 0.757 |
| SV* | ALL | RFF | 0.819 | 0.757 | 0.865 | 0.686 |
| SV* | ALL | JRR | 0.855 | **0.822** | 0.911 | **0.779** |
| AM | ALL | None | 0.851 | 0.785 | 0.895 | 0.722 |
| AM* | ALL | RFF | 0.831 | 0.758 | 0.847 | 0.642 |
| AM* | ALL | JRR | **0.882** | **0.801** | **0.910** | **0.730** |
| KV* | ALL | RFF | 0.822 | 0.759 | 0.833 | 0.669 |
| KV* | ALL | JRR | **0.866** | **0.805** | **0.899** | **0.741** |

Table 4.4: Averaged metric scores of system in all tasks which is averaged by the total type of task, * denotes that the system applies multi-query and fusion approach, JRR represents jina-reranker-v2-base-multilingual

Overall, various versions of the SV system demonstrate the greatest efficacy in comprehensive tasks, and the Fusion SV system with JRR reranker exhibiting the highest level of answer relevancy. In general, the Fusion version of systems with JRR reranker exhibits slight improvements in answer relevancy and context recall, with averaged gains of 1.8% and 2%, respectively. These findings align with those of other studies on reranker performance. Moreover, the RFF reranking algorithm is not an appropriate means of reranking the retrieved context in the RAG retrieval pipeline. The metric scores are notably inferior to those of the non-fusion system. The RFF algorithm fails to consider the context and places undue reliance on the quality of the system, which is capable of providing accurate score estimation for every document in response to a given query. The pre-trained embedding model is constrained by its inability to compare contexts with queries based on semantic relevancy. Consequently, repeating parts or even the entirety of a query will result in an artificially high score. In contrast, transformer-based rerankers are trained to rank question-answer pairs, making them well-suited for post-processing retrieved documents from the RAG system.

In the alternative, an examination of tables 4.1-4.4 suggests that the fidelity of systems in separated tasks is randomly distributed, exhibiting no discernible trend or tendency to track. If we assume that the critic LLM (GPT4o-mini) is sufficiently capable of determining faithful claims in the generated answer according to context, one possible explanation for this phenomenon is the ”hallucination” problem for generator LLM (Gemma-27B). Given that it is not a particularly large model, it may be more

prone to creating unfaithful content. However, there are numerous potential causes for LLM hallucination, and further investigation is required to identify the underlying factors.

Given that the fusion version of systems typically performs better, we have elected to utilise this approach for the resulting systems. The answer correctness will be measured during the final testing phase, as this approach incorporates the considerations of ground truth.

| **System** | **Task** | **Reranker** | **Score** |
|---|---|---|---|
| SV* | User question | JRR | **0.644** |
| AM* | User question | JRR | 0.618 |
| KV* | User question | JRR | 0.621 |

Table 4.5: Averaged Answer Correctness of system in all tasks which is averaged by the total type of task, * denotes that the system applies multi-query and fusion approach, JRR represents jina-reranker-v2-base-multilingual

The ordering and performance are consistent with the answer relevancy and preceding points that have been made.

### 4.3.2 Human generated dataset

The rated score of 10 questions by expert for the final selected systems are as follows:

| **System** | **Task** | **Reranker** | **Mean** | **Median** | **Std** |
|---|---|---|---|---|---|
| SV* | User Question | JRR | 4.8 | 5 | 2.75 |
| AM* | User Question | JRR | 4.9 | 5 | 1.87 |
| KV* | User Question | JRR | **5.3** | 5 | 1.85 |

Table 4.6: Statistics of user rating of system in the human-generated dataset, which simulates the actual question of user.

Conversely, in this small user study, KV system is in favour of user with lowest fluctuation in score. It adds evidence to the significance to the use of knowledge graph for improving RAG system. Given the limited resources, it may not have enough information to answer every questions that user gave. One example as the following:

**Question: Which quantum machine learning techniques deal best with noise?**

| System | Answer | Score |
|---|---|---|
| SV* + JRR | Q-LEAR # | 5 |
| AM* + JRR | Q-LEAR # | 5 |
| KV* + JRR | Q-LEAR # | 5 |

Table 4.7: # denotes the text before it represents the central idea

#### 4.3.2.1 Case study 1

**In what situations can distributed quantum computing be applied advantageously.**

| System | Answer | Score |
|---|---|---|
| SV* + JRR | Only explaining contexts and questions # | 2 |
| AM* + JRR | Only explaining contexts and questions # | 2 |
| KV* + JRR | Smart grid management<br>IoT cooperation<br>UAV trajectory planning # | 6 |

Table 4.8: # denotes the text before it represents the central idea, four out of ten questions are rated 8 or higher among the systems

In the presented case, the knowledge graph of the KV system successfully retrieved some applications that could potentially benefit from parallel quantum computing. The knowledge graph could indeed assist in matching some terminologies for the retrieval system. However, due to the limited size of the dataset and the capacity of the LLM, which helps to synthesise knowledge to build the KG, the answer does not consist of explanations of these applications.

#### 4.3.2.2 Case study 2

**What quantum programs are enabled by coherent quantum control that you can't write without it?**

| System | Answer | Score |
|---|---|---|
| SV* + JRR | The text states that coherent control allows for the implementation of universal quantum computation by supplementing classical operations with quantum gates acting on a control system. It implies that without coherent control, achieving universal quantum computation solely through classical operations would be impossible. | 9 |
| AM* + JRR | Explaining Coherent Parallelization(CP). # | 5 |
| KV* + JRR | Explaining Coherent Parallelization(CP). # | 5 |

Table 4.9: # denotes the text before it represents the central idea

In Case 2, the SV* + JRR system provides a satisfactory response to the user, whereas the other two systems retrieved irrelevant content and proceeded to answer with it. This demonstrates that the chunking size of documents impacts the retrieval process for different queries. To develop a more comprehensive and effective system, a collection of diverse RAG systems with variations in chunking methodology and retrieval pipelines should be constructed. In such a case, generated answers could be evaluated and the most prominent one could be selected according to the context of the user query.

In addition, in this example, the retrieved contexts and answers are identical for AM* + JRR and KV* + JRR. This suggests that the knowledge graph constructed does not yield any more pertinent content than the auto-merge retriever. In certain instances, the graph quality is insufficient. In production-ready applications, it requires a superior and more promising LLM for synthesising the knowledge in academic papers.

# Chapter 5

# Conclusions

## 5.1 Summary

In conclusion, this project has successfully developed three functional RAG systems for synthesising quantum computing knowledge from academic papers. Furthermore, the systems have been shown to provide relevant answers to user queries posed by LLM. The results of the experiments demonstrated that the knowledge graph has the potential to enhance the retrieval performance of the existing retrieval system. Additionally, the evaluated data indicates that reciprocal rank fusion is less effective than transformer-based rerankers, even in the absence of a reranker. This is because it relies on the quality of the retrieval system, without considering the contexts of the retrieved documents. The evaluation results demonstrate that the KV* + JRR system exhibits the highest performance in terms of answer relevancy and correctness, as determined by the query, context, and ground truth. In the user study, the systems were able to provide answers that were rated 8 or higher for 40% of the questions. The KV* + JRR system demonstrated overall superiority in this study, as well as an ability to synthesise terminology. These findings illustrate the practical utility of the system and techniques developed to assist both novice and experienced researchers in acquiring quantum computing knowledge or in constructing an RAG system for academic papers.

In summary, this study addresses the research gap between the latest techniques of RAG systems and implements a practical system to assist users in investigating the field of quantum computing, thereby meeting the original objectives.

## 5.2 Limitations

Given the constraints on cost and resources, the construction of an external knowledge system of quantum computing, as well as the generation of evaluation datasets by both LLM and expert, was limited in scope. Due to time and cost limitations, only one knowledge graph was generated with a small LLM model (Qwen2-7B) and only a heuristic PDF parsing package was used to parse the quantum computing papers. Further investigations are required to optimise the generation process and procedures in order to obtain the best quality knowledge graph for retrieval from the well-formulated content parsed by vision models.

## 5.3 Future work

In order to enhance the project, it is necessary to address a number of areas for improvement in the future:

1. accumulate a considerably more substantial corpus of quantum computing literature, meticulously annotated in Markdown format using the marker-pdf tool to facilitate enhanced presentation of tables and equations.

2. Investigate methodologies for the encoding of images, tables and equations from paper documents and the subsequent indexing of these for the purpose of facilitating searches.

3. Conduct experiments in order to ascertain the optimal chunking size and construction pipeline for a knowledge graph, utilising a commercial large language model.

4. Undertake a comprehensive analysis of a substantial number of users and expectations, along with their opinions on the retrieval and answer quality of the systems in question.

5. Integrate performing systems in order to create a comprehensive collection of expectations, with the aim of enhancing the system's robustness and retrieval quality.

# Bibliography


[1] Felipe Almeida and Geraldo Xexéo. Word embeddings: A survey, 2023.

[2] Negar Arabzadeh, Xinyi Yan, and Charles L. A. Clarke. Predicting efficiency/effectiveness trade-offs for dense vs. sparse retrieval strategy selection. In *Proceedings of the 30th ACM International Conference on Information & Knowledge Management*, CIKM '21, page 2862–2866, New York, NY, USA, 2021. Association for Computing Machinery.

[3] arXiv. arxiv: An open access archive for scholarly articles, 2024. Accessed: 2024-08-17.

[4] Andreas Bayerstadler, Guillaume Becquin, Julia Binder, Thierry Botter, Hans Ehm, Thomas Ehmer, Marvin Erdmann, Norbert Gaus, Philipp Harbach, Maximilian Hess, Johannes Klepsch, Martin Leib, Sebastian Luber, Andre Luckow, Maximilian Mansky, Wolfgang Mauerer, Florian Neukart, Christoph Niedermeier, Lilly Palackal, Ruben Pfeiffer, Carsten Polenz, Johanna Sepulveda, Tammo Sievers, Brian Standen, Michael Streif, Thomas Strohm, Clemens Utschig-Utschig, Daniel Volz, Horst Weiss, Fabian Winter, Quantum Technology, and Application Consortium QUTAC. Industry quantum computing applications. *EPJ Quantum Technology*, 8(1):25, Nov 2021.

[5] Paul Benioff. The computer as a physical system: A microscopic quantum mechanical hamiltonian model of computers as represented by turing machines. *Journal of Statistical Physics*, 22:563–591, 05 1980.

[6] Lukas Blecher, Guillem Cucurull, Thomas Scialom, and Robert Stojnic. Nougat: Neural optical understanding for academic documents. In *The Twelfth International Conference on Learning Representations*, 2024.

[7] Wei-Lin Chiang, Lianmin Zheng, Ying Sheng, Anastasios Nikolas Angelopoulos, Tianle Li, Dacheng Li, Banghua Zhu, Hao Zhang, Michael Jordan, Joseph E.

Gonzalez, and Ion Stoica. Chatbot arena: An open platform for evaluating LLMs by human preference. In *Forty-first International Conference on Machine Learning*, 2024.

[8] Eunsol Choi, He He, Mohit Iyyer, Mark Yatskar, Wen-tau Yih, Yejin Choi, Percy Liang, and Luke Zettlemoyer. QuAC: Question answering in context. In Ellen Riloff, David Chiang, Julia Hockenmaier, and Jun'ichi Tsujii, editors, *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 2174–2184, Brussels, Belgium, October-November 2018. Association for Computational Linguistics.

[9] Gordon V. Cormack, Charles L A Clarke, and Stefan Buettcher. Reciprocal rank fusion outperforms condorcet and individual rank learning methods. In *Proceedings of the 32nd International ACM SIGIR Conference on Research and Development in Information Retrieval*, SIGIR '09, page 758–759, New York, NY, USA, 2009. Association for Computing Machinery.

[10] David Deutsch. Quantum theory, the church–turing principle and the universal quantum computer. *Proceedings of the Royal Society of London. Series A, Mathematical and Physical Sciences*, 400:97–117, 1985.

[11] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 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, Volume 1 (Long and Short Papers)*, pages 4171–4186, 2019.

[12] Matouš Eibich, Shivay Nagpal, and Alexander Fred-Ojala. Aragog: Advanced rag output grading, 2024.

[13] Elsevier. Quantum computing research trends report, 2021. Accessed: 2024-08-15.

[14] Shahul Es, Jithin James, Luis Espinosa Anke, and Steven Schockaert. RAGAs: Automated evaluation of retrieval augmented generation. In Nikolaos Aletras and Orphee De Clercq, editors, *Proceedings of the 18th Conference of the European Chapter of the Association for Computational Linguistics: System Demonstrations*, pages 150–158, St. Julians, Malta, March 2024. Association for Computational Linguistics.

[15] ExplodingGradients. Synthetic test data generation, 2024. Accessed: 2024-08-16.

[16] Richard P. Feynman. Simulating physics with computers. *International Journal of Theoretical Physics*, 21(6):467–488, Jun 1982.

[17] Paulo Finardi, Leonardo Avila, Rodrigo Castaldoni, Pedro Gengo, Celio Larcher, Marcos Piau, Pablo Costa, and Vinicius Caridá. The chronicles of rag: The retriever, the chunk and the generator, 2024.

[18] Yunfan Gao, Yun Xiong, Xinyu Gao, Kangxiang Jia, Jinliu Pan, Yuxi Bi, Yi Dai, Jiawei Sun, and Haofen Wang. Retrieval-augmented generation for large language models: A survey. *arXiv preprint arXiv:2312.10997*, 2023.

[19] Xiaoxin He, Yijun Tian, Yifei Sun, Nitesh V. Chawla, Thomas Laurent, Yann LeCun, Xavier Bresson, and Bryan Hooi. G-retriever: Retrieval-augmented generation for textual graph understanding and question answering, 2024.

[20] Ziwei Ji, Nayeon Lee, Rita Frieske, Tiezheng Yu, Dan Su, Yan Xu, Etsuko Ishii, Ye Jin Bang, Andrea Madotto, and Pascale Fung. Survey of hallucination in natural language generation. *ACM Comput. Surv.*, 55(12), mar 2023.

[21] Huiqiang Jiang, Qianhui Wu, Chin-Yew Lin, Yuqing Yang, and Lili Qiu. LLMLingua: Compressing prompts for accelerated inference of large language models. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pages 13358–13376. Association for Computational Linguistics, December 2023.

[22] Jina AI. Jina reranker v2 base multilingual, 2024. Accessed: 2024-08-17.

[23] Jina AI. Reranker api, 2024. Accessed: 2024-08-16.

[24] K Sparck Jones, Steve Walker, and Stephen E. Robertson. A probabilistic model of information retrieval: development and comparative experiments: Part 2. *Information processing & management*, 36(6):809–840, 2000.

[25] Nikhil Kandpal, Haikang Deng, Adam Roberts, Eric Wallace, and Colin Raffel. Large language models struggle to learn long-tail knowledge. In Andreas Krause, Emma Brunskill, Kyunghyun Cho, Barbara Engelhardt, Sivan Sabato, and Jonathan Scarlett, editors, *Proceedings of the 40th International Conference on Machine Learning*, volume 202 of *Proceedings of Machine Learning Research*, pages 15696–15707. PMLR, 23–29 Jul 2023.

[26] Minki Kang, Jin Myung Kwak, Jinheon Baek, and Sung Ju Hwang. Knowledge graph-augmented language models for knowledge-grounded dialogue generation, 2023.

[27] Vladimir Karpukhin, Barlas Oğuz, Sewon Min, Patrick Lewis, Ledell Wu, Sergey Edunov, Danqi Chen, and Wen Tau Yih. Dense passage retrieval for open-domain question answering. In *EMNLP 2020 - 2020 Conference on Empirical Methods in Natural Language Processing, Proceedings of the Conference*, EMNLP 2020 - 2020 Conference on Empirical Methods in Natural Language Processing, Proceedings of the Conference, pages 6769–6781. Association for Computational Linguistics (ACL), 2020. Publisher Copyright: © 2020 Association for Computational Linguistics; 2020 Conference on Empirical Methods in Natural Language Processing, EMNLP 2020 ; Conference date: 16-11-2020 Through 20-11-2020.

[28] J. Kietzmann, D. S. Demetis, T. Eriksson, and A. Dabirian. Hello quantum! how quantum computing will change the world. *IT Professional*, 23(04):106–111, jul 2021.

[29] 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. Retrieval-augmented generation for knowledge-intensive nlp tasks. In *Proceedings of the 34th International Conference on Neural Information Processing Systems*, NIPS '20, Red Hook, NY, USA, 2020. Curran Associates Inc.

[30] Shu-Shen Li, Gui Long, Feng-Shan Bai, Song-Lin Feng, and Hou-Zhi Zheng. Quantum computing. *Proceedings of the National Academy of Sciences*, 98:11847–11848, 10 2001.

[31] Zehan Li, Xin Zhang, Yanzhao Zhang, Dingkun Long, Pengjun Xie, and Meishan Zhang. Towards general text embeddings with multi-stage contrastive learning, 2023.

[32] Nelson F. Liu, Kevin Lin, John Hewitt, Ashwin Paranjape, Michele Bevilacqua, Fabio Petroni, and Percy Liang. Lost in the Middle: How Language Models Use Long Contexts. *Transactions of the Association for Computational Linguistics*, 12:157–173, 02 2024.

[33] LlamaIndex. Auto merging retriever, 2024. Accessed: 2024-08-17.

[34] LlamaIndex. Hierarchical, 2024. Accessed: 2024-08-17.

[35] LlamaIndex. Llamaindex: Data framework for llm applications, 2024. Accessed: 2024-08-16.

[36] LlamaIndex. Llamaparse: Genai-native document parsing platform, 2024. Accessed: 2024-08-17.

[37] LlamaIndex. Markdown, 2024. Accessed: 2024-08-17.

[38] LlamaIndex. Property graph, 2024. Accessed: 2024-08-17.

[39] LlamaIndex. Sentence splitter, 2024. Accessed: 2024-08-17.

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

[41] MongoDB, Inc. Mongodb atlas vector search, 2024. Accessed: 2024-08-16.

[42] Rajiv Movva, Sidhika Balachandar, Kenny Peng, Gabriel Agostini, Nikhil Garg, and Emma Pierson. Topics, authors, and institutions in large language model research: Trends from 17K arXiv papers. In Kevin Duh, Helena Gomez, and Steven Bethard, editors, *Proceedings of the 2024 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (Volume 1: Long Papers)*, pages 1223–1243, Mexico City, Mexico, June 2024. Association for Computational Linguistics.

[43] Niklas Muennighoff, Nouamane Tazi, Loic Magne, and Nils Reimers. MTEB: Massive text embedding benchmark. In Andreas Vlachos and Isabelle Augenstein, editors, *Proceedings of the 17th Conference of the European Chapter of the Association for Computational Linguistics*, pages 2014–2037, Dubrovnik, Croatia, May 2023. Association for Computational Linguistics.

[44] Neo4j Engineering. Neo4j vector index and search, 2024. Accessed: 2024-08-16.

[45] Ka Chun Ng. Informatics project proposal 2024 - learning quantum computing, 2024. Unpublished work.

[46] nlmatics. nlm-ingestor, 2024. Accessed: 2024-08-17.

[47] OpenAI. Gpt-4o mini: advancing cost-efficient intelligence, 2024. Accessed: 2024-08-17.

[48] Oded Ovadia, Menachem Brief, Moshik Mishaeli, and Oren Elisha. Fine-tuning or retrieval? comparing knowledge injection in llms, 2024.

[49] Mandeep Pannu, Anne James, and Robert Bird. A comparison of information retrieval models. In *Proceedings of the Western Canadian Conference on Computing Education*, WCCCE '14, New York, NY, USA, 2014. Association for Computing Machinery.

[50] Wenjun Peng, Guiyang Li, Yue Jiang, Zilong Wang, Dan Ou, Xiaoyi Zeng, Derong Xu, Tong Xu, and Enhong Chen. Large language model based long-tail query rewriting in taobao search. In *Companion Proceedings of the ACM on Web Conference 2024*, WWW '24, page 20–28, New York, NY, USA, 2024. Association for Computing Machinery.

[51] Tyler Procko. Graph retrieval-augmented generation for large language models: A survey. *Available at SSRN*, 2024.

[52] py-pdf. Benchmarks for pdf libraries, 2024. Accessed: 2024-08-17.

[53] pymupdf. Rag, 2024. Accessed: 2024-08-17.

[54] Qwen. Qwen/qwen2-7b, 2024. Accessed: 2024-08-17.

[55] Ragas. Answer correctness, 2024. Accessed: 2024-08-17.

[56] Ragas. Answer relevance, 2024. Accessed: 2024-08-17.

[57] Ragas. Context precision, 2024. Accessed: 2024-08-17.

[58] Ragas. Context recall, 2024. Accessed: 2024-08-17.

[59] Ragas. Faithfulness, 2024. Accessed: 2024-08-17.

[60] Ragas. Ragas documentation, 2024. Accessed: 2024-08-17.

[61] Pranav Rajpurkar, Jian Zhang, Konstantin Lopyrev, and Percy Liang. SQuAD: 100,000+ questions for machine comprehension of text. In Jian Su, Kevin Duh,

and Xavier Carreras, editors, *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pages 2383–2392, Austin, Texas, November 2016. Association for Computational Linguistics.

[62] Adrian H. Raudaschl. Forget rag, the future is rag-fusion, 2023. Accessed: 2024-08-15.

[63] S.E. ROBERTSON. The probability ranking principle in ir. *Journal of Documentation*, 33(4):294–304, Jan 1977.

[64] run-llama. default_prompts.py, 2024. Accessed: 2024-08-17.

[65] Maria Schuld and Nathan Killoran. Quantum machine learning in feature hilbert spaces. *Phys. Rev. Lett.*, 122:040504, Feb 2019.

[66] Shivansh Kaushik. Advanced text retrieval with elasticsearch & llamaindex: Sentence window retrieval, 2024. Accessed: 2024-08-16.

[67] Amit Singhal et al. Modern information retrieval: A brief overview. *IEEE Data Eng. Bull.*, 24(4):35–43, 2001.

[68] Superlinked. Evaluation of rag retrieval chunking methods, 2024. Accessed: 2024-08-16.

[69] Gemma Team. Gemma, 2024.

[70] Vik Paruchuri. Marker, 2024. Accessed: 2024-08-17.

[71] Jianguo Wang, Xiaomeng Yi, Rentong Guo, Hai Jin, Peng Xu, Shengjun Li, Xiangyu Wang, Xiangzhou Guo, Chengming Li, Xiaohai Xu, Kun Yu, Yuxing Yuan, Yinghao Zou, Jiquan Long, Yudong Cai, Zhenxiang Li, Zhifeng Zhang, Yihua Mo, Jun Gu, Ruiyi Jiang, Yi Wei, and Charles Xie. Milvus: A purpose-built vector data management system. In *Proceedings of the 2021 International Conference on Management of Data*, SIGMOD '21, page 2614–2627, New York, NY, USA, 2021. Association for Computing Machinery.

[72] Yuqi Wang, Boran Jiang, Yi Luo, Dawei He, Peng Cheng, and Liangcai Gao. Reasoning on efficient knowledge paths:knowledge graph guides large language model for domain question answering, 2024.

[73] Can Xu, Qingfeng Sun, Kai Zheng, Xiubo Geng, Pu Zhao, Jiazhan Feng, Chongyang Tao, and Daxin Jiang. Wizardlm: Empowering large language models to follow complex instructions, 2023.

[74] Liang Xu, Lu Lu, and Minglu Liu. Construction and application of a knowledge graph-based question answering system for nanjing yunjin digital resources. *Heritage Science*, 11(1):222, Oct 2023.

[75] Liang Xu, Lu Lu, Minglu Liu, Chengxuan Song, and Lizhen Wu. Nanjing yunjin intelligent question-answering system based on knowledge graphs and retrieval augmented generation technology. *Heritage Science*, 12(1):118, Apr 2024.

[76] Zhentao Xu, Mark Jerome Cruz, Matthew Guevara, Tie Wang, Manasi Deshpande, Xiaofeng Wang, and Zheng Li. Retrieval-augmented generation with knowledge graphs for customer service question answering. In *Proceedings of the 47th International ACM SIGIR Conference on Research and Development in Information Retrieval*, SIGIR '24, page 2905–2909, New York, NY, USA, 2024. Association for Computing Machinery.

[77] Jihong Yan, Chengyu Wang, Wenliang Cheng, Ming Gao, and Aoying Zhou. A retrospective of knowledge graphs. *Frontiers of Computer Science*, 12(1):55–74, Feb 2018.

[78] Zhilin Yang, Peng Qi, Saizheng Zhang, Yoshua Bengio, William Cohen, Ruslan Salakhutdinov, and Christopher D. Manning. HotpotQA: A dataset for diverse, explainable multi-hop question answering. In Ellen Riloff, David Chiang, Julia Hockenmaier, and Jun'ichi Tsujii, editors, *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 2369–2380, Brussels, Belgium, October-November 2018. Association for Computational Linguistics.

[79] Xin Zhang, Yanzhao Zhang, Dingkun Long, Wen Xie, Ziqi Dai, Jialong Tang, Huan Lin, Baosong Yang, Pengjun Xie, Fei Huang, Meishan Zhang, Wenjie Li, and Min Zhang. mgte: Generalized long-context text representation and reranking models for multilingual text retrieval, 2024.