QuARK: LLM-Based Domain-Specific Question Answering using Retrieval Augmented Generation and Knowledge Graphs

Edward Burgin, Sourav Dutta and **Mingxue Wang**
Huawei Ireland Research Centre
Dublin, Ireland
edwardburgin@huawei-partners.com
{sourav.dutta2, wangmingxue1}@huawei.com

Abstract

Retrieval Augmented Generation (RAG) has been pivotal in the utilization of Large Language Models (LLM) to improve the factuality of long-form question answering systems in industrial settings. Knowledge graphs (KG) represent a linking of disparate information sources that potentially yield useful information for mitigating the issues of insufficient knowledge and hallucination within the LLM-RAG pipeline. However, the creation of *domain-specific* KG is costly and usually requires a domain expert. To alleviate the above challenges, this work proposes QuARK, a novel domain-specific question answering framework to enhance the knowledge capabilities of LLM by integrating structured KG, thereby significantly reducing the reliance on the "generic" latent knowledge of LLMs. Here, we showcase how LLMs can be deployed to not only act in dynamic information retrieval and in answer generating frameworks, but also as flexible agents to automatically extract relevant entities and relations for the automated construction of domain-specific KGs. Crucially we propose how the pairing of question decomposition and semantic triplet retrieval within RAG can enable optimal subgraph retrieval. Experimental evaluations of our framework on financial domain public dataset, demonstrate that it enables a robust pipeline incorporating schema-free KG within a RAG framework to improve the overall accuracy by nearly **13%**.

1 Introduction

Large Language Models (LLMs) have demonstrated exceptional generative capabilities and achieved significant progress in the development of versatile intelligent agents. LLMs have been shown to be able to solve complex reasoning tasks through prompt engineering and in-context learning (Dong et al., 2024) – sometimes even without fine-tuning for specific tasks. In this context, LLMs of the extended families of GPT, LLaMa, Gemini, DeepSeek, and Qwen to name a few (OpenAI et al., 2024; Yang et al., 2025; Grattafiori et al., 2024; Team et al., 2024; DeepSeek-AI et al., 2025), have been pivotal in advancing the state-of-the-art in the domains of natural language processing, image generation, multi-modal computing, and AI agents.

In spite of such over-arching success, it might be fair to say that LLMs have had mixed penetration in organizational settings. Domain-specificity of industrial information along with incomplete domain knowledge, which mismatches with the LLM pre-training open-domain datasets, and "hallucinations" wherein incorrect information is presented in a compelling manner, provide research challenges in this area (Huang et al., 2025).

To partially mitigate the above issues, recent studies explored Retrieval Augmented Generation (RAG) methodology (Lewis et al., 2020). Here, a knowledge repository (usually outside the domain of training) is sent to the LLM for optimizing its output based on the "authoritative" additional information provided. In a nutshell, RAG aims to dynamically inject specific information into LLM prompts (at inference/generation time) to reduce the possibility of hallucinations and improve integration of external up-to-date knowledge.

Knowledge Graphs (KG) provide a mechanism to represent factual knowledge in the form of subject-predicate-object (SPO) triples, capturing relations among entities or objects, i.e., (entity) - [relationship] - (entity). This allows for structured knowledge augmentation into the LLM answer generation pipeline, thereby boosting the overall performance of the framework (Pan et al., 2023).

The above approaches tend to work well, in practice, for open-domain scenarios. However, for industrial settings creation of meaningful, precise, and large domain-specific knowledge graphs poses a significant challenge. Carefully crafting knowledge graphs for each of the relevant business domains requires domain experts and is manually intensive – rendering it infeasible in most scenarios. Additionally, new entities may emerge and relationships might evolve with time, which would require appropriately updating the KG.

To this end, in this paper, we propose the novel ***Qu**estion **A**nswering using **R**etrieval Augmented Generation and **K**nowledge Graphs* (`QuARK`) framework. `QuARK` employs a suite of multiple LLMs (with different prompts) to tackle:

(i) extraction of domain-specific entities and relations in documents for automated KG construction;

(ii) breakdown of query for in-depth understanding and retrieval of relevant implicit information;

(iii) efficient identification of semantically relevant document snippets to the query for RAG; and

(iv) precise generation of final answer to the user query given the contexts and KG information.

This provides a standalone end-to-end framework for tackling domain-specific question answering. It should be noted that `QuARK` can be inherently multi-lingual based on the choice of LLMs used, without any changes to the overall pipeline. We also empirically evaluate the `QuARK` framework on a financial QA benchmark dataset and showcase significant accuracy gains, in terms of accuracy of the answers generated, compared to existing LLM approaches.

2 `QuARK` Model Architecture

In this section, we discuss the different modules and internal workings of the `QuARK` framework.

Document Pre-Processing. In real-life, organizational documents like manuals, financial records, etc., tend to be heterogeneous in terms of content and file types. For example, a typical product manual would contain unstructured texts along with diagrams, while financial documents tend to contain structured tables along with textual contents. Further, documents may be in different formats, i.e., pdfs, ppts, or simply txt.

In this context, we used the *PyMuPDFLoader* software for conversion of the financial pdf documents to text. To detect pages with tables in the corpus, `QuARK` uses *Detectron2* (Wu et al., 2019), a neural network extractor for detection of structured tabular entries within a document. Subsequently, the *Granite 3.2* vision model (Team et al., 2025) was used for extraction and representation of the tabular fields and information in HTML format. This provided a consistent textual representation of financial documents in our evaluation benchmark.

Document Embedding. Given a collection of documents, `QuARK` then creates a dense-vector representation to capture the semantic information present. For generating semantic embeddings of texts within the financial documents we used *multilingual-e5-large* (mE5) (Wang et al., 2024a) text embedding language model. Since, documents can be quite large in practice, we segment the entire documents into chunks of size 1024 tokens with a chunk overlap of 30 tokens to provide concept overlap between adjacent chunks. The embeddings of document chunks are indexed and stored using the *Chroma* vector database (obtained from `trychroma.com`), referred to as "document store".

It should be noted that, although, `QuARK` considers only text and table based information, it can be easily extended to images (present in the documents) – by detecting images and captions using vision based LLM models and storing the corresponding image embeddings.

KG Construction. An important contribution of this work is the automated construction of domain-specific Knowledge Graphs (KG) using LLM for boosting the efficacy of the QA framework. Graph creation through LLM, in particular through quantized models has become increasingly appealing due to the long context understanding, fast inference, and increasingly reliable dynamic instruction following of the latest LLM models. Thus, in order to create such a KG without a pre-defined schema and without the need of a domain specialist (for thorough understanding of domain), we utilized a prompt-based LLM approach.

In this regards, we use the quantized *Qwen2-72b-Instruct-q5_K_M* LLM (Yang et al., 2025) (5 bit K-quantization with medium configuration) for processing the entire collection of documents (in the dataset) to extract SPO triples from the chunks, thus forming the domain-specific KG. The following prompt was utilized to generate the KG:

Knowledge Graph Generation Prompt

Convert the following text into a comprehensive list of RDF triples, as a comma separated list of the following format (subject)-[predicate]-(object) in between [GRAPH] and [/GRAPH] tags. Try to perform entity resolution on text, and keep relation descriptions simple and standardized. Any information that you don't think will be useful should be discarded. We are particularly interested in parameters, variable names, databases, products and business keywords. Do not explain your decisions, just output results. Do not add any notes, as output will be used directly.

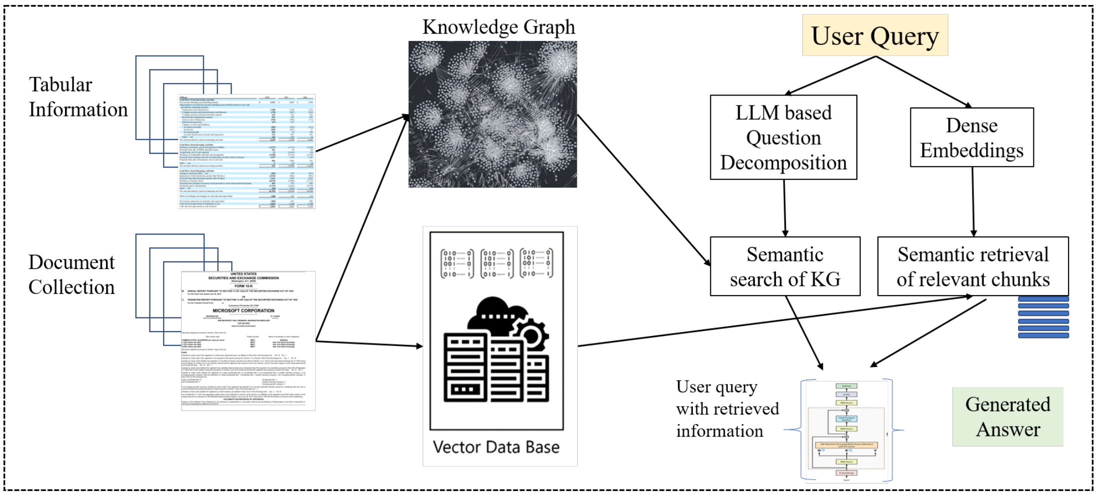

Figure 1: Depiction of the modular architectural components of `QuARK`.

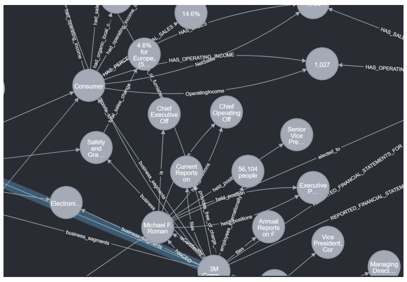

Figure 2: Example KG created from document texts by `QuARK`.

Subject	Predicate	Object
(3M COMPANY)	[STATE_OF_INCORPORATION]	(Delaware)
(3M COMPANY)	[IRS_EMPLOYER_IDENTIFICATION_NO.]	(41-0417775)
(3M COMPANY)	[PRINCIPAL_EXECUTIVE_OFFICES]	(3M Center, St. Paul, Minnesota 55144)
(3M COMPANY)	[TELEPHONE_NUMBER]	(651) 733-1110
(3M COMPANY)	[SECURITIES_REGISTERED]	(Common Stock, Par Value $.01 Per Share)
(3M COMPANY)	[SECURITIES_REGISTERED]	(1.500% Notes due 2026)
(3M COMPANY)	[SECURITIES_REGISTERED]	(Floating Rate Notes due 2020)
(3M COMPANY)	[SECURITIES_REGISTERED]	(0.375% Notes due 2022)
(3M COMPANY)	[SECURITIES_REGISTERED]	(0.950% Notes due 2023)
(3M COMPANY)	[SECURITIES_REGISTERED]	(1.750% Notes due 2030)
(3M COMPANY)	[SECURITIES_REGISTERED]	(1.500% Notes due 2031)
(3M COMPANY)	[TRADED_ON]	(SWX Swiss Exchange)

Table 1: Example of KG triplets constructed from tabular data in `QuARK` for 3M Company.

A snapshot of the corresponding KG obtained is shown in Figure 2. However, the above procedure was unable to generate well-formed triples for the tabular structures present in the documents. To alleviate the above and generate a more complete knowledge graph, we additionally used *LLaMa3.3-70B-Instruct-q5_K_M* LLM (Grattafiori et al., 2024) (5 bit K-quantization with medium configuration, generated with LLaMa-cpp) to convert the information within the tables (detected, extracted by vLLM, and pre-curated to HTML) to KG entries. Table 1 presents an example of KG triples, for an organization, extracted by `QuARK`.

Observe, that there may be overlaps or duplicates among the triples extracted between the above two KG generation processes (but can be easily removed based on exact text matching between triples). Finally, the semantic vector representations of the extracted KG triples (using the mE5 model as above) are also stored in a "KG store".

From the above examples, we observe that prompt based LLM methodology showcases good performance (both in terms of quality and comprehensiveness) in extracting domain relevant entities and relations for automated construction of the KG. As a final thought, in certain scenarios where the KG is considered to be incomplete or of lower quality, use of traditional NLP techniques and further LLM prompting techniques can be used to further refine and improve the extracted KG.

Semantic Query Understanding. Given a user query, we aim to understand not only the semantics of the question, but also subtle and implicit information requested. In this respect, on one hand we propose the use of *semantic vector representation* for the query (by using mE5 model) to capture the overall context of the user question. On the other hand, we would like to understand (and subsequently retrieve) fine-grained information required to precisely answer questions that might be inadvertently vague or required additional information. Here we use LLaMa3.3-70B-Instruct-q5_K_M to decompose the user query into individually relevant and retrievable facts. This is concomitant to the fact, that in order to answer many of the questions, different aspects of information might be required, which are not explicitly stated in the question, and therefore must be surmised from the query.

In our framework, we prompt the LLM to generate a break down of a rather abstractly posed user query into individual "bullet points" of necessary information pieces (that should be retrieved from the KG and/or document stores). As an example (from the dataset), consider the below user query, wherein the LLM lists the information nuggets required to generate the precise answer.

Example Question

Assume that you are a public equities analyst. Answer the following question by primarily using information that is shown in the balance sheet: what is the year end FY2018 net PPNE for 3M? Answer in USD billions.

Semantically Searchable Question Decomposition

● 3M's FY2018 balance sheet
● Definition of Net Property, Plant, and Equipment (Net PPNE)
● Calculation formula for Net PPNE: Gross PPNE - Accumulated Depreciation
● Location of Gross PPNE on a balance sheet
● Location of Accumulated Depreciation on a balance sheet
● 3M's FY2018 Gross PPNE value in USD billions
● 3M's FY2018 Accumulated Depreciation value in USD billions

This novelty in `QuARK` provides a comprehensive understanding of both semantics and concepts that diverse and complex user question entails, and enables higher efficiency of our QA system. *Question decomposition* is thus critical for effective information recall, and we use the following prompt:

Question Decomposition Prompt

Create a list of bullet points of the information that you would like to use to query a knowledge base in order to answer a question that will follow, only responding with a list of bullet points. Each individual bullet point should be holistic, so the bullet point can be used in isolation from the others. [QUESTION] · · · [/QUESTION]

Retrieval Augmentation. Given the dense representation of the query and the document chunks (using mE5 language model), we select the top-4 chunks (each comprising 1024 tokens) from the "document store" (i.e., Chroma vector DB) based on their *cosine similarity* score to the query.

It should be noted that "decomposition" of different user queries (as shown above) would have differing number of relevant information requirements (i.e., number of bullet-points generated by the LLM). To extract the most relevant triples to a user query, we compute the cosine similarity between the KG triple embeddings and the embeddings of the bullet point representation of the query along with the question itself. We apply a threshold (set at 0.7) to the semantic similarity and return those above threshold for each of the query bullet-points generated. Pairing textual retrieval with semantic graph search, thus provides an effective RAG method for retrieving individual as well as consolidated facts required to answer a question.

Answer Generation. Finally, the original user query along with the document chunks extracted and the relevant KG triples found (during the RAG process) are sent to an LLM for answer generation. In our framework we use the 4-bit K-quantized *DeepSeek-R1-70B-LLaMa3-Q4_K_M* LLM (DeepSeek-AI et al., 2025) (henceforth referred to as DeepSeek in this paper). We prompt the DeepSeek model to generate answers to user queries, based on the retrieved document chunk and KG triples presented to it as primary information sources, using the following prompt:

LLM QA prompt with KG and Document Chunks

System Prompt Use the following pieces of context to answer the user's question. If you don't know the answer, just say that you don't know, don't try to make up an answer. The following knowledge graph subset could be useful to answer the user question: [KG]· · ·[/KG] And here is some relevant documents: [CONTEXT]· · ·[/CONTEXT] *Human Prompt* [QUESTION]· · ·[/QUESTION]

An overview of the `QuARK` architecture is presented in Figure 1. The use of quantized general-purpose LLMs (without expensive fine-tuning) in `QuARK` enables it to be not only efficient across domains but also light-weight and standalone, for applicability in varied production environments.

3 Domain-Specific Dataset

In this work, we consider the **FinanceBench** dataset (Islam et al., 2023) from the financial domain. The dataset includes documents covering 40 companies that are publicly traded in the USA with 361 public filings released between 2015 and 2023, with each document comprising a mix of textual and tabular information. Such mix of textual and tabular information in documents and metric questions are also common in other domains like AIOps. The FinanceBench dataset contains 150 questions (with manually checked answers by experts) catering to three different scenarios – (i) 50 domain relevant questions, (ii) 50 novel questions generated by annotators, and (iii) 50 "metric" questions based on financial analysis and computations.

This ensures a well-rounded evaluation for different QA methodologies (and LLM performance), by providing the following challenges – (i) understanding of specific terminology (of the financial domain), e.g., securities trading; (ii) basic level

Data Setup	Human-Eval	LLM-Eval
Single Store	41.3	40.7
Shared Store	19.3	21.3

Table 2: Comparison of LLaMa2-70B model QA accuracy (%) between our proposed LLM based evaluation and human evaluation (reported in (Islam et al., 2023)).

of mathematical acumen, e.g., aggregate operating profit amount across a range of years; (iii) long context understanding based on dependencies of financial concepts; and (iv) processing of tabular information, e.g., balance sheets of an organization. The benchmark also reports the performances of several LLMs based QA systems.

Further, the evaluation consists of *two* scenarios:

• **Shared Store** – the entire collection of documents is provided (along with the query) to the LLM-based QA framework for generating the relevant answer. Observe that in an organizational settings, this mimics the *challenging real-life scenario* wherein documents across different product categories and business lines are stored in a common data store. Thus during RAG, the retrieved chunks can span across different documents, introducing disparate contexts and irrelevant information.

• **Single Store** – here only the document(s) relevant to a user query is provided to the LLM for obtaining the final answer. The text chunks extracted during RAG (in this setting) would then only fetch information relevant to the user query, even if they are from different, albeit related, documents.

For completeness of evaluation, two special-case scenarios are also considered: (1) *Closed Book* – wherein no information or documents (from the dataset) are provided to the LLM to assess the "generic" world-knowledge (from training data) of the models; and (2) *Open Book* – where the exact document passage (or chunk) from the correct document(s) is provided to the LLM (along with the question) to assess its reasoning and understanding capabilities. This is referred to as the *Oracle*, and usually represents a performance upper-bound.

Evaluation. It should be noted that the original evaluation of the benchmark dataset involved a panel of finance professional based assessment of the LLM generated answers. However, since this is expensive in practice, we use an automated and more cost-effective method by employing *LLM-as-a-judge* approach (as shown in (Zheng et al., 2023)). To this end, we use LLaMa3.3 and provide the LLM with the user question along with the ground truth answer. To evaluate a new candidate answer, we prompt the LLM to assess if the candidate answer is similar to the ground truth and actually answers the original question.

Data Setup	Framework	Accuracy (%)
Open Book	Oracle*	*74.0*
Closed Book	World Knowledge*	18.0
Shared Store	LLaMa2-70B	21.3
	GPT-4-Turbo	19.3†
	DeepSeek	30.0
	QuARK	**43.3**
Single Store	LLaMa2-70B	40.7
	GPT-4-Turbo	50.0†
	DeepSeek	43.33
	QuARK	**52.0**

† represents results as reported in (Islam et al., 2023)
* Standalone DeepSeek-R1-70B-LLaMa3-Q4_K_M model is used

Table 3: QA Accuracy evaluation of competing frameworks on *FinanceBench* dataset.

Since, the ground truth answers and the outputs (of the different approaches) were provided in the benchmark dataset, in Table 2 we showcase the similarity in accuracy computation between the human assessment provided and the LLM based automated judgment observed in our pipeline. As the evaluation based on an LLM judge was seen to be similar (and in accordance) to that of the human annotations provided, we use this setup for evaluation in the remainder of our experiments.

4 Empirical Results

Setup. All answers generated in our QA framework were performed using the 4-bit K-quantized *DeepSeek-R1-70B-LLaMa3-Q4_K_M* LLM (referred as DeepSeek), which is distilled from the original 70.6B parameter DeepSeek-R1 model which was based on LLaMa3.3-70B-Instruct. LLM inference for answer generation was performed on 2 V100 Nvidia Tesla cards with 32 GB each. For uniformity of evaluation, the LLM context was limited to 2048 tokens for all experiments.

Results. From Table 3, we observe that our proposed QuARK framework provided significant gains in terms of accuracy of the final answers generated by the LLM. Interestingly, in the harder (but more practical) setting of *Shared Store* (wherein all documents are stored in a common vector store), QuARK gains more than **13%** compared to existing QA baselines based on LLaMa and GPT4-Turbo.

Data Setup	**No KG**	**With KG**
Shared Store	30.00	43.30
Single Store	43.33	52.00

Table 4: Effect of KG on QA accuracy (%) in QuARK.

Data Setup	**# KG Triples**	**Accuracy (%)**
Shared Store	1	40.00
	2	**43.30**
	3	40.00
Single Store	1	49.33
	2	**52.00**
	3	45.33

Table 5: Effect of the number of KG triples in QuARK.

Even in the simplified *Single Store* setup, we observe QuARK to outperform the other approaches.

This efficacy of our proposed framework can be attributed to the end-to-end integration of a suite of LLMs working in tandem to: (i) initially preprocess documents, (ii) semantically understand the document contents, (iii) automatically construct domain-specific KG, (iv) user query understanding based on unstructured break-up of required information, (v) RAG powered knowledge retrieval from documents and KG, and (vi) final answer generation based on the provided information prompts.

Since, automated construction of domain-specific KG and its use within our RAG pipeline is crucial for the enhanced performance of QuARK, we perform *ablation* to study the effects of KG usage and retrieval parameters. In Table 4, we see a sharp drop in performance of our framework without the use of the constructed KG. This alludes to the fact, that the constructed KG is indeed of good quality and that the inclusion of such curated information triples help in accurate and precise QA.

Additionally, in Table 5, we study the effects of varying amounts of information retrieved from the KG and provided to the LLM for answer generation. We observe that limited or too much information augmentation (from the KG) both degrade the overall performance of our system. It is understandable, that extremely limited information would not be able to provide sufficient information boost for the LLM to perform better. While, too much information poses the problems of longer prompts and inadvertent noise inclusion. Hence, the optimal setting of extracting 2 KG triples (based on semantic similarity) for each "bullet point" of query decomposition has been used in our experimental setup.

5 Related Work

Use of knowledge graphs within the LLM pipeline has been recently well-explored (He et al., 2024). However, most of the existing techniques rely on pre-curated KGs that are provided – which itself is a costly process. A combination of VectorRAG (on documents) and GraphRAG (on KG) into HybridRAG was proposed to utilize a set of verbs to instantiate the desired relations between entities, demonstrated improvement upon just VectorRAG alone for a financial test corpus (Sarmah et al., 2024). To showcase the utility of merging graphs with RAG retrieval of text chunks, graph construction of customer user issues with the purpose of retaining structures along with linking events that would have otherwise been lost between text chunks, have been explored in (Xu et al., 2024). In addition to textual information, SubgraphRAG demonstrated the usefulness of grounding of RAG responses through the addition of retrieved graph based information (using a multilayer-perceptron for subgraph retrieval) to the LLM (Li et al., 2025). Knowledge Graph prompting for multi-document retrieval also blends knowledge graphs with traditional textual RAG (Wang et al., 2024b). Domain-specific use of KG for medical applications QA systems have been studied in (Yang et al., 2024).

A related and close approach QuARK also proposes the use of LLM to extract knowledge graphs (Chen et al., 2024). However, it requires the expensive process of pre-defined KG schema for clustering of generated triples therein (whereas our approach is schema-free). Further, it does not include the query decomposition stage (of QuARK) for enhanced understanding of questions, and would potentially fail for complex queries.

6 Conclusion

In this paper, we have proposed QuARK, a domain-specific QA framework, operating on a suite of LLMs to tackle – (i) pre-processing of diverse documents, (ii) semantic understand of contents, (iii) automated construction of domain-specific KG, (iv) enhanced user query understanding via decomposition, (v) RAG based knowledge retrieval, and (vi) precise answer generation. We showcase the efficacy of our pipeline by a significant 13% improvement in accuracy on benchmark financial dataset (compared to existing methods) – impressing upon the quality and utility of the automatically constructed domain-specific KG by LLM.

References

Yuemin Chen, Feifan Wu, Jingwei Wang, Hao Qian, Ziqi Liu, Zhiqiang Zhang, Jun Zhou, and Meng Wang. 2024. Knowledge-Augmented Financial Market Analysis and Report Generation. In *Conference on Empirical Methods in Natural Language Processing: Industry Track*, pages 1207–1217.

DeepSeek-AI, Daya Guo, Dejian Yang, Haowei Zhang, Junxiao Song, Ruoyu Zhang, Runxin Xu, Qihao Zhu, Shirong Ma, Peiyi Wang, Xiao Bi, Xiaokang Zhang, Xingkai Yu, Yu Wu, Z. F. Wu, Zhibin Gou, Zhihong Shao, Zhuoshu Li, Ziyi Gao, and 181 others. 2025. DeepSeek-R1: Incentivizing Reasoning Capability in LLMs via Reinforcement Learning. *Preprint*, arXiv:2501.12948.

Qingxiu Dong, Lei Li, Damai Dai, Ce Zheng, Zhiyong Wu, Baobao Chang, Xu Sun, Jingjing Xu, Lei Li, and Zhifang Sui. 2024. A Survey on In-context Learning. In *Conference on Empirical Methods in Natural Language Processing*, pages 1107–1128.

Aaron Grattafiori, Abhimanyu Dubey, Abhinav Jauhri, Abhinav Pandey, Abhishek Kadian, Ahmad Al-Dahle, Aiesha Letman, Akhil Mathur, Alan Schelten, Alex Vaughan, Amy Yang, Angela Fan, Anirudh Goyal, Anthony Hartshorn, Aobo Yang, Archi Mitra, Archie Sravankumar, Artem Korenev, Arthur Hinsvark, and 542 others. 2024. The Llama 3 Herd of Models.

Xiaoxin He, Yijun Tian, Yifei Sun, Nitesh V. Chawla, Thomas Laurent, Yann LeCun, Xavier Bresson, and Bryan Hooi. 2024. G-Retriever: Retrieval-Augmented Generation for Textual Graph Understanding and Question Answering. In *International Conference on Neural Information Processing Systems*, pages 132876–132907.

Lei Huang, Weijiang Yu, Weitao Ma, Weihong Zhong, Zhangyin Feng, Haotian Wang, Qianglong Chen, WeihuaTo Peng, Xiaocheng Feng, Bing Qin, and Ting Liu. 2025. A Survey on Hallucination in Large Language Models: Principles, Taxonomy, Challenges, and Open Questions. *ACM Trans. Inf. Syst.*, 43(2):1–55.

Pranab Islam, Anand Kannappan, Douwe Kiela, Rebecca Qian, Nino Scherrer, and Bertie Vidgen. 2023. FinanceBench: A New Benchmark for Financial Question Answering. *Preprint*, arXiv:2311.11944.

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. 2020. Retrieval-Augmented Generation for Knowledge-intensive NLP Tasks. In *International Conference on Neural Information Processing Systems*.

Mufei Li, Siqi Miao, and Pan Li. 2025. Simple Is Effective: The Roles of Graphs and Large Language Models in Knowledge-Graph-Based Retrieval-Augmented Generation. In *International Conference on Learning Representations*.

OpenAI, Josh Achiam, Steven Adler, Sandhini Agarwal, Lama Ahmad, Ilge Akkaya, Florencia Leoni Aleman, Diogo Almeida, Janko Altenschmidt, Sam Altman, Shyamal Anadkat, Red Avila, Igor Babuschkin, Suchir Balaji, Valerie Balcom, Paul Baltescu, Haiming Bao, Mohammad Bavarian, Jeff Belgum, and 262 others. 2024. GPT-4 Technical Report. *Preprint*, arXiv:2303.08774.

Shirui Pan, Linhao Luo, Yufei Wang, Chen Chen, Jiapu Wang, and Xindong Wu. 2023. Unifying Large Language Models and Knowledge Graphs: A Roadmap. In *International Conference on Computational Linguistics*.

Bhaskarjit Sarmah, Benika Hall, Rohan Rao, Sunil Patel, Stefano Pasquali, and Dhagash Mehta. 2024. HybridRAG: Integrating Knowledge Graphs and Vector Retrieval Augmented Generation for Efficient Information Extraction. In *International Conference on AI in Finance*, pages 608–616.

Gemini Team, Petko Georgiev, Ving Ian Lei, Ryan Burnell, Libin Bai, Anmol Gulati, Garrett Tanzer, Damien Vincent, Zhufeng Pan, Shibo Wang, Soroosh Mariooryad, Yifan Ding, Xinyang Geng, Fred Alcober, Roy Frostig, Mark Omernick, Lexi Walker, Cosmin Paduraru, Christina Sorokin, and 1118 others. 2024. Gemini 1.5: Unlocking Multimodal Understanding across Millions of tokens of Context. *Preprint*, arXiv:2403.05530.

Granite Vision Team, Leonid Karlinsky, Assaf Arbelle, Abraham Daniels, Ahmed Nassar, Amit Alfassi, Bo Wu, Eli Schwartz, Dhiraj Joshi, Jovana Kondic, Nimrod Shabtay, Pengyuan Li, Roei Herzig, Shafiq Abedin, Shaked Perek, Sivan Harary, Udi Barzelay, Adi Raz Goldfarb, Aude Oliva, and 44 others. 2025. Granite Vision: A Lightweight, Open-source Multimodal Model for Enterprise Intelligence. *Preprint*, arXiv:2502.09927.

Liang Wang, Nan Yang, Xiaolong Huang, Linjun Yang, Rangan Majumder, and Furu Wei. 2024a. Multilingual E5 Text Embeddings: A Technical Report. *Preprint*, arXiv:2402.05672.

Yu Wang, Nedim Lipka, Ryan A. Rossi, Alexa Siu, Ruiyi Zhang, and Tyler Derr. 2024b. Knowledge Graph Prompting for Multi-Document Question Answering. In *AAAI Conference on Artificial Intelligence*, pages 19206–19214.

Yuxin Wu, Alexander Kirillov, Francisco Massa, Wan-Yen Lo, and Ross Girshick. 2019. Detectron2. `https://github.com/facebookresearch/detectron2`.

Zhentao Xu, Mark Jerome Cruz, Matthew Guevara, Tie Wang, Manasi Deshpande, Xiaofeng Wang, and Zheng Li. 2024. Retrieval-augmented generation with knowledge graphs for customer service question answering. *Proceedings of the 47th International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 2905–2909.

An Yang, Anfeng Li, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chang Gao, Chengen Huang, Chenxu Lv, Chujie Zheng, Dayiheng Liu, Fan Zhou, Fei Huang, Feng Hu, Hao Ge, Haoran Wei, Huan Lin, Jialong Tang, and 41 others. 2025. Qwen3 Technical Report. *Preprint*, arXiv:2505.09388.

Rui Yang, Haoran Liu, Edison Marrese-Taylor, Qingcheng Zeng, Yu He Ke, Wanxin Li, Lechao Cheng, Qingyu Chen, James Caverlee, Yutaka Matsuo, and Irene Li. 2024. KG-Rank: Enhancing Large Language Models for Medical QA with Knowledge Graphs and Ranking Techniques. In *Workshop on Biomedical Natural Language Processing*, pages 155–166.

Lianmin Zheng, Wei-Lin Chiang, Ying Sheng, Siyuan Zhuang, Zhanghao Wu, Yonghao Zhuang, Zi Lin, Zhuohan Li, Dacheng Li, Eric P. Xing, Hao Zhang, Joseph E. Gonzalez, and Ion Stoica. 2023. Judging LLM-as-a-judge with MT-bench and Chatbot Arena. In *International Conference on Neural Information Processing Systems*, pages 46595–46623.