A Comprehensive Survey on Integrating Large Language Models with Knowledge-Based Methods

Lilian Some, Wenli Yang, Michael Bain, Byeong Kang

[a] *University of Tasmania, Churchill Ave, Hobart, 7005, Tasmania, Australia*
[b] *University of Tasmania, Churchill Ave, Hobart, 7005, TAS, Australia*
[c] *University of New South Wales, High St, Sydney, 2052, NSW, Australia*
[d] *University of Tasmania, Churchill Ave, Hobart, 7005, TAS, Australia*

Abstract sjkshjsdhshkhsjkhd

The rapid development of artificial intelligence has brought about substantial advancements in the field. One promising direction is the integration of Large Language Models (LLMs) with structured knowledge-based systems. This approach aims to enhance AI capabilities by combining the generative language understanding of LLMs with the precise knowledge representation of structured systems. This survey explores the synergy between LLMs and knowledge bases, focusing on real-world applications and addressing associated technical, operational, and ethical challenges. Through a comprehensive literature review, the study identifies critical issues and evaluates existing solutions. The paper highlights the benefits of integrating generative AI with knowledge bases, including improved data contextualization, enhanced model accuracy, and better utilization of knowledge resources. The findings provide a detailed overview of the current state of research, identify key gaps, and offer actionable recommendations. These insights contribute to advancing AI technologies and support their practical deployment across various sectors.

Keywords: LLMs, Knowledge-Based, Knowledge Integration, RAG

1. Introduction

The rapid advancements in Large Language Models (LLMs) have significantly transformed the field of artificial intelligence. These models demonstrate unprecedented proficiency in understanding and generating human-like text. Built on deep learning architectures, LLMs excel in various natural language processing tasks, including text generation, sentiment analysis, and complex dialogue systems.

Recent surveys have explored diverse aspects of LLMs, such as their architectures, training methodologies, and performance evaluation benchmarks. Many focus on specific topics, such as detailed analyses of state-of-the-art models [1], innovations in scaling laws [2], and pre-training techniques on large datasets [3]. Others examine domain-specific fine-tuning [4], reinforcement learning from human feedback [5], and transfer learning strategies [6]. Despite these valuable contributions, there remains a lack of holistic perspectives that connect foundational principles, practical applications, and challenges associated with implementing LLMs in real-world scenarios.

This survey addresses this gap by presenting a comprehensive analysis of LLMs' foundational principles and their applications across diverse domains. Although LLMs have achieved remarkable progress, their practical deployment faces challenges. Issues such as interpretability, high computational demands, and scalability impede their broader adoption. This study also investigates the integration of generative AI with knowledge bases, emphasizing how this synergy can mitigate these limitations and unlock new opportunities.

To guide this analysis, several key assumptions are outlined:

Challenges in Real-World Applications: LLMs encounter substantial obstacles in real-world settings, particularly in terms of interpretability, computational requirements, and scalability, which limit their effectiveness and broader applicability.

Mitigation through Integration: The integration of LLMs with knowledge bases—using methods such as Retrieval-Augmented Generation (RAG), Knowledge Graphs, and Prompt Engineering—offers promising solutions. This synergy enhances data contextualization, improves model accuracy, and reduces computational costs.

Barriers to Adoption: Persistent challenges, including the need for interpretability, efficient resource utilization, and seamless integration with existing systems, continue to hinder the widespread adoption of LLMs.

Building on these assumptions, this survey provides a structured and integrated analysis of LLMs. The primary contributions are as follows:

- Providing a comprehensive overview of LLMs, with a focus on their foundational principles and architectural variations.

- Analyzing the practical applications of LLMs across diverse domains, emphasizing their real-world impacts.

- Critically evaluating the technical, ethical, and operational challenges associated with implementing LLMs, alongside potential solutions.

- Investigating the enhancement of LLMs through integration with external knowledge bases, identifying opportunities and challenges in this approach.

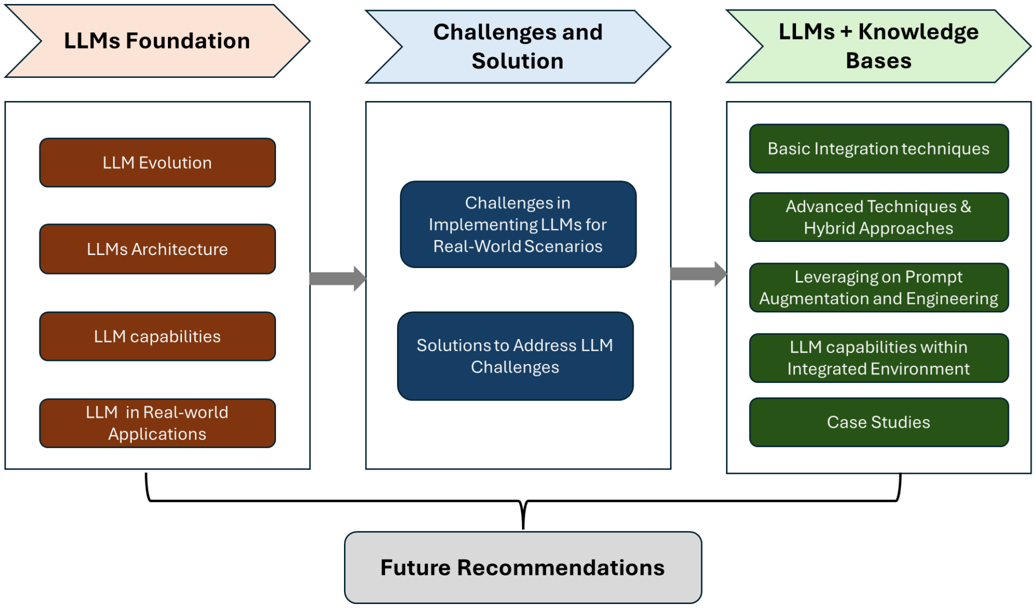
LLMs Foundation
Challenges and Solution
LLMs + Knowledge Bases
LLM Evolution
LLMs Architecture
LLM capabilities
LLM in Real-world Applications
Challenges in Implementing LLMs for Real-World Scenarios
Solutions to Address LLM Challenges
Basic Integration techniques
Advanced Techniques & Hybrid Approaches
Leveraging on Prompt Augmentation and Engineering
LLM capabilities within Integrated Environment
Case Studies
Future Recommendations

Figure 1: Overall Structure of the Paper Organization

This survey paper begins with an overview of Large Language Models (LLMs), detailing their evolution, underlying architecture, and diverse capabilities. It then transitions to exploring LLMs in real-world applications, highlighting the challenges of implementing these models, categorized into technical, operational, and ethical/social dimensions, along with potential solutions to address these issues. Building on the foundational understanding of LLMs, the section on Integrating LLMs with Knowledge Bases outlines integration techniques designed to enhance LLMs with structured knowledge, advanced hybrid approaches, prompt augmentation strategies, and the resulting capabilities within integrated environments. This section also offers practical insights through case studies to examine real-world applications. Finally, the paper concludes with Recommendations for Future Development and Implementation, providing actionable guidance to help advance the field. Figure 1 shows the overall structure of the paper organization.

2. Overview of LLMs

2.1. LLM Evolution

Large Language Models (LLMs) are artificial intelligence models specializing in understanding and generating human-like text. They are characterized by their massive size, encompassing billions of parameters and trained on vast amounts of unlabelled textual data, enabling them to learn intricate language patterns and semantic relationships [7]. Figure 2 visually represents the progression of LLMs over time, highlighting key milestones and advancements in AI language processing.

LLMs are predominantly built on the transformer architecture, which leverages on self-attention mechanism to effectively process long text sequences [8], [9]. LLMs have undergone evolutionary phases starting from basic statistical language models, and progressing to sophisticated neural language models, then to Pre-Trained Language Models, culminating in the development of contemporary Large Language Models. The foundation stage of LLM systems springs from early Natural Language Processing (NLP) models such as n-grams, and TF-IDF (Term Frequency Inverse Document Frequency), alongside classical machine learning algorithms like Naive Bayes and Support Vector Machines (SVMs). These early methods focused on statistical methods of predicting words by proximity and frequency; for instance, n-grams predicted the next word based on previous sequences [10], [7]. Initially, these systems relied heavily on rule-based algorithms designed by domain experts and the approach was rather difficult, expensive, and required extensive human effort in feature engineering [11]. Cognizant of these limitations, the research focus shifted to learning-based models that could automatically extract patterns from data, reducing reliance on manually crafted rules [12].

The mid-2010s ushered the transition to the Neural Language Model (NML) phase, where NLPs would predict the probability distribution of the next text given the previous words in sequence, utilizing deep learning models such as Convolutional Neural Networks (CNNs), Recurrent Neural

Networks (RNNs), Long Short-Term Memory networks (LSTMs), Gated Recurrent Units (GRUs), and attention-based Sequence-to-Sequence (Seq2Seq) architectures [13]. These models, enabled a deeper understanding of human language through learning vector representations of words, also known as word embeddings [10], [7].

Subsequently, Pre-Trained Language Models (PLMs) like BERT and ELMo emerged as a pivotal development towards the current powerful LLMs. PLMs leveraged the emergence of Neural networks to vectorize words and understand the context in which words occurred. These models pre-trained on massive text datasets which facilitated a more nuanced understanding of language and transfer learning, where the models could be fine-tuned for specific downstream tasks with minimal data [7]. Nonetheless, the limitations of prior statistical and Neural models led to the introduction of the transformer revolution, characterized by the development of sophisticated deep-learning models. The transformer architecture, which is fully based on an attention mechanism, supports parallel processing and more efficient handling of long-range dependencies, enabling advanced language modelling capabilities such as generating human-like text, translating languages, summarizing complex information, and even composing various types of creative content [12], [14], [7], [13].

2.1.1. Emergence of Groundbreaking Models

Groundbreaking models such as the BERT and the GPT series were made possible by the transformer architecture [7], [14], [7], [13], [15]. Leveraging on Masked Language Modelling and bidirectional training, BERT (Bidirectional Encoder Representations from Transformers) significantly increased performance in a range of NLP tasks and marked a significant leap in the ability of language models to understand context [7], [13], [14]. The GPT (Generative Pre-trained Transformers) series, particularly GPT-3, pushed the limits of what LLMs could accomplish with their outstanding text production and comprehension skills [7], [16], [17]. Together, BERT and the GPT series have laid the groundwork for further advancements in LLMs. Their success has spurred research into model scaling, multimodal integration, domain-specific applications and ethical considerations [18], [19].

2.1.2. Current Trends and Innovations in LLM Development

Over the years, various models introduced unique features that advanced the field. Several notable firsts have been achieved by pre-trained Language

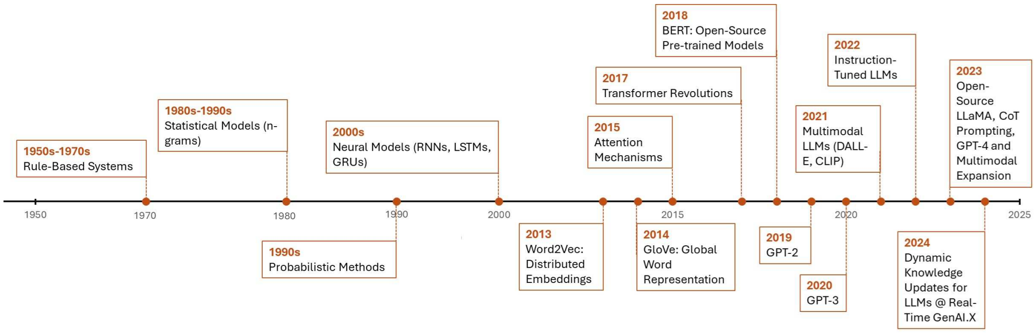
1950s-1970s
Rule-Based Systems
1980s-1990s
Statistical Models (n-grams)
1990s
Probabilistic Methods
2000s
Neural Models (RNNs, LSTMs, GRUs)
2013
Word2Vec: Distributed Embeddings
2014
GloVe: Global Word Representation
2015
Attention Mechanisms
2017
Transformer Revolutions
2018
BERT: Open-Source Pre-trained Models
2019
GPT-2
2020
GPT-3
2021
Multimodal LLMs (DALL-E, CLIP)
2022
Instruction-Tuned LLMs
2023
Open-Source LLaMA, CoT Prompting, GPT-4 and Multimodal Expansion
2024
Dynamic Knowledge Updates for LLMs @ Real-Time GenAI.X
1950
1970
1980
1990
2000
2015
2020
2025

Figure 2: The Evolution of Large Language Models

Models (PLMs). In 2018 OpenAI's GPT was the first to implement a transformer architecture for auto-regressive text generation, while Google's BERT introduced bidirectional training for enhanced contextual understanding [20]. GPT-2, which showed the benefits and hazards of large-scale text generation, and Google's T5 [7], [21] which combined NLP jobs onto a single platform in 2020, both advanced the field. In 2019, Facebook AI published RoBERTa, an optimized version of BERT, and in 2020, OpenAI released GPT-3, which has 175 billion parameters and sets new norms in few-shot learning [14], [7], [13]. Other notable models in this period included the debut of BART by Facebook AI, Longformer by the Allen Institute and Google Research's Reformer which were optimized for distinct areas of NLP jobs, such as text generation and effective handling of large sequences [14], [15].

Open-source initiatives, such as EleutherAI's GPT-Neo and GPT-J, emerged in 2021 to promote democratic LLMs; subsequently, the BLOOM model introduced in 2022 was a collaborative multilingual open-access model. In 2023, Meta AI's LLaMA series emphasized efficiency, whilst Anthropic launched Claude to promote safety and alignment [21]. The Falcon series from Mistral released in 2024 had an emphasis on high performance and open-source accessibility. Meanwhile, Google DeepMind's Gemini integrated improved reasoning and multimodal capabilities. This landscape depicts the rapid diversification and expansion of LLMs, underscoring ongoing efforts to balance innovation, accessibility, efficiency, and moral considerations. Nevertheless, these developments have completely transformed LLMs' capacity to comprehend, produce, and interact with human language [14], [22], [19].

2.1.3. Advancements in OpenAI's GPT Models

The OpenAI GPT series has dramatically reshaped language understanding and generation. The transition from foundational GPT-1 to GPT-4 has demonstrated the power of scaling up transformer architectures while facing ongoing challenges with factual accuracy and bias. GPT-1 launched in 2018, was a radical point in demonstrating the capabilities of transformer-based models in solving NLP tasks such as auto-regressive text generation. It acted as the foundation for current GPTs, having commenced with only 117 million parameters and laid a foundation for earlier LLM models. This advancement demonstrated that computing comprehension of language might be enhanced by pre-training a model on a corpus of data without supervision and then optimizing it to produce human-like text, answer questions, and perform tasks like translation and summarization [7]. However, its abilities remained capped despite the notable advancements, particularly in managing increasingly intricate assignments.

With a significant jump to 1.5 billion parameters, GPT-2's size and performance improved dramatically at its inception in 2019. Text generation, summarizing, and more realistic conversational engagement were the possibilities of this sophisticated model, which generated text that was more coherent and context-aware. Additionally, few-shot learning was supported by GPT-2, allowing the model to produce excellent content with little input; nonetheless, it was plagued with accuracy issues, though, occasionally producing plausible text.

With its staggering 175 billion parameters when it was released in 2020, GPT-3 was a ”game-changer” and one of the most potent models at the time. It performed very well on a variety of tasks, including writing, coding, and deciphering challenging challenges [23], [7]. The capabilities included zero-shot and few-shot learning, where the model could perform tasks without further fine-tuning. The versatility of GPT-3 meant it could be applied in many varied forms, from chatbots and virtual assistants to creative writing tools. Despite this, it exhibited many factual inaccuracies due to the huge, sometimes uncaring nature of the dataset it trained on, often resulting in what is known as 'hallucination'. It also propagated biases from the training data, and due to its large size, it is computationally expensive to run.

GPT-4, released in 2023, further advanced the capabilities of its predecessor, although OpenAI has not revealed the parameter size. Its distinguishing feature is its multimodal capability in that it can process not just text but also images, thereby expanding usefulness into new areas such as image captioning and visual question-answering. GPT-4 continued running generalized tasks associated with content creation, coding, and conversational dialogue. Further tuning of GPT-3 on output control is done through superior reinforcement learning from human feedback that helps reduce mistakes and lends reliability to its responses. Just like its previous models, however, GPT-4 produced a certain number of mistakes on several occasions with the view of reflecting biases from training [21].

2.2. LLM Architecture

The transformer has become the predominant architectural design for LLMs surpassing convolutional and recurrent neural networks' performance in understanding language and natural language generation [24], [25]. The Transformer architecture is the best suited for LLMs because of their capacity to comprehend and generate natural language by learning intricate word patterns of context and meaning. Additionally, transformers are conducive for training on large corpora and have abundant computational capacity with parallel computation. The architecture can be easily adapted for specific tasks with robust performance leading to large gains on downstream tasks like text classification, language understanding, reference resolution, common-sense inference, summarization, and machine translation [7], [25]. The architecture's training on vast corpora—including books, online forums, and publicly available websites like Wikipedia—enables it to produce contextually appropriate and coherent responses. However, this advantageous gain has a wide range of practical challenges to be addressed for the model's effective utilization. For instance, need for training, analysing, scaling and augmentation of models across platforms. Since transformers are the building block for many research and applications, there is also a need for distribution, fine-tuning, and deployment within the AI industry [14].

Building upon the transformer, LLMs are typically designed using one of three main architectures: encoder, decoder and encoder-decoder [26], [27] as shown in Figure 3. These architectures showcase the agility of transformers and their impact beyond NLP.

Encoder LLMs include models like BERT, CodeBERT, and GraphBERT along with other models like ALBERT, RoBERTa, and ELECTRA [11], [28]. The encoder-LLMs denote a category of LLMs that train the encoder to generate a fixed-dimensional bidirectional model based on Masked Language Modelling (MLM) and Next Sequence Prediction (NSP). MLM

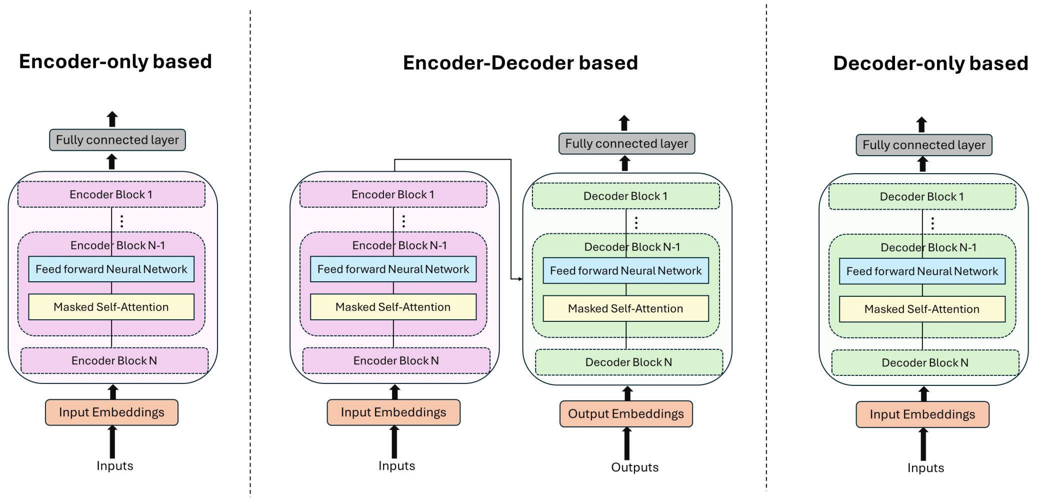

Figure 3: Overview of LLM Architectures

predicts the initial tokens that are already masked, while NSP predicts if the given sentences occur sequentially in a text [14], [29].

Bidirectional Encoder Representation Transformer (BERT) is based on an encoder architecture that captures the bi-directional context of unlabelled text instead of right to left in all contexts [20], [14]. As a result, the underlying mechanism leverages multi-modal encoder layers' self-attention mechanism to generate deep contextual relationships. Its training methodology is based on masked language modelling (MLM) and next sentence prediction (NSP) and trains large corpora to learn contextual relationships. BERT is fine-tuned for specific downstream tasks with just one additional output layer to create a sophisticated model for numerous tasks such as inference, and question answering without consequential architectural modification; its concept is simple yet empirically powerful [8]. These models often require an additional prediction head to handle specific downstream tasks.

Decoder LLMs like GPT-3 and GPT-4, LLaMA and PaLM models, train the decoder component to support auto-regressive text [19], [30]. GPT models capture the uni-directional context of words to generate the next word, given all the prior words. GPT's underlying mechanism is the multiple transformer decoder layers with a self-attention mechanism, feed-forward neural networks, layer normalization, and residual connections [14]. The self-attention mechanism allows each word to focus on every other word in the input sequence, allowing the model to capture long-range dependencies and contextual information effectively [31], [14]. Decoder LLMs are geared towards text generation and have been part of significant improvements in few-shot learning. Their development reflects a movement towards models that are increasingly fine-tuned for particular tasks [32], [19].

Encoder-decoder LLMs such as T5 and BART train both encoder and decoder to support text-to-text generation tasks i.e., translation, summarization, and classification, with versatility in handling comprehension and generation [32], [33], [26]. T5 LLM reformulates all text-based language issues as text-to-text problems, is trained in modified Masked Language modelling (MLM) called 'span corruption' and is fine-tuned on a range of NLP tasks using labelled datasets into text-to-text format.

Furthermore, various specialized transformers are designed to solve particular challenges i.e. Reformer, Longformer models and Knowledge enhanced LLMs. Introduced in 2020, the reformer architecture modifies the current transformer architecture to reduce its memory and computational cost [34]. The architecture leverages locality-sensitive hashing and reversible layers to handle long sequences efficiently. This solves the problem of the short attention span experienced by the transformer. Longformer uses a locality-sensitive attention method to enhance the processing of lengthy textual inputs. This approach improves efficiency for extended sequences by allowing each token to attend exclusively to its globally significant tokens and its relevant local environment [18]. A more general trend in LLM development is the move from encoder-decoder architectures with multiple uses to more task-specific, decoder-only models that are tailored for tasks like text production. Additionally, the democratization of LLMs—facilitated by open-source initiatives—is improving accessibility and diversity in research and development. However, it is crucial to remember that existing viewpoints on the evolution of LLMs might not adequately account for more recent or less well-established models. The shift towards open-source models such as LLaMA and BLOOM represents a democratization of LLM research, promoting transparency, reproducibility, and collaborative development [35]. These models serve as foundational platforms for building domain-specific adaptations, reducing dependency on proprietary models, and fostering innovation across academia and industry. This collaborative approach is crucial for overcoming the limitations of closed models and ensuring that the advancements in LLM technology benefit a broader range of stakeholders [21].

2.3. LLM Capabilities

Large language models (LLMs) are flexible tools that can be used for a variety of tasks due to their broad range of abilities. Their foundational capability is text generation, but it can be broadly categorized into areas such as language understanding and generation, reasoning and decision-making, knowledge management, multimodal processing, and adaptability [16], [14]. . The table 1 provides a structured summary of the key capabilities of AI systems, organized into distinct categories with corresponding examples.

LLMs are proficient at natural language processing (NLP) tasks like language translation, summarization, and text completion, making them useful in producing responses that are both intelligible and appropriate for the given context [9], [36], [22]. They may also engage in complicated reasoning tasks, such as question answering (QA), solving issues, and providing explanations across multiple topics. Moreover, LLMs can be fine-tuned for specific tasks like authoring scientific papers, sentiment analysis, and code production. They also excel in large-scale information retrieval, extracting, synthesizing, and summarizing data from various sources. LLMs are even capable of producing creative works such as conversations, storytelling, and poetry, showcasing their ability to mimic human creativity. Their ability to integrate external knowledge bases like ERNIE or E-BERT further enhances their capacity to provide current or specialized information.

LLMs' potential is unlimited, particularly in understanding and generating human language, opening opportunities across many domains [37]. Their use has expanded from scientific domains to the general public, facilitated by chat-based interfaces and applications where interactions occur through natural language [16]. With in-context learning, LLMs can adapt quickly to new tasks through few-shot learning without needing extensive fine-tuning [22]. The integration of multimodal capabilities further enhances their scope, allowing LLMs to process and understand information from multiple modalities such as text, images, audio, and video for richer, contextually aware outputs [38], [39].

Prompting techniques like chain-of-thought enable LLMs' reasoning, enabling them to tackle complex tasks through logical progression [40]. Additionally, LLMs transcend specific disciplines and can be applied broadly rather than being confined to niche areas. The emergence of domain-specific LLMs, such as Financial LLMs and Medical LLMs , highlights a trend toward specialization. These models are designed to address specific industry needs by leveraging highly specialized knowledge [22], [22]. Lastly, LLMs exhibit autonomous agent capabilities that interact with the environment, planning actions, and executing tasks based on natural language instructions [41]. Such agents include those that can control software applications, access external knowledge bases, and even perform physical actions in the real world [38], [41].

Despite their wide-ranging capabilities, LLMs face limitations, particularly when applied to highly specialized fields like healthcare, law, medicine, or journalism. These areas often require additional data, resources, and supplementary technology to achieve optimal results [16]. LLMs also exhibit shortcomings in reasoning, factual consistency, and the demand for computational resources, especially when handling complex or domain-specific content without fine-tuning. In such cases, the models may require significant additional inputs to perform adequately. Several authors also note that while LLM technological capabilities are expanding at an astonishing pace, their limitations still pose significant challenges [37].

By empowering machines to create data from existing data, LLMs promise to revolutionize various sectors, including healthcare, entertainment, finance, creative arts, and research. Their ability to produce plausible text from vast datasets allows them to solve creative problems where no 'correct' solution exists [16]. This capability to generate content without predefined answers sets them apart from traditional machine learning models, which typically focus on solving problems with well-defined boundaries and optimal solutions.

2.4. LLMs in Real-world applications

The LLM revolution is often compared to the industrial revolution due to its transformative potential to address some of the world's most pressing challenges. In our discussion, we will focus on key areas where LLMs are making a profound impact, including policy-making, finance, medicine and healthcare, as well as cross-domain applications. These sectors showcase the vast influence of LLMs, from shaping effective public policies and revolutionizing financial strategies to improving healthcare outcomes and enabling interdisciplinary solutions that tackle complex global challenges.

2.4.1. Policy Making

LLMs process vast corpora that facilitate evidence-based policymaking, helping organizations and governments comprehend trends, forecast potential outcomes, and assess the impact of decisions across critical sectors such as the economy, climate change and health. Recent research on LLMs underpins the

Table 1: Categorization of Key Capabilities of LLMs

Category	**Key Capabilities**	**Examples**
Language Understanding and Generation	- Natural language processing (NLP): Translation, summarization, text completion - Creative content generation: Storytelling, poetry, conversations - Document creation	Writing scientific papers, blogs, sentiment analysis, generating reports
Reasoning and Decision-Making	- Complex reasoning: Question answering (QA), logical reasoning, problem-solving - Decision support: Data analysis for insights, strategy planning	Planning in finance, healthcare decision-making
Knowledge Management and Retrieval	- Information retrieval: Data extraction, synthesis, summarization - Knowledge integration: Leveraging external knowledge bases (e.g., ERNIE, E-BERT)	Domain-specific applications in medicine, law, and finance
Code & Program Development	- Code generation: Writing, debugging, optimizing - Automation: Generating scripts, optimizing workflows	Assisting in software development, task automation
Multimodal Processing	- Multimodal integration: Processing and generating outputs combining text, images, audio, and video	Image captioning, audio transcription, video summarization
Adaptability and Learning	- Prompt engineering: Few-shot and zero-shot learning, chain-of-thought reasoning - Domain specialization: Tailoring models for industries (Finance LLMs, Medical LLMs)	Financial analysis, medical diagnosis
Autonomous Functionality	- Autonomous agents: Interacting with environments, planning, executing tasks	Controlling software, robotics, virtual assistants
Sectoral Impact and Creativity	- Cross-industry applications: Transforming healthcare, education, finance, research - Creative problem-solving: Generating solutions for ill-defined challenges	Ideation in creative arts, scientific innovation, entertainment

potential of LLMs to generate insights that aid policy formulation aligned with sustainable goals [16]. Major corporations are taking decisive strides with LLM-powered data at corporate executive levels. While risk analysis has been done, mitigation strategies are still lagging in the preliminary stages [37]. Thus corporations may need to reskill and anticipate workforce changes. However, through the integration of large data with comprehensive analysis, LLMs have the potential to empower stakeholders to address global challenges with precision and efficacy.

LLMs are powerful tools for processing socioeconomic data, identifying social injustice or inequalities and recommending equitable interventions that promote fair distribution of resources. Researchers delineate diverse applications of LLMs with the potential to promote equitable distribution of resources, financial inclusion, and economic enhancement of marginalized communities [10]. Thus, organizations and policymakers can formulate strategies for social justice and equitable growth.

2.4.2. Finance

LLMs have been used to perform various financial tasks. The utility of these models is validated by their superior performance in tasks like linguistic tasks, sentiment analysis, risk management, fraud detection, reports and summarization, financial time series analysis, financial reasoning, stock movement prediction, text classification, agent-based modelling, customer service, content creation, marketing, personal investment advice, regulatory compliance, legal analysis and named entity recognition, where domain expertise and context-specific knowledge are critical [42], [13]. LLMs can be used to develop robo-advisors that provide personalized financial advice to investors, detect fraudulent activities by analysing patterns in financial transactions and identify anomalies. A growing body of research emphasizes the trend towards creating domain-specific LLMs tailored for specialized tasks. Fin-LLMs, such as FinBERT and BloombergGPT, are prime examples of models adapted to handle financial data by leveraging domain-specific corpora such as reports, news, and social media content and incorporating financial-domain prompt engineering [13]. The utility of these models is validated by their superior performance in tasks like sentiment analysis, stock movement prediction, text classification and named entity recognition, where domain expertise and context-specific knowledge are critical. These models provide significant value by improving decision-making, automating tasks, and enhancing the overall efficiency of financial institutions.

2.4.3. Medicine and Healthcare

LLMs can be used in a variety of NLP applications in the medical and healthcare field, such as question-answering (QA) systems, Chatbots, and fact verification. Their ability to process large amounts of data, including medical literature, clinical trials, and patient data, makes them well-suited to identifying potential drug candidates, diagnosing diseases more accurately, medical education, and developing personalized treatment plans [30], [32], [43], [44]. LLMs can be used to support medical research and literature reviews. This includes tasks like gathering and analyzing data from scientific literature and electronic medical records and identifying relevant research articles [10]. LLMs can also improve patient care through applications like Chatbots that provide patients with information about their care and treatment, and tools that assist medical practitioners with tasks like analyzing X-ray images [23]. The specialization models utilized in the medical field include BioGPT [45], Med-PaLM [46], and BioMedGPT, PubMedBERT [47], BlueBERT, SciBERT and ClinicalBERT [17]. These models have demonstrated promising results in tasks such as query answering, relation extraction, and named entity recognition. Furthermore, ChatGPT and LLM agents are already being used to provide information about patient care while LLM-based agents engage in multi-round discussions through role-playing, potentially enhancing LLM expertise and reasoning capabilities [32].

Tool-integrated reasoning, as proposed by [48], where external symbolic computation tools are combined with LLMs to solve complex mathematical problems. These models are designed to interleave natural language reasoning with tool use, significantly enhancing their problem-solving capabilities to provide personalized medication suggestions and psychological consultations [9]. Benefits of tool integration reasoning include enhanced problem solving, improved accuracy and efficiency, transparency and interpretability [9]. Research shows that LLMs have been used to foster the expansion of medical science. This intervention is in full awareness that algorithmic bias, accuracy, and fairness are not-withstanding [23].

Education sector stakeholders potentially may reap the advantages of the rapid advancement of LLMs which provide personalized learning. These LLMs can democratize access to information, thus addressing disparities and global issues. For example, ChatGPT offers significant opportunities through the generation of personalized content that addresses diverse learning needs [21]. LLMs can assist educators by automating grading tasks and facilitate language learning by providing real-time translation, grammar correction, and personalized language exercises [49], [50]. However, the authors highlight concerns related to potential biases that may lead to misinformation; hence, the need for responsible and equitable use of LLMs.

In humanitarian and crisis management, LLMs have the potential to analyze data from news, media, and reports to provide real-time insights for need identification and resource allocation during pandemics, natural disasters, and other humanitarian emergencies [30]. LLMs can be leveraged to assist in identifying the most critical needs and prioritizing resource allocation [22], developing effective communication strategies, and developing predictive models that anticipate the spread of disease outbreaks or the impact of natural disasters. These examples highlight the potential of LLMs to impact crisis management, by optimizing resource allocation and communication strategies.

2.4.4. Cross domain LLMs

While there is no explicit definition of "cross-domain LLMs," the discussions around mixed-domain training and the use of external knowledge bases suggest that cross-domain LLMs are models designed to function effectively across multiple knowledge domains. Cross-domain LLMs exhibit most of these features: knowledge generalization across domains, multitasking, transfer learning and fine-tuning, scalability [51], [52], [53], [54]. LLMs are inherently versatile and adaptable across platforms, with applications in legal document analysis, customer service, and academic research. FinLLMs, for instance, utilize mixed-domain LLMs trained on both general and financial corpora, incorporating prompt engineering and instruction fine-tuning to adapt models to specific financial tasks. Similarly, MedLLMs apply LLM technology to healthcare, leveraging models like PharmacyGPT and Psy-LLM [32].

3. Challenges in Implementing LLMs for Real-World Scenarios

Implementing LLMs in real-world applications presents numerous challenges that span technical, operational, and ethical dimensions. Despite the extensive capabilities, LLMs experience limitations in the following areas: accuracy and reliability, explainability, reliance on data lacking updating mechanisms, and broader concerns such as information provenance, privacy, data security, and potential plagiarism. These challenges necessitate ongoing research and development to ensure the responsible and effective use of LLMs in real-world contexts.

3.1. Technical Challenges

Technical challenges with LLMs include model interpretability, as their complex decision-making processes are difficult to understand, and the need for substantial computational resources, which can be costly and inefficient.

3.1.1. Model Interpretability

Even though LLM models show remarkable performance across a range of tasks, it is still challenging to comprehend the logic underlying the text predictions or generation. Concerns arise from this lack of openness, particularly in important applications where justifiable decision-making is imminent as often users seek to understand how and why a model is returning specific outputs. Such transparency helps build or create confidence, ensures ethical application, or encourages further innovation on the platform [55]. Other concerns are caused by model complexity explainability in decision making, bias [56], debugging and error identification [57], trust in high-stake use [19] and ethical concerns. More so, LLM models are referred to as 'black boxes' because they are intrinsically complex and difficult to decipher [21], [10]. The models frequently have millions to billions of parameters, resulting in a complex network of linked nodes that support generative functionality. Because of this complexity, it is difficult for humans to trace the specific model's inputs and their outputs [52]. An example of such a problem is GPT-3, which has 175 billion parameters. LLMs are trained using deep learning methods, often called "black-box" models, which means that even experts cannot fully access or understand their internal mechanisms. This lack of transparency hinders efforts to comprehend how particular inputs lead to specific outcomes [21]. LLM interpretability poses challenges to trust and adoption. Thus, in many real-world use cases, it would be advantageous for the stakeholders to understand and justify what the LLMs are doing, especially in areas that are strictly governed by laws, such as medicine, finance, and legal systems [58]. It is equally challenging but important to ensure that LLMs are not perpetuating prejudices or rendering unjust or discriminatory outcomes if we do not comprehend the LLM decision-making process.

3.1.2. Managing Computational Resources

The development of LLMs remains overly complex and needs robust computational power for both training and inference operations. The associated computational costs in LLMs range from training cost, inference time, resource restraints in IDE, and token costs in conversation-style APR, prompting length costs to optimization of multiple sub-tasks. Training big language models is an expensive undertaking that calls for a lot of computer power. For instance, 64 32-GB V100 GPUs were utilized in training Microsoft's InferFix model [19]. This underscores the necessity of significant hardware investments, which may be a challenge for investors and institutions with constrained budgets. Sources cite that energy consumption during training is a growing concern due to the carbon footprint being generated by the AI industry, raising environmental sustainability issues [23], [59], [60]. Additionally, longer inference times can cause delays in response generation, even when larger LLM's hardware is well equipped [24]. In real-time applications like chatbots, virtual assistants and interactive systems where prompt replies are essential, this could be troublesome.

Moreover, LLMs require substantial storage and runtime memory, and integrating them into Integrated Development Environments (IDEs) can be difficult, posing challenges for code completion, bug-detection and code generation [61]. Performance problems may arise from coding LLMs, particularly on devices with limited resources. This demand for high computational power can limit their use, especially for small organizations [62]. In addition, the cost of generated tokens and input can add up in conversation-style Automated Program Repair (APR), where LLMs work together iteratively to produce fixes. This can result in considerable costs [19].

3.2. Operational Challenges

Operational challenges are equally significant, particularly concerning the integration of LLMs with existing infrastructures and ensuring scalability and reliability; maintenance and updates are prevalent across domains.

3.2.1. Integration with Existing Systems

The LLM integration process is not simply about connecting software or systems; the process requires a multifaceted approach that considers numerous factors, including architectural compatibility, data flow management, and operational efficiency.LLMs are typically based on intricate deep learning architectures, which may not easily interface with systems that may use outdated technologies or have distinct architectural designs. For instance, integrating with legacy systems that do not have access to contemporary APIs or that use different data formats can be quite difficult [63].

Concurrently, during the integration process, data flow management between LLMs and systems is vital for data consistency [64], [65], [66]. Research reiterates the significance of improving data access and suggests time-and-space efficient methods for LLMs as it impacts cost, latency, and hardware strain because integration processes require a large amount of processing power [22], [65]. Nevertheless, it may not always be possible to effectively optimize operational LLMs for specific scenarios since it demands a significant amount of computational resources. Therefore, it is essential to lessen the pre-training load and increase retrieval efficiency to improve operational efficiency in production-ready systems [67].

3.2.2. Scalability and Reliability

Scalability refers to an LLM system's capacity to handle increasing workloads and data volumes while maintaining performance, whereas reliability concerns the system's ability to consistently produce accurate and trustworthy outputs, regardless of the operating environment or task complexity [68]. Research investigates computational demands, data complexity sensibility to model fine-tuning and adaptability as the major concerns for scaling LLMs. Scalability is a salient feature exhibited by LLMs that enables them to effectively manage workload increase, which becomes challenging because of the enormous amount of processing power needed for both training and inference. Maintaining performance while scaling becomes more difficult as the model size and data volume increase. This frequently leads to increased expenses, resource usage, and possible processing bottlenecks [25], [69]. For example, to ensure that jobs are equitably dispersed over various servers and that network latencies are kept to a minimum when growing LLMs across distributed infrastructures, complex orchestration is needed. The difficulty of guaranteeing reliability—the system's capacity to operate consistently under a range of loads and conditions—also confronts large-scale deployments. Due to LLM's intricate structures, reliance on knowledge bases, and hardware constraints, LLMs are prone to malfunctions [7]. LLM systems must be created to function at various levels of throughput and be optimal in any operating environment. This is particularly critical in industries where availability and up-time are critical, such as the computer, automation, manufacturing, navigation, and software industries [62]. Striking a balance between data inputs involves multiple trial-and-error attempts, making the fine-tuning process difficult. This unpredictable nature of model training underscores the need for flexible LLM infrastructure and subsequently the computing power [70]. Multi-bucketing and continuous batching techniques optimize the computational efficiency of LLMs by selecting appropriate buckets for data input and enhance multi-text request processing thus, using computational resources efficiently. These problems are mitigated by constructing a perspective and robust infrastructure, incorporating redundancy, continuous management, and iterative improvement of the system. Robust error handling, redundancy methods, and system monitoring are strategies that provide continuous availability and minimize downtime in high-demand systems. Additionally, upholding scalability and reliability calls for eliminating failure points, and preventing service degradation to support LLM functions in a production environment.

3.2.3. Maintenance and Updates Challenges

Large language models (LLMs) present maintenance and updating difficulties that must be resolved to maintain system stability and dependability. To evolve and address ongoing LLM problems, models undergo continual updates. However, ensuring compatibility with new versions, managing model updates without disrupting existing workflows, and addressing security vulnerabilities require robust maintenance procedures [66]. Robust maintenance procedures are necessary to provide compatibility with new versions, manage upgrades without interfering with existing operations, and address security risks as models undergo regular modifications to adapt and handle ongoing LLM problems [66], [58], [71]. Managing numerous versions of LLMs requires efficient version control and rollback methods that enable prompt recovery in the event of malfunctions by rolling back to earlier versions. Furthermore, to avoid unanticipated problems or performance degradation and to make sure that any changes enhance the system without creating new problems, comprehensive testing and validation of updates before deployment are essential. Both the ongoing system performance monitoring and the gathering of user input preceding LLM updates are crucial because they reveal hidden issues and potential areas for development and allow for prompt adjustments to increase reliability. Adaptability is an essential feature that new LLM versions must incorporate for LLMs to stay useful and efficient in a quickly changing technical environment [32]. Finally, to prevent creating vulnerabilities or jeopardizing sensitive data during updates/upgrades, security and privacy issues must be properly mitigated. Strong security protocols are essential to safeguard information security and maintain user trust during the update process.

3.3. Ethical and Social Implications

LLMs are plagued by multiple ethical and social problems related to bias perpetuation, misinformation, the need for transparency in AI-driven decision-making, and potential impacts on employment. A primary concern is whether the algorithm operates fairly and responsibly, given the general risk that the technology can be misused [58], [55]. Additionally, the credibility of sources daunts the LLMs family, as the models do not trace the provenance of the content they generate, creating uncertainty around source reliability [72].

The reliance on proprietary datasets such as those used in OpenAI poses risks to privacy and data security. In some instances, LLMs may produce data similar to those from training data without attribution, causing ripples in creative and academic spheres as it raises concerns about plagiarism and intellectual property [73], [32]. Financial LLMs, for example, face unique challenges in ensuring data privacy while utilizing proprietary data without breaches [13]. Moreover, the automation of mundane tasks by LLMs could impact employment, displacing roles traditionally performed by humans.

Addressing these social issues calls for the development of new ethical standards within the generative AI sphere and the implementation of measures to reduce bias and embrace transparency in Gen AI operations [74]. Techniques like RAG are increasingly adopted in financial domains to address these privacy and trust challenges. By tackling these ethical and social issues, stakeholders can work to ensure the responsible use of LLMs [73].

In conclusion, LLMs are capable of extending new opportunities for innovative development, but it is necessary to resolve the associated challenges. In this way, focusing on the improvement of the technical characteristics of the LLM systems and the operational issues -such as interpretability, efficiency, integration, and scalability, ensures that the LLM technologies are applied appropriately and responsibly [62]. Emergent and evolving challenges and broader concerns necessitate adaptive strategies involving regulatory frameworks, ethical guidelines, and technical solutions like differential privacy and federated learning [16]. Such approaches assist in developing trustworthy and accurate LLM systems that will be useful in generating reliable outcomes in various applications [75].

4. Solutions to Address LLM Challenges

As large language models (LLMs) continue to face significant challenges in real-world applications, addressing these issues is essential for ensuring their responsible and effective deployment.Key challenges such as interpretability, computational resource demands, and data quality require targeted solutions. Enhanced data training methods are crucial to enhance the accuracy and relevance of LLM outputs, by ensuring these models are trained on diverse, high-quality datasets. To tackle the challenge of computational inefficiency, the development of efficient algorithms and advanced hardware is necessary, enabling LLMs to operate at scale without excessive resource consumption. Explainable AI (XAI) techniques will address concerns about model interpretability, helping users understand the reasoning behind LLM outputs and increasing trust in these systems. Additionally, addressing privacy, data security, and ethical concerns requires the integration of regulatory frameworks, ethical guidelines, and technical solutions like differential privacy and federated learning. By focusing on these solutions, the challenges surrounding LLMs can be mitigated, ensuring that these technologies can generate reliable, ethical, and efficient outcomes across diverse sectors.

4.0.1. Enhanced Data Training Techniques

Research stresses that high-quality training data is the fundamental component of robust and effective LLMs because extensive text datasets teach LLM patterns, trends, and other insights. However, incomplete, biased, or out-of-date data will be passed over and retained thus compromising the resulting LLM [63], [76]. Domain-specific fine-tuning improves LLMs by training them on specialized datasets catered to specific industries, such as banking, biomedicine, and medical education [13], [77], [78]. In finance, fine-tuning can be done with news articles, market data, and company financials to enhance their ability to perform financial forecasting, risk analysis and investment analysis [13]. This increases LLM proficiency with domain-specific tasks by acquainting them with pertinent terminology and information. Data cleaning and filtering address problems with noisy or biased data, guaranteeing high-quality inputs that prevent models from learning undesirable patterns. Likewise, data augmentation creates new data to expand training datasets when available data is scarce, improving the model's robustness [7].

Integrating several data sources, including text, code, and transcripts of conversations, enhances an LLM's comprehension across different knowledge areas [24], [79]. For example, an LLM integrated with data from GitHub, text from Wikipedia and chats from chatbots can perform a wide range of tasks including code generation, text summary and dialogue. However, more research is needed to refine methods for integrating these sources effectively.

Scalability of data curation remains challenging, requiring strong infrastructure and efficient data handling to support storage, indexing, updating, and retrieval of large datasets [68]. As such, robust infrastructure is needed to handle the storage, processing, and analysis of vast data, while efficient handling of data is a prerequisite for indexing, updating and retrieval of heterogeneous data. Given that quality datasets are necessary for the best possible LLM performance, measuring and ensuring data quality is also critical. To fully utilize the promise of these models, research and innovation in scalable data curation and data training are required, a continual action that is critical for enhancing the accuracy, reliability, and fairness of LLMs.

4.0.2. Optimizing Algorithms and Hardware for Efficiency

To overcome computational cost challenges, optimized algorithms and hardware should be implemented for improved performance and efficiency.

Research highlights large computational demands placed on Large Language Models (LLMs) during deployment and training, and suggests ways to increase effectiveness. By selectively activating specific parts of the network to optimize resources, techniques such as mixture-of-experts explained (MoE) models can improve performance while decreasing complexity. Distributed computing Frameworks can be adopted for distributing the processing load among several machines and accelerating training [80], [66]. More so, deep learning operations have been accelerated by specialized hardware, such as GPUs, and TPUs and further innovation on the hardware will keep computing costs down. Originally intended for rendering graphics in video games and visual effects industries, their parallel processing efficiently, has made GPUs extremely effective in deep learning applications. On the other hand, TPUs which are tailored specifically for deep learning applications, offer an even greater level of computational power and efficiency than GPUs, providing a significant advantage in performance for such tasks [35], [13]. Additionally, model compression techniques—such as knowledge distillation and model pruning—enhance deployability on devices with limited resources without compromising performance [68]. Pruning eliminates crucial connections or neurons within the model, while quantization, reduces the precision of numerical values, and knowledge distillation, to replicate the performance of a larger ”teacher” model in a smaller ’student’ model [32]. Similarly, Parameter-Efficient Fine-tuning (PEFT) like LOoRA reduce processing needs by focusing on fine-tuning a limited set of parameters [21], thus reducing computational overload. This strategy proves beneficial in financial models to improve effectiveness and flexibility for financial functions where tasks necessitate minor adjustments to specific model parameters rather than a complete overhaul [81]. PEFT can boost the efficiency and adaptability of retrieval models in recommendation systems that cater to changing user behaviours [32] Using distributed computing systems improves productivity by facilitating model training and inference on numerous devices simultaneously. Expanding the workload among multiple sources can greatly diminish the time needed for training and inference processes. This enables the creation and implementation of extensive and intricate models. Examples of frameworks leveraging distributed systems include DenseX, EAR, UPRISE, RAST, Self-MEM, FLARE, Filter-rerank, R-GQA, LLM-R, LM-Indexer, and BEQUE, each optimizing efficiency in large-scale deep learning [79], [52] [79], [52]. Finally, developing expansive hardware infrastructure, cloud technology offers scalable, on-demand computing capabilities that enable LLM training and implementation at a reasonable cost.

4.0.3. Explainable AI (XAI) for Transparency and Trust

The key obstacles to making LLM explicable include limited context window, implicit bias, lack of transparency, model complexity and scale, data complexity, and assessment complexity. Despite this, there is considerable research interest in this area to unravel LLMs explainability.

Models have a restricted limited context windows, as a result, LLM limits the input and output of text LLM generates. This constraint hinders the explainability of the process, particularly in summarization tasks, where LLMS struggle due to limited text access [14], [24]. Implicit biases from training data can also surface in outputs, making it hard to trace why a model generates biased responses. The bias can be subtle and hard to decipher. For example, gender bias can be exhibited when a nuanced news article is used as part of a training dataset. The internal logic of LLMs is also often opaque, with their complex neural networks hindering transparency and trust in their decisions. As models grow larger, with billions of parameters, explaining their behaviour becomes increasingly challenging. The diversity and scale of training data make it hard to identify specific influences on outputs, and the lack of standardized metrics complicates efforts to assess and compare explainability across different LLMs [21].

There are several methods or approaches to improve LLM explainability. Engineering prompts elicit more explicable responses from LLMs through the use of strategies such as interaction and iteration, formatting with an example, explicit instruction, system-specific instruction, and control tokens. An illustration of this might be an explainability prompt like "Explain the concept of LLM as if I am 10 years old" [82], [76]). Model interpretation tools like benchmark tests and tools like RGB, RECALL, and CRUD [83] [82] and reference tools like model agents like AutoGPT, Toolformer, and Graph-Toolformer where LLMs employ active judgment in their operations can be used to gain insight into the decision-making process of LLMs [77], [84], [82]) [76]. Based on the particular requirements of each task, meta-reasoning prompting (MRP) assists LLMs in dynamically choosing and utilizing various reasoning strategies, potentially increasing the transparency of their reasoning process [79].

5. Integrating LLMs with Knowledge Bases

This section explores various techniques for integrating Large Language Models (LLMs) with knowledge bases, highlighting the potential solutions these integrations offer. By examining the methods used to combine LLMs with structured knowledge, we discuss how this synergy enhances the models' ability to provide more accurate, contextually relevant, and data-driven outputs. The integration strategies also open doors to improved reasoning capabilities, allowing LLMs to access and leverage external knowledge to address domain-specific queries and tasks more effective.

5.1. Integration Techniques for Enhancing LLMs with Structured Knowledge

Techniques for integrating LLMs with knowledge bases aim to enhance the models' performance by grounding their outputs in structured information. Key methods include using knowledge bases (KBs), which provide structured data for LLMs to query, and knowledge graphs, which organize information in interconnected nodes and relationships to improve context understanding. Additionally, Retrieval-Augmented Generation (RAG) combines document retrieval and generation, enabling LLMs to access relevant data from external sources and generate more accurate, up-to-date responses. These integration techniques enhance LLMs' ability to produce reliable, context-aware outputs.

5.1.1. Basic Knowledge Bases

Basic Knowledge Bases (KBs) are structured repositories that store facts in a relational format, providing an organized way to query and retrieve predefined facts, relationships, rules, and context, offering high precision when factual data is required.

In their inherent nature, Knowledge Bases help improve factual accuracy as they store verified information. Integrating LLM with the Knowledge base enables the models to cross-reference outputs with factual data stored in KBs thereby reducing hallucination and improving reliability. LLMs that access Wikidata, an open data knowledge base storing structured data on various domains (about people, places, and events), can generate factual outputs such as birthdate or scientific facts [24]. Studies show that Wikidata integration with Knowledge Enhanced PLMs promotes entity-aware training, through entity linking to Wikipedia, and joint training based on WiKidata's Knowledge Graph [24]. Several notable models, including K-BERT, KgPLM, FaE, JAKET, LUKE, WKLM, and CoLAKE, utilize Wikidata as a knowledge source in various ways including knowledge injection, knowledge-guided attention, and Knowledge Graph embedding, which improves factual accuracy, reasoning and enhanced interpretability and explainability [24]. Specialized knowledge bases like UMLS (Unified Medical Language System) for medical terminology or Freebase for general knowledge, provide LLMs with domain-specific data, enhancing their performance on tasks requiring specialized expertise [24]. For instance, LLMs integrated with the UMLS can retrieve up-to-date information on drug interactions or disease symptoms, which would significantly improve the model's ability to provide medical diagnoses or treatment recommendations [85]. However, most KBs are static and riddled with limited reasoning capacities, often requiring manual upgrades. This makes them unsuitable for real-time or fast-changing information and limits their utility in fields such as current events or rapidly evolving research domains. While KBs store factual data, they do not inherently capture complex relationships or provide the inference capabilities of more advanced systems like Knowledge Graphs. For applications requiring real-time updates or complex inference, knowledge bases alone are insufficient, and more advanced systems are needed.

5.1.2. Knowledge Graphs

Knowledge Graphs (KGs) provide a structured approach to representing knowledge, enabling LLMs to overcome inherent limitations in understanding and reasoning about the connections between real-world entities and concepts. This integration is valuable in complex reasoning tasks like multi-hop question answering and semantic disambiguation where connection between concepts is valuable. Additionally, the KGs allow LLMs access to up-to-date knowledge without the need for retraining, making them a powerful tool to keep LLMs relevant and current [86].

Knowledge Graphs (KGs) are integrated into Large Language Models (LLMs) using several advanced methods designed to enhance their reasoning capabilities: embedding, Graph Neural Networks, prompting and semi-structured chain of thought. These techniques vary in approaches to processing and utilizing structured knowledge.

Knowledge Graph Embedding transforms entities and relationships into numerical vectors for efficient processing, while Graph Neural Networks capture complex relationships within the KG, enhancing multi-hop reasoning abilities [87]. Knowledge Graph prompting injects relevant KG data directly into the LLM's input, guiding reasoning and improving accuracy [8]. Semi-structured chain of thought combines both structured KG data and unstructured text to generate reasoning chains, leveraging all available knowledge for more complex reasoning tasks [76], [67], [87]. Each method provides a distinct way to enhance the LLM's reasoning and response capabilities. Key techniques for integration include KG-enhanced LLMs, LLM-augmented KGs and synergized LLMs and KGs. In KG-Enhanced LLMs, KGs are used to augment the training or inference process of LLMs, supplying them with structured information to support reasoning and understanding. Whereas in LLM-augmented KGs, LLMs assist in various KG-related tasks, such as embedding, completion, construction, and question answering [88]. Lastly, in synergized LLMs and KGs, the unified framework combines the strengths of both KGs and LLMs, creating a model that leverages KGs for structured knowledge and LLMs for language understanding [33].

Notable advantages of Knowledge Graphs over knowledge bases include increased reasoning, inference, and improved interpretability. KGs store information in a network of nodes (entities)and edges (relationships) and emphasize relationships between entities, making them more flexible and dynamic than knowledge bases with unstructured text. Unlike KBs, which store isolated facts, integrated LLMs can reason about complex relationships and infer new information based on existing graphs. As such Knowledge Graphs such as Google Knowledge Graph, ConceptNet and DBPedia, LLMs can answer questions by traversing relationships between entities. Specifically, ConceptNet supports common-sense reasoning, DBPedia provides structured data for general knowledge queries, and Google Knowledge Graph aids in answering factual questions by linking related entities[76], [24].

Research demonstrates several ways in which Knowledge Graphs (KGs) can be used to enhance the capabilities of large language models (LLMs). KGs naturally support explainability by visualizing the graph of entities and relationships. KGs can enhance explainability by showing how the model arrived at a specific conclusion. A KG can map the relationships between legal precedents, statutes, and case law, providing an explainable pathway that LLMs follow to generate legal opinions [86, 10, 7].

KGs can be updated with new information, making it ideal for real-time applications (like stock exchanges, active medical diagnosis, and navigation) where information must be frequently refreshed. Applicable frameworks like MedGraph, WeKnow-RAG and Think on Graph enable LLMs can retrieve real-time knowledge about evolving topics, such as the latest news developments or ongoing live events [81], [56], [58], [51].

KGs offer a structured and interconnected way to represent knowledge, which is beneficial for LLMs in understanding relationships and making inferences. Unlike relational databases or NoSQL solutions, KGs use graph structure to explicitly link information, making it easier for LLMs to understand relationships and make inferences [87], [35], [76]. This interconnected structure facilitates the use of specialized query languages (like SPARQL and Cypher) that are well-suited for navigating complex relationships between entities [86], [65].

KGs enhance factual accuracy and knowledge probing by providing LLMs with a factual grounding that might be missing or weakly represented in their training data. This is especially helpful for tasks such as answering questions, where providing accurate information is crucial [59]. For example, the LAMA (Language Model Analysis) dataset is designed to test how well LLMs have internalized factual knowledge and by incorporating knowledge from KGs, LLMs can perform better on such knowledge probing tasks [59]. KGs that encode commonsense knowledge, like ConceptNet and ATOMIC, are invaluable for LLMs, they boost common-sense reasoning and natural language generation [59], [24].

While KGs offer significant advantages, the tradeoffs include incompleteness, complexity, scalability, and performance. KGs are often plagued with missing links and relationships between entities. LLMs can be trained to predict these missing links, effectively completing and refining the KG, thus benefiting both LLMs, by providing them with a more complete knowledge base, and KGs, by improving their accuracy and coverage [76]. Creating and maintaining large-scale KGs is a resource-intensive process that requires careful design and data curation [68]. Scaling across broad domains or diverse industries can be difficult and costly. Although KGs improve reasoning capabilities, integration with LLMs may introduce latency and reduce overall system performance. Hence, though Knowledge Graphs may offer a more advanced solution than KBs by providing a dynamic, relationship-focused structure that enhances reasoning and interpretability, the complexity and scalability challenges make KGs less feasible for broad, real-time applications unless coupled with significant computational resources. Therefore, KG is best suited for high-stakes domain-specific tasks where reasoning and relationships between entities are critical, such as pharmaceutical drug discovery and research, Failure Mode and Effects Analysis (FMEA), financial fraud detection, and support threat intelligence in cyber-security [87], [86], [89].

5.1.3. Retrieval Augmented Generation

Retrieval-augmented generation (RAG) is a hybrid approach that integrates LLMs with a retrieval mechanism to fetch relevant information from external documents or knowledge in real-time during the text generation process. This technique allows LLMs to dynamically access vast external corpora (like knowledge graphs, databases or search engines) to retrieve the most relevant information [63], [22], [79], [77]. RAG enables language models to retrieve factual information and generate more accurate and contextually aware outputs, especially in cases where a language model's training data may be outdated or incomplete [71]. This approach provides the LLM with an "external memory" to supplement its internal knowledge base, thus enhancing LLM quality and accuracy.

RAG comprise of retrievers, generators, and knowledge bases [22]. The retriever dynamically fetches relevant information from external corpora, the generator uses this retrieved information to generate a response, and the knowledge base is a collection of text such as scientific articles, news articles, or Knowledge Graphs [90], [29], [68]. This architecture allows the LLM to access up-to-date knowledge beyond its static training data, alleviating hallucination issues by grounding the generated response in factual data.

RAG enhances retrieval and generation through several key techniques which may include document chunking, embedding models, retrieval techniques, querying, knowledge graph integration, iterative retrieval and generation and self-reflection. Chunking breaks down large text into manageable sizes using a mix of static and semantic methods to maintain context [90], [53]. Helps embedding models to represent information and queries in a way that maintains semantic meaning, improving retrieval accuracy [27]. RAG employs retrieval methods such as dense retrieval, which uses vector representations (cosine similarity) for semantic matching, and sparse encoding for keyword matches between queries and documents; with hybrid approaches combining the strengths of both [90], [68]. Dense Passage Retrieval (DPR) technique utilizes dense embeddings to match semantically relevant document chunks to queries and the combination of DPR with traditional sparse retrieval (i.e., BM25) has been shown to further enhance retrieval precision in complex or high-precision tasks [81], [57].

Query expansion using techniques like Query2doc involves expanding the original query to include additional terms, increasing the effectiveness of retrieval [91]. Furthermore, Knowledge Graphs can be integrated into RAG systems for structured reasoning, allowing more accurate and meaningful results [86], [51]. Iterative retrieval and generation involve repeating these processes to refine outputs, guided by self-evaluation mechanisms that assess the adequacy of retrieved information [27], [15]. Self-reflection features in advanced systems like Self-RAG allow the model to assess the relevance and accuracy of the information it retrieves and generates, improving overall output quality and contributing to better explainability [77].

Retrieval-augmented generation (RAG) excels in complex, knowledge-driven tasks by linking large language models (LLMs) with real-time or domain-specific retrieval, significantly enhancing the scope and accuracy of AI applications. RAG capabilities are experienced in varied settings. In cross-domain applicability, RAG is used for question answering, dialogue generation, summarisation, fact-checking or verification, information extraction, and reasoning [57], [27], [86], [35], [54].

In question-answering (QA) systems, Naive RAG improves response accuracy, especially for multi-hop queries or long-form answers where standalone LLMs might lack context. This is crucial for fact-checking, where retrieving authoritative sources ensures the reliability of generated content [18], [29]. RAG's role in dialogue systems is equally transformative, as it enriches real-time responses, especially in task-oriented conversations like customer support [92]. Modular RAG capabilities such as text summarisation are invaluable for condensing large documents into coherent, concise summaries, boosting content generation efficiency [93]. Additionally, RAG's ability to retrieve and integrate specialized knowledge makes it indispensable in fields such as law and medicine, where precision is critical [94].

Furthermore, RAG strengthens recommendation systems by providing personalized suggestions based on user data, and in code search and generation, it supports developers by retrieving relevant code snippets to address complex technical challenges. Overall, RAG's versatility across diverse applications highlights its vital role in enhancing LLMs' performance, making them more accurate and contextually aware in real-world scenarios.

Overall, RAG systems are ideal for real-time data retrieval of unstructured data in environments with dynamic data volumes. For example, OpenAI RAG offers LLM the ability to query data and retrieve and utilize vast external data in its response, which improves the relevance and factual accuracy of generated text and real-time knowledge access [29]. Therefore, a knowledge graph is best suited for high-stakes, domain-specific tasks where reasoning and relationships between entities are critical. RAG renders domain flexibility through retrieving domain-specific documents from external sources making it easier for LLMs to answer specialized queries without needing extensive domain-specific training. Table 2 summarized techniques for integrating Large Language Models (LLMs) with knowledge bases (KBs), Knowledge Graphs (KGs), and Retrieval-Augmented Generation (RAG).

Technique	Description	Typical Methods	Advantages	Challenges
Basic Knowledge Bases (KBs)	Structured repositories storing predefined facts, relationships, and rules.	Knowledge Injection, Knowledge-guided Attention, Knowledge Graph Embedding	High factual accuracy, reduces hallucinations, domain-specific knowledge (e.g., UMLS, Wikidata)	Static, limited reasoning, slow updates, lacks complex relationship handling.
Knowledge Graphs (KGs)	Represent knowledge through interconnected nodes (entities) and edges (relationships).	Graph Neural Networks, Knowledge Graph Embedding, KG Prompting, Semi-structured Chain of Thought	Enhanced reasoning, dynamic updates, supports multi-hop reasoning, explainability, improves interpretability	Missing links, scalability issues, complexity, resource-intensive maintenance.
Retrieval-Augmented Generation (RAG)	Combines document retrieval with generation, allowing LLMs to access external data in real-time.	Document Chunking, Embedding Models, Dense/Sparse Retrieval, Knowledge Graph Integration, Self-Reflection	Real-time access to up-to-date information, dynamic context, alleviates hallucination, accurate outputs.	Requires efficient retrieval systems, latency, computational complexity.

Table 2: Integration Techniques for Enhancing LLMs with Structured Knowledge

5.2. Advanced Techniques and Hybrid Approaches

The integration of Retrieval-Augmented Generation (RAG) with large language models (LLMs) has significantly enhanced their ability to handle complex tasks by incorporating external knowledge sources. However, to truly optimize LLM performance, advanced techniques and hybrid approaches that go beyond traditional RAG systems are essential. This section explores such advanced methods, focusing on the integration of Knowledge Graphs (KGs) and Knowledge Bases (KBs) with LLMs. By combining these systems, LLMs can leverage structured and hierarchical data, enabling them to reason more effectively and retrieve relevant information more accurately. The techniques discussed here span across various approaches, including specialized retrieval mechanisms, multi-modal data integration, cognitive reasoning methods, and domain-specific optimizations. The table 3 highlights the key applications, benefits, and frameworks for each technique, showcasing how they contribute to more effective and efficient LLM performance in diverse domains.

Technique	Description	Key Applications	Benefits	Key Models/Frameworks
Enhanced Retrieval Techniques	Focus on improving retrieval accuracy and handling complex data, integrating multi-modal and hierarchical knowledge.	Enterprise settings, healthcare, legal reasoning, diagnostics	Improved retrieval accuracy, better handling of complex data, enhanced information security and transparency	T-RAG, REALM, TRACE
Knowledge Graph Integration	Integrates structured knowledge from Knowledge Graphs (KGs) for domain-specific question answering and reasoning.	Healthcare, biomedical question-answering.	Increased accuracy in question-answering, reduced noise, focused retrieval of relevant information	Triple-aware reasoning, KG-RAG, CRAG.
AdaptiveRetrieval and Evaluation	Evaluates and adapts retrieval processes using fine-tuning or reinforcement learning techniques.	Optimal retrieval, fine-tuning, reinforcement learning	More effective retrieval strategies, dynamic adaptation for optimal results, improved model efficiency.	CRAG, SLM, fine-tuning techniques.
Cognitive Reasoning and Decision-Making	Integrates cognitive strategies like introspection, evaluation, and dynamic response generation for improved reasoning and decision-making.	Multi-hop question-answering, interactive tasks.	Enhanced reasoning over conflicting knowledge, better decision-making in dynamic environments.	MetaRAG, ReAct.

Table 3: Advanced Techniques and Hybrid approaches for Enhancing integration LLLMs-Knowledge Bases.

5.2.1. Enhanced Retrieval Techniques

Focuses on improving retrieval accuracy, data integration, and handling complex data structures like multi-modal and hierarchical knowledge. Typical examples like T-RAG proposes a tree-based structure framework that categorizes and retrieves hierarchical organizational knowledge. The T-RAG framework not only improves retrieval accuracy in enterprise settings but also maintains the necessary data security and transparency required for sensitive applications [95]. This is effective when standard RAG encounters challenges when trying to integrate interconnected facts in scenarios where reasoning is needed to answer questions [96]. Moreover, the REALM framework further enhances enterprise use by integrating multi-modal data (e.g., clinical notes, time-series data) into a cohesive RAG pipeline. In the healthcare domain, REALM retrieves relevant medical knowledge from external sources like PrimeKG, ensuring that hallucinations are eliminated through entity matching and validation [57]. TRACE, a graph-based retrieval model creates multi-hop reasoning chains by transforming retrieved documents into Knowledge Graphs, allowing the LLM to follow complex reasoning paths across multiple documents. This method ensures that responses are anchored on logically connected facts, essential for tasks such as legal reasoning and medical diagnostics [15], [96].

5.2.2. Knowledge Graph Integration

Techniques that integrate structured knowledge from Knowledge Graphs (KGs) to support domain-specific question answering and reasoning. Specialized models include triple-aware reasoning, KG-RAG, and CRAG. With targeted refined retrieval techniques each of these models is designed to enhance performance in healthcare. Triple-aware reasoning integrates structured knowledge from KGs into the RAG process by filtering and selecting relevant triples for complex question-answering tasks. This filtering mechanism reduces noise and ensures that LLMs focus on pertinent information for user questions, improving both accuracy and reasoning capabilities [55]. Another hybrid model, KG-RAG, is optimized for domain-specific question answering. By leveraging minimal graph schema for context extraction and reducing token consumption, KG-RAG achieves a significant performance boost on biomedical multiple-choice questions, further demonstrating its efficacy [51].

5.2.3. Adaptive Retrieval and Evaluation

Models designed to evaluate the quality of retrieved documents and adapt the retrieval process using techniques like fine-tuning or reinforcement learning. For instance, CRAG evaluates the quality of retrieved documents for a query and triggers different knowledge retrieval actions based on confidence levels. CRAG achieves this by training a lightweight retrieval evaluator [35], [81]. Moreover, integrating RAG with reinforcement learning allows models to learn optimal retrieval and generation strategies through trial and error, potentially leading to more effective and efficient RAG systems. Ongoing research focuses on optimal integration of RAG and fine-tuning with a target of harnessing both parameterized and non-parameterized advantages, whether sequentially, alternately, or through end-to-end joint training [35], [81]. An additional focus area is leveraging RAG in specialized Small Language Models (SLM) and fine-tuning them based on the results of RAG systems [35], [81].

5.2.4. Cognitive Reasoning and Decision-Making

Models that integrate cognitive strategies to improve reasoning, decision-making, and dynamic response generation. Significant advancements in this area are MetaRAG and ReAct. MetaRAG integrates retrieval-augmented generation with metacognitive capabilities to enhance multi-hop question-answering tasks. It applies metacognitive principles—monitoring, evaluating, and planning—to allow LLMs to self-assess their reasoning paths and adjust dynamically based on identified errors or knowledge gaps [22]. The advanced approach outperforms traditional RAG and reflection models by leveraging metacognitive regulation, resulting in improved accuracy, particularly in scenarios requiring complex reasoning over conflicting knowledge [79]. MetaRAG's use of introspection and adaptive reasoning aligns with trends in incorporating cognitive psychology principles into AI systems to enhance performance. This suggests a broader move towards making LLMs more "intelligent" by mimicking human cognitive processes.

ReAct interleaves reasoning steps (thoughts) with task-specific actions (acting) to improve interpretability and decision-making in LLMs. This framework dynamically retrieves and utilizes external information, combining internal model reasoning with real-world grounding [97]. ReAct achieves significant improvements over baselines like chain-of-thought and action-only methods by integrating reasoning and acting, reaching state-of-the-art results in benchmarks like question answering (HotpotQA), fact verification and textual games (ALFWorld) [97]. Like MetaRAG, ReAct enhances reasoning by integrating dynamic information retrieval. However, it focuses more on interactive decision-making and real-world navigation tasks, highlighting the potential for LLMs to operate in more complex, interactive environments.

5.2.5. Domain-Specific Optimization

Techniques optimized for specific domains (e.g., healthcare) that aim to improve reliability, accuracy, and safety in specialized contexts. For some specific domains, such as in Biomedicine, Self-BioRAG combines on-demand retrieval of domain-specific documents and self-reflection critique capabilities to improve the quality and reliability of generated content and the efficiency of the model. The core concept of Self-BioRAG lies in its adaptive retrieval approach, where the model selectively determines when and what information to retrieve based on the context of the question ensuring that generated responses are both relevant and evidence-based [77]. In comparison to proprietary LLMs (like GPT-4, Med-PaLM) and open LLMs with RAG capabilities, Self-BioRAG outpaces the models in most biomedical benchmark datasets demonstrating the reliability of domain-specific retrieval and reflective techniques leading to more accurate and reliable medical reasoning. Another example like MedGraphRAG contributes to the domain by using hierarchical graphs to align clinical questions and retrieve medical knowledge, thus enhancing structured retrieval mechanisms. This enhances safety by ensuring that LLM-generated medical advice is evidence-based, reducing the potential for harmful hallucinations [53].

5.3. Leveraging on Prompt Augmentation and Engineering for LLM-Knowledge Base Integration

This section introduces foundational prompt engineering techniques designed to enhance the retrieval and generation capabilities of large language models (LLMs). By strategically adapting prompts, these techniques guide models to produce more accurate, contextually relevant, and domain-specific outputs. This foundational approach sets the stage for the subsequent exploration of advanced integration methods, focusing on optimizing the interaction between LLMs and knowledge bases for improved performance across various applications.

Chain-of-thought (CoT) prompting has emerged as a powerful technique for guiding LLMs in complex reasoning tasks by encouraging them to embrace step-by-step reasoning pathways, before formulating a response. CoT exhibits human-like patterns in solving multi-step problems such as arithmetic problems, commonsense reasoning, and symbolic tasks. [82], [63], [79], [58].

Building upon CoT, several enhancements, including Buffer of Thoughts (BoT), Strategic Chain-of-Thought (SCoT), and Graph of Thought have been developed. BoT introduces a meta-buffer that stores high-level thought templates distilled from previous problem-solving processes. These templates can be retrieved and instantiated for new tasks, enabling more efficient and accurate reasoning. BoT outperforms standard multi-query prompting methods by achieving superior performance in tasks requiring multi-step reasoning, such as mathematical problem-solving and logical reasoning, while reducing computational costs. The use of a meta-buffer allows for the creation of generalizable and reusable reasoning structures, optimizing both reasoning accuracy and efficiency. This approach mirrors human cognitive processes of using ”mental templates” or heuristics to solve problems [84]. Strategic Chain-of-Thought (SCoT) enhances CoT by incorporating strategy elicitation to guide reasoning processes. It employs a two-stage process within a single prompt, enhancing the stability and consistency of reasoning paths [91]. SCoT outperforms existing methods like Self-consistency and Buffer of Thoughts by using strategic knowledge to generate accurate single-query reasoning paths, demonstrating the importance of strategic planning in LLM reasoning. Both BoT and SCoT leverage cognitive-inspired methods to refine reasoning paths, demonstrating a trend towards incorporating human-like cognitive processes, such as strategy selection and problem decomposition, into LLMs.

Graph-of-Thoughts (GoT) technique extends CoT by representing the reasoning process as a graph, allowing for more complex and non-linear reasoning patterns. This approach enables the exploration of multiple reasoning paths and the evaluation of different options, potentially leading to more accurate and insightful solutions [64], [84], [69]. Other chains of thought techniques include Mindmap, IRCOT, Reasoning on Graphs, CoT with Consistency, Program Aided Language Model (PAL), Reason and Act, reflection and Tree of thought [87, 22], [38], [36], [40]

Advanced prompting augmented methods are critical for maximizing LLM performance across various tasks. This is exhibited in prompt augmentation systems (PAS), and Slim Proxy Language (SlimPLM) alongside Self-memory techniques. The Prompt Augmentation System (PAS) introduces a novel and data-efficient method for enhancing prompts. As a plug-and-play automatic prompt engineering system, generating complementary prompts to enhance LLM outputs. This method aligns with high data efficiency, achieving state-of-the-art results with minimal data. PAS significantly outperforms Automatic Prompt Engineering (APE) models, demonstrating the effectiveness of generating high-quality complementary prompts automatically. The emphasis on data efficiency and automatic augmentation aligns with broader trends in developing scalable and adaptable prompt engineering solutions for LLMs. It also underscores the importance of reducing the dependency on large datasets, making LLM applications more accessible and sustainable [69].

SlimPLM (Slim Proxy Language Model), is a smaller proxy language model that assesses the LLM's knowledge and determines whether retrieval is necessary, optimizing the use of external resources [98]. Self-memory prompting is an adaptive form of prompting where the LLM's own generated outputs are used to enrich subsequent prompts through self-feedback loops [54], hence improving the quality and consistency of responses. This prospective prompting framework solves the limitations of using the internal memory of traditional RAG by incorporating a retrieval-augmented generator and a memory selector. The generator uses both the input text and retrieved memory to generate output, while the memory selector refines this output to create an unbounded memory pool that is iteratively used for subsequent generations with quality improvement in tasks such as Neural Machine Translation, abstractive text summarization, and dialogue generation [54].

In summary, prompt engineering techniques play a crucial role in enhancing the accuracy, efficiency, and reasoning capabilities of large language models (LLMs). By refining how prompts are structured, these techniques optimize retrieval and generation processes, enabling LLMs to perform more effectively across diverse tasks. They serve as a foundational element for developing robust, context-aware AI applications, laying the groundwork for more sophisticated and adaptable models in real-world scenarios.

5.4. LLM capabilities within Integrated Environment

This section focuses on how integrated large language models (LLMs) effectively address the challenges encountered by traditional LLMs, as discussed in Chapter 2. By leveraging advanced integration techniques, such as the incorporation of knowledge graphs, knowledge bases, and specialized retrieval mechanisms, these integrated models overcome limitations in accuracy, scalability, and interpretability, providing more robust and contextually aware solutions.

5.4.1. Enhancing Interpretability in Integrated LLMs

Retrieval-augmented generation (RAG) serves as a bridge between the generation and retrieval processes, allowing users to directly trace the pathways of information used by the LLM. By integrating retrieval-based methods, RAG systems give LLMs real-time access to external, up-to-date data sources, such as large document databases, the web, or specialized knowledge bases [29], [35]. This provides anchoring for the model's generated responses, enhancing interpretability and transparency by allowing users to verify the source of the information. It also reduces the risk of hallucinations, a common issue with LLMs that rely on static training data [63].

RAG is especially effective for tasks that require current or specific knowledge. For instance, in domains such as biomedicine or law, where verifying conclusions against well-established knowledge is essential, RAG ensures that the LLM's responses are grounded in authoritative, real-world data [77]. The Chronicles of RAG has demonstrated how effective external knowledge integration can be in improving trust and transparency by allowing users to track the origin of the generated content [90].

Knowledge Graphs (KGs) provide a structured representation of relationships between entities. By allowing LLMs to access well-organized, structured information, KGs enable LLMs to reason through complex tasks in a logical and traceable manner [35]. This enhances model transparency, as users can follow the reasoning process more clearly. For instance, systems like TRACE use KGs to map logical connections between retrieved evidence, enabling multi-hop reasoning (i.e., connecting multiple sources to answer complex queries [15]. The integration of KGs into LLMs thus, strengthens the interpretability and traceability of their outputs.

Advanced prompting methods, such as Chain-of-Thought (CoT) prompting, further improve interpretability by breaking down complex questions into smaller, more manageable sub-tasks. This step-by-step reasoning enables LLMs to adopt intermediate reasoning that can be traced and verified, making the decision-making process clearer to the user [58], [92].

Systems like TRACE construct reasoning chains that decompose a task into sub-steps, guiding the LLM through each sub-task by identifying key pieces of evidence and logically connecting them. CoT is particularly useful for tasks that require logical reasoning or multi-step calculations, such as mathematical problem-solving, multi-hop question answering, or decision-making tasks [35].

When combined, RAG, Knowledge Graphs, and Chain-of-Thought prompting create a powerful framework for enhancing interpretability in large language models (LLMs). This integrated approach grounds LLM outputs in verifiable real-world sources reduces hallucinations and offers a structured reasoning framework. Together, these methods increase traceability and make the model's decision-making processes more understandable and easier to validate. This combined framework significantly improves both the reliability and interpretability of LLMs.

5.4.2. Managing Computational Costs in Integrated LLMs

Techniques like Retrieval-Augmented Generation (RAG) aim to mitigate these challenges by offloading knowledge retrieval to external databases, reducing the need for models to store all knowledge internally. This external retrieval system allows for more efficient use of resources, both in memory and processing and enables the model to access up-to-date information dynamically. For instance, an exploration of RAG demonstrates how retrieving external documents allows the LLM to reduce its internal storage needs, thus lowering computational demands during both training and inference [90]. This flexibility enhances the scalability of the models, making them more adaptable to real-time updates without requiring retraining, which is resource-intensive [90].

Further improvements to RAG's computational efficiency have been achieved through optimized retrieval methods, such as graph-based re-ranking (G-RAG). G-RAG refines the retrieval process by prioritizing the most relevant documents through a semantic graph that links related documents. This targeted retrieval not only reduces the volume of irrelevant information processed but also increases the precision of the responses, thus reducing overall computational costs [90]. This re-ranking technique significantly minimizes the workload on LLMs while maintaining high response quality. However, while re-ranking shows promise, there is still room for improvement, particularly in reducing the time overhead associated with complex knowledge graph constructions.

Another critical technique contributing to computational efficiency is few-shot prompting, which allows LLMs to adapt to new tasks without requiring full retraining. Few-shot prompting operates by providing a few examples to the model during inference, enabling it to generalize across similar tasks [80], [29]. This method, as exemplified in RAG, reduces the need for continuous updates or domain-specific retraining, a process that traditionally demands significant computational resources. The ability of models to adapt dynamically via few-shot learning echoes broader trends in transfer learning, where models pre-trained on large datasets are fine-tuned on smaller, task-specific data. Few-shot prompting, though efficient, may have limitations in complex domains where more extensive task-specific training might be necessary for optimal performance [54], [99].

While RAG, optimized retrieval methods, and few-shot prompting present robust solutions for computational efficiency, some challenges remain unaddressed. For example, RAG's reliance on external databases introduces potential delays in retrieval and integration, which may not always align with real-time processing needs. Furthermore, as noted in the critique of RAG, over-reliance on external documents can lead to instances where the model unnecessarily retrieves information it already possesses, contributing to inefficiency. Additionally, research suggests that even with optimization strategies, LLMs sometimes struggle with long or complex queries, such as those affected by the "Lost in the Middle" problem, where the model focuses too much on the start and end of a document while neglecting the middle sections [100], [58]

5.4.3. Solving Scalability and Reliability in Integrated LLMs

Scaling Large Language Models (LLMs) across various domains presents challenges in terms of handling diverse queries, and vast datasets, and ensuring relevant, accurate responses. A key solution to enhance scalability is to combine an external Retrieval-Augmented Generation (RAG) with Knowledge Graphs (KGs) [89], [22], [53]. This hybrid system leverages the dynamic knowledge retrieval capabilities of RAG with the structured reasoning of KGs, allowing LLMs to efficiently scale across different domains while maintaining factual accuracy. RAG enables real-time retrieval of external information, hence preventing the need for LLMs to store all knowledge internally. When integrated with KGs, which organize and structure domain-specific data such as medical or legal information, the system ensures precise reasoning and decision-making, especially in large specialized LLM [68], [56]. Similarly, in the *Graph-Based Retriever*approach discussed, a hybrid RAG-KG system enhances scalability by addressing the issue of information overload in biomedical literature [89], [56], [81], [101]. In the medical domain, repetitive and redundant data can overwhelm retrieval systems. By using KGs to down-sample over-represented topics and ensure balanced information retrieval, the hybrid RAG-KG system ensures that critical but less prominent information is not overshadowed, making it scalable across large, complex datasets. This hybrid approach broadens the applicability of LLMs to fields requiring precision, such as biomedicine and law, without sacrificing accuracy or efficiency [89].

RAG enhances reliability by grounding responses in external, verifiable sources, rather than relying solely on pre-trained model weights. This grounding ensures that LLM outputs are based on up-to-date, trustworthy information. Structured retrieval methods, particularly graph-based reranking (G-RAG), further improve reliability by optimizing which documents are retrieved [102]. Graph neural networks (GNNs) evaluate the relationships between retrieved documents, elevating the most contextually relevant ones, and reducing the chance of irrelevant or inaccurate information being used in the final output.

In tasks like open-domain question answering, where reasoning often requires drawing from multiple sources, G-RAG's structured retrieval process ensures the model has access to the most reliable documents. Research has demonstrated that by employing Abstract Meaning Representation (AMR) graphs to map semantic relationships between documents, G-RAG filters out less relevant data and prioritizes documents that offer more accurate answers. This significantly reduces the risk of hallucinations—situations where the model generates inaccurate or fictional information—making the system more dependable for open-domain tasks [102]. Lastly, leveraging smaller proxy models to identify knowledge gaps in the LLM, the system can selectively engage retrieval mechanisms, optimizing resource utilization [98].

5.4.4. Seamless Integration of LLMs with Existing Systems

Integrating Large Language Models (LLMs) into existing systems presents a variety of challenges, including data access, real-time updates, scalability, security, and interpretability. To address these, multiple advanced techniques have been developed beyond the common methods like Retrieval-Augmented Generation (RAG) and Knowledge Graphs (KGs).

RAG allows LLMs to dynamically retrieve relevant information from external sources during inference, bridging gaps in knowledge and providing access to both structured and unstructured data. LLMs are further enhanced with API integration and data pipelines, ensuring seamless real-time access to diverse data sources such as legacy systems, external APIs, and proprietary databases [65] [22]. This integration can be simplified through RESTful APIs or more complex solutions like GraphQL, which allows querying across multiple data endpoints efficiently.

In scenarios where keeping LLMs updated is critical, especially in dynamic fields like healthcare and finance, RAG retrieves the latest information at inference time, eliminating the need for constant retraining. Knowledge Graphs (KGs), which offer structured data and incremental updates, further allow LLMs to access evolving knowledge without undergoing full retraining cycles. This combination is particularly useful in specialized domains where new information is frequently introduced, such as in BIORAG for biomedical data.

Tool-integrated reasoning through search engines, calculators, and code interpreters can extend LLM capabilities, and enhance accuracy, efficiency and integration with specific tasks or domains [32]. Incorporating tools into the thinking process of LLMs can boost effectiveness in tasks that require calculations or specialized algorithms, hence enhancing their performance and lessening their workload. Models such as TORA use tool-integrated reasoning to tackle challenging problems by interweaving natural language reasoning with external tools, like computation libraries and symbolic solvers [48].

Additionally, adapters can be used to fine-tune LLMs for specific tasks without altering the core model parameters [92]. These lightweight neural modules can be plugged into the LLM and trained with domain-specific data, making them highly adaptable to different tasks and reducing computational costs. For example, the UP-RISE framework employs adapters to select the most appropriate prompts from a pool for a specific zero-shot task [35]. This flexibility is particularly useful for enterprise systems, where multiple domain-specific adaptations may be required.

Another promising method for improving scalability is multi-agent architectures, where distinct agents handle specific system tasks, such as data ingestion, knowledge retrieval, and response generation. This division of labour distributes the workload more efficiently and allows LLMs to integrate smoothly with complex systems [103].

To further enhance the system's ability to scale, model merging can combine the parameters of multiple pre-trained models into a unified model, integrating different task-specific capabilities into a single, more versatile LLM. Additionally, tool augmentation extends the LLM abilities by integrating it with external tools, APIs, and real-time data streams for specialized tasks like scheduling, booking, or domain-specific reasoning.

5.4.5. Adapting LLMs to Changing Knowledge Bases

One prominent issue is limited temporal knowledge, how models handle time-related information and reasoning. Despite their vast capacity, traditional LLMs are bound by the constraints of their training data. They do not automatically acquire new information after training, which can lead to outdated or irrelevant responses, particularly concerning recent events or discoveries. For example, an LLM trained before a significant geopolitical event or medical breakthrough may still produce content that reflects an outdated understanding. To address this, research explored methods like continuous learning and dynamic training. Continuous learning enables LLMs to incrementally adapt to new information, allowing them to incorporate evolving language and updated knowledge without needing to retrain from scratch [47]. This is particularly valuable in fields where timely and accurate information is crucial. Dynamic training similarly focuses on updating models continuously, ensuring that LLMs remain relevant over time as new research and discoveries emerge [104]. Another substantial challenge is the difficulty in updating knowledge. Traditionally, modifying an LLM's knowledge base requires retraining the entire model, a process that is both computationally intensive and costly [105]. This has spurred the development of techniques like Retrieval-Augmented Generation (RAG), fine-tuning, and incremental learning. RAG systems are particularly effective because they allow LLMs to retrieve external information during inference, circumventing the need for constant retraining [68], [95], [81].

Through retrieval, models can access up-to-date knowledge from databases, Knowledge Graphs, or other dynamic sources like the Internet. Sources advocate future research on several areas time-aware retrieval, forward-looking active retrieval augmented generation, real-time QA, and timeliness in Gen IR [35], [71], [104].

Time-aware retrieval potential incorporates temporal metadata like timestamps into the retrieval process of RAG to access the up-to-date data. FLARE anticipates future queries during text generation enabling more timely and relevant access [71]. Realtime QA evaluates the LLM's ability to handle recent events whilst GenIR advocates for research in real-time knowledge access and continual learning and editing to maintain knowledge currency [32]. Notable research is demonstrated with common sense reasoning explored reasoning about time, where LLMs are instructed to retrieve information given a specific time or events [84].

5.5. Case Studies: Practical Applications of LLM-Knowledge Base Integration

This section explores real-world case studies that demonstrate the practical applications of integrating Large Language Models (LLMs) with knowledge bases. The cases discussed include FinAgent, a multimodal foundation agent designed for financial trading; the Unified Medical Language System, which facilitates the integration of medical knowledge for improved healthcare outcomes; Codex LLMs, which enhance programming capabilities by connecting code generation with vast knowledge bases; and BloombergGPT, a specialized LLM built for financial data analysis and decision-making. These case studies highlight the diverse ways in which LLM-Knowledge Base integration is applied across various industries, showcasing the potential for enhanced accuracy, decision-making, and efficiency.

5.5.1. FinAgent: A Multimodal Foundation Agent for Financial Trading

A notable advancement in finance is the integration of multimodal data both text and visual to trading agents. FinAgent exemplifies this by leveraging multimodal LLMs and reinforcement learning to optimize buy, sell, and hold decisions for maximizing profits within defined risk constraints. The system incorporates market intelligence, tool augmentation, specialized prompt generation, dual-level reflection and diversified memory retrieval its trading adaptability and performance in stocks and crypto [42].

FinAgent gathers, processes, and analyzes diverse data sources, including news, prices, and financial reports like a RAG-like pipeline, by leveraging on market intelligence capabilities, hence, providing it a comprehensive understanding of current market conditions and historical trends, this enables it to make more data-informed trading decisions. It also integrates established trading strategies and expert insights as augmented tools, leverages existing financial knowledge and combines it with its data-driven insights for more robust and explainable decision-making [42], [70]. Furthermore, the system incorporates a task-specific prompt generator that curates prompts for diverse trading scenarios and outcomes enabling FinAgent to interact effectively with the LLMs and guide output towards generating actionable trading decisions. FinAgent can learn from its experiences, and adapt to changing market dynamics, by dual analyzing market intelligence and price movements through a (low-level reflection) and reflecting on past trading outcomes (high-level reflection). Thus, it continuously improves its trading strategies [70].

With diversified memory retrieval, it supplies market intelligence and reflection capabilities with efficient storage and retrieval capabilities. The robust memory retrieval allows it to access past market intelligence from multiple perspectives, enhancing its understanding of market patterns and trends. In 2022-2024, FinAgent were tested on real datasets and Fin Agent's performance was evaluated against 12 baseline methods, including rule-based trading strategies, machine learning and deep learning models, reinforcement learning algorithms, and other LLM-based trading systems [70]. The dataset encompassed five major tech stocks (AAPL, AMZN, GOOGL, MSFT, TSLA) and one cryptocurrency (ETHUSD) [70]. FinAgent demonstrated better performance across traditional financial metrics, surpassing baseline methods, especially profitability. Its success is attributed to the effective integration of multimodal market intelligence, leveraging augmented tools and expert knowledge, utilising diversified memory retrieval, and engaging in dual-level reflection for continuous learning and adaptation.

FinAgent is a promising advancement that combines the reasoning capabilities of LLMs with the adaptability of reinforcement learning and a comprehensive suite of specialized modules set a new benchmark for the AI field. FinAgent' architecture and functionality align with core RAG Principles: which are external knowledge integration, contextualized decision-making, leverages memory and retrieval and reasoning and reflective capabilities [79], [68] [70]. However, it differs from a traditional RAG, it handles multimodal data including text, numerical and visual information while rag deals with text. It integrates the RAG Principle in reinforcement Learning enabling dynamic adaptation of the market not typically found in a standard RAG. Sources lack insights on retrieval methods such as keyword-based, and semantic similarity which is discussed in detail in context. Overall, it integrates an rag-approach to enhance decision-making in financial trading. In conclusion, the successful application of FinAagent on real-world datasets demonstrates its potential to transform investment strategies and enhance trading outcomes.

5.5.2. The Unified Medical Language System

The Unified Medical Language System (UMLS), is an impactful and comprehensive resource developed by the U.S. National Library of Medicine that integrates a vast number of medical vocabularies and standards. Domiciled at the bottom level of the multi-level medical graph it plays a crucial role in improving interoperability in healthcare systems whilst facilitating medical diagnosis and treatment recommendations [45], [43].

A significant challenge for data integration and analysis in the biomedical domain arises from the existence of numerous biomedical literature databases, and biomedical ontologies (containing various names of genes, proteins, diseases, and molecular functions). Each ontology exhibits its controlled vocabulary and nomenclature, creating difficulties in data integration. UMLs unify this vast and diverse landscape by providing a unified representation of medical concepts and their relationships. Strategically, its role as a foundational data source, grounds medical entities and their relationships within the multi-level medical graph. With a key role in enhancing data exchange and comprehension, the structure helps int entity linking and bridge the knowledge gap for Retrieval Augmented applications [45], [105], [77]. Additionally, it facilitates medical diagnosis and treatment recommendations as it provides a standardized and comprehensive medical language structure, which aids the development of tools and applications for medical diagnosis and treatment recommendations.

The resource constitutes components for graph construction, i.e., meta-thesaurus, semantic network, and specialized lexicon.

1. The meta-thesaurus integrates over 2 million medical vocabularies (SNOMED CT, ICD-10, etc.) for close to a million concepts from more than 60 families of biomedical vocabularies, as well as 12 million relations to provide a unified representation of medical concepts. 2. The semantic network organizes these concepts hierarchically and defines relationships between them. The lexicon offers linguistic insights to help with natural language processing tasks in the medical domain [106], [43].

UMLS is utilized in Medical Graph RAG via entity linking and foundational Medical Knowledge. It links medical vocabularies by comparing their text embedding and provides a foundation for most multilevel medical graphs by offering well-defined medical terminology and their relationships [77], [45].

Improved accuracy, interoperability, reduced hallucinations, enhanced reasoning and decision-making, and reliable information are some of the benefits accrued UMLS in Medical Graph RAG [77], [45]. UMLS fosters system Integration with multiple healthcare information systems, facilitating seamless data exchange and a more holistic understanding of patient information. The UMLS ensures that the retrieved information is based on reliable and established medical knowledge.

In practice, an LLM fine-tuned on a medical corpus integrated with UMLS could access up-to-date information on drug interactions, disease symptoms, and treatment protocols, significantly enhancing its ability to provide accurate diagnoses and recommendations. UMLS may support clinical decisions by aiding healthcare professionals in making informed decisions by providing up-to-date information on drug interactions, treatment options, and potential complications [23], [45], [47].

In research, studies have depicted how UMLS evaluate and improve the performance of LLMs in diagnosis generation and found that grounding LLM predictions using UMLS knowledge paths led to performance gains [45]. Other studies have demonstrated the effectiveness of integrating UMLS knowledge into LLMs for medical question answering. The UMLS-augmented LLM framework showed improvements in the factuality, completeness, and relevance of generated answers [43].

Potential methods of integrating UMLS include medical education, medical research, and clinical decision support. LLMs, when integrated with UMLS, can be used to create personalized study plans, generate learning materials, and assist with medical writing as well as analyze vast amounts of medical literature, aiding researchers in extracting relevant information and summarizing findings [23], [45], [47].

While sources endorse the use of specialized knowledge bases, there are challenges involved stemming from KB Integration, maintenance and updates and bias mitigation. Systems integrating UMLS require sophisticated techniques to ensure seamless knowledge retrieval and utilization during text generation. UMLS needs regular updates to reflect the latest advancements in medical fields, requiring ongoing maintenance efforts to ensure the LLM's knowledge remains current. Additionally, using the entirety of the vast size of the UMLS Knowledge base can introduce noise and impact efficiency [45], [43]. Its symbolic nature of knowledge may require further adaptation to integrate effectively with the sub-symbolic representations used by LLMs. Developing methods to bridge this gap is crucial for seamless knowledge integration [45]. Lastly, there are minimal insights into how biases have been mitigated to accurately represent medical knowledge which is vital for medical systems.

Lastly, the performance of LLMs in medical tasks requires specialized metrics. Traditional metrics like ROUGE and BERT-based scores may not suffice. Current research emphasizes the need for robust evaluation metrics that align with human judgment in healthcare [45]. In conclusion, UMLS represents a critical resource in the ongoing development of sophisticated and reliable medical AI systems. Continued research and development, focusing on overcoming the identified challenges, will be crucial in harnessing the full potential of integrating the UMLS with LLMs for the advancement of medicine.

5.5.3. Codex LLMs

Codex, a variant of GPT-3 is a Large Language Model (LLM) developed by OpenAI and is known for its use in code generation, test assertion generation, program repair, and documentation [107], [35], [28]. Like other GPT models, Codex is based on a transformer architecture and predicts the probability of the next token in a sequence. However, Codex is uniquely trained on publicly available data from GitHub, programming documentation, and coding forums making it adept at understanding and generating code [107], [28]. Codex can be augmented with retrieval mechanisms, forming Retrieval-Augmented Generation (RAG) systems. This integration combines Codex's generative capabilities to access and retrieve relevant information from external knowledge sources. Codex has numerous applications in various areas of software development, offering powerful tools for both developers and researchers. Codex excels in code generation and completion, generates code blocks from natural language inputs, and provides context-aware code suggestions, showing effectiveness in platforms like Leetcode [107]. In test generation, Codex automatically creates unit tests, achieving higher code coverage than traditional methods, as seen with tools like CoverUp [107], [108]. Codex is being applied to automatically fix bugs, with researchers fine-tuning Codex on bug-fixing datasets and exploring methods to guide its suggestions while minimizing token usage. Through the UniLog framework, Codex also aids in log statement generation, where it generates log messages and predicts verbosity levels [28]. Finally, Codex's training on programming documentation helps improve code maintainability and developer understanding.

Codex, a variant of GPT-3, is a large language model (LLM) developed by OpenAI known for its use in code generation, test assertion generation, program repair, and documentation [107], [35], [28]. Like other GPT models, Codex is based on the transformer architecture and predicts the probability of the next token in a sequence. However, Codex is uniquely trained on publicly available data from sources like GitHub, programming documentation, and coding forums, making it highly adept at understanding and generating code [107], [28].

Codex can be augmented with retrieval mechanisms, forming Retrieval-Augmented Generation (RAG) systems that combine its generative capabilities with external knowledge sources. This integration enhances Codex's ability to retrieve relevant information, improving code relevance and accuracy.

Codex has numerous applications in software development, providing powerful tools for both developers and researchers. It excels in code generation and completion, generating code blocks from natural language inputs and offering context-aware code suggestions. Codex has shown effectiveness on platforms like Leetcode [107]. For test generation, Codex automatically creates unit tests, achieving higher code coverage than traditional methods, as demonstrated with tools like CoverUp [107], [108]. Codex is also applied to bug fixing, where it is fine-tuned on bug-fixing datasets to guide its suggestions while minimizing token usage. Using frameworks like UniLog, Codex supports log statement generation by producing log messages and predicting verbosity levels [28]. Additionally, its training in programming documentation enhances code maintainability and developer understanding.

A notable application of Codex is its integration into GitHub Copilot, a code assistant tool that operates within integrated development environments (IDEs) such as Visual Studio Code and JetBrains [28], [109]. Integrated by an intermediary Copilot plugin, Codex connects to IDEs and captures user inputs, analyzes the context (from comments, function, or signatures) and the model generates context-aware code and completion recommendations. Key architectural components integrated in Codex are an API layer, context engine, model fine-tuning, updates, and security and privacy considerations. The data flow starts at the input capture from the user within the IDE. The input subsequently goes into context parsing, then the parsed-out input is transmitted to Codex-Model for code generation. Finally, the generated code is returned to the GitHub Copilot plugin within the IDE for user review, selection, and feedback [110], [109].

Codex has been integrated into multiple RAG systems, each employing unique techniques: EPR released in 2021, integrates Codex with other LLMs like GPT-3, J, and Neo while adapting demonstration-based retrieval and inference-time reasoning [105]. Released in 2022, OpenBook employs a GOPHER LLM for input, question, answering and fact verification and combines semantic encoding and symbolic reasoning [105]. In the same year, DSP was released utilizing GPT-3.5 for tasks such as open question answering, multiple-choice question answering, and conversational question answering and leveraged on ColBERTv2 for retrieval [105]. Later IRCoT was released in 2023 and leverages GPT-3 and Flan-T5 for open question answering and BM25 for search [35], [105]. Additionally, CodeLLaMA has been adapted for code-related tasks, potentially overlapping with Codex's capabilities [105].

Despite its versatility, Codex faces challenges, including limited contextual understanding in large codebases [35], high computational costs leading to latency, performance bottlenecks, security of suggested code, bias present in existing code repositories, Codex over-reliance without understanding the underlying concepts [28]. However, its integration brings benefits such as scalability, cross-language support, adaptability, and real-time assistance. Nonetheless, this calls for the promotion of responsible use of code and understanding of programming principles.

Evaluation of Codex-generated code is an ongoing research area. While platforms like Leetcode provide a dataset of coding problems, the current benchmarks (noise robustness, negative rejection, information integration and counterfactual robustness) exhibit limitations for evaluating LLM-generated code [63]. Therefore, research should be directed to establish more comprehensive and reliable benchmarks for assessing the capabilities of code-generating LLMs.

5.5.4. BloombergGPT

This case study diverges from the RAG-enhanced model, demonstrating a proprietary LLM within the financial sector. BloombergGPT is deemed as the first mixed-domain FinLLM to leverage the BLOOM model [76], [13]. With 50 billion parameters, the LLM aligns with the concept of domain-specific pre-training, and trained on a large general corpus (345 billion tokens) from books, articles, websites, and code and an even larger financial corpus (363 billion tokens) known as FinPile, includes data from various sources Web, News, Filings, Press releases, and Bloomberg's proprietary data.

BloombergGPT evaluated its performance across a wide range of tasks; 5 benchmark Financial NLP tasks 12 internal tasks and 42 General-purpose NLP tasks. Sources highlight that BloombergGPT achieved a 43% EM Accuracy score on the ConvFinQA dataset, however, this score is slightly below the performance of a general crowd (47%) and significantly lower than the score achieved by GPT-4 with zero-shot prompting (69%-76%) [13]. BloombergGPT design and training data suggest it could be valuable for a variety of financial tasks, including sentiment analysis, risk assessment, portfolio management, financial forecasting, fraud detection, and regulatory compliance.

Similar to the challenges other LLMs face, BloombergGPT and other FinLLMs face their own set of challenges data quality and bias, model explainability, regulatory compliance, ethical concerns about potential market manipulation, unfair advantages for certain investors, and the displacement of human financial professionals. Addressing these concerns through responsible development and regulation is crucial.

While Bloomberg is a closed model there are other open-sourced models including FinGPT and LLaMA. FinGPT another FinLLM, uses an open-source framework and datasets while relying on instruction fine-tuning and techniques like Low-rank adaptation. While BloombergGPT's training data and evaluation tasks cover a broad range of financial topics, the source notes that the evaluation datasets used for most instruction-fine-tuned FinLLMs overlap.

BloombergGPT represents a significant step in the evolution of financial LLMs leveraging vast amounts of proprietary data and paved the way for the rise of potential for financial applications from data analysis, sentiment analysis, enhanced QA services, and detecting emerging events that may affect stock values to investment research and customer service.

As with any LLM, it is important to be aware of potential limitations, bias in financial data, closed-source nature, and ethical considerations regarding fairness, transparency, and accountability need to be carefully addressed [76], [13].

5.6. Recommendations for Future Development and Implementation

This section provides key recommendations for advancing the development and effective implementation of Large Language Models (LLMs) integrated with knowledge bases system. It highlights strategies to improve modularity, optimize iterative processes, and carefully evaluate the choice between open-source and proprietary models. By adopting these recommendations, organizations can enhance the adaptability, scalability, and efficiency of LLMs, ensuring they remain relevant and effective in dynamic, real-world applications.

5.6.1. Modularity in Pipelines

A modular pipeline is crucial for enhancing the flexibility and scalability of LLM systems. Adopting self-improving frameworks like DsPy allows the system's individual components to be managed independently, facilitating easier maintenance, updates, and scalability. By integrating advanced reasoning techniques, such as chain of thought and graph of thought, into the modular pipeline, systems can optimize prompt generation and reasoning. These techniques improve the overall retrieval process, ensuring that

LLMs can efficiently generate and refine prompts to access the most relevant information.

The modular approach enables the integration of new technologies and strategies without overhauling the entire system. As components like the retriever or generator can be easily swapped or updated, this approach ensures that the LLM remains adaptable to specific domain requirements, offering flexibility across different industries and applications.

5.6.2. Iterative Process

LLMs should not be treated as static systems but as part of an ongoing iterative process. Continuous fine-tuning and updates are essential to maintaining the effectiveness of the retrieval mechanisms and ensuring that the models adapt to evolving data sources. By regularly updating knowledge bases, retraining models on fresh data, and optimizing retrieval techniques, organizations can keep the system aligned with new information and emerging trends.

An iterative approach ensures that LLMs remain agile and responsive to the dynamic needs of industries. It supports continuous improvement, allowing organizations to address shifting industry standards, regulatory changes, and new technological advancements. This approach is especially valuable in rapidly evolving sectors such as healthcare, finance, and technology.

5.6.3. Open Source vs. Proprietary Models

A key consideration for LLMs integration is whether to use open-source models or proprietary models. Proprietary models, while highly effective, come with significant financial costs and potential limitations in terms of customization. They may also restrict flexibility due to vendor lock-in.

On the other hand, open-source models offer greater control and customization options, particularly for domain-specific applications. Fine-tuning smaller open-source models on specialized datasets (e.g., medical, legal, or financial data) can yield performance comparable to proprietary models for specific tasks. Furthermore, open-source models allow organizations to maintain full control over their data, addressing concerns about data privacy and security.

In regulated industries such as healthcare, where compliance with privacy standards is critical, open-source models offer a significant advantage by enabling organizations to maintain full oversight of their data processing. A well-engineered RAG pipeline can help enhance the performance of open-source models, bridging the gap between them and their proprietary counterparts.

5.6.4. Recommendations for LLMs Optimization

Finally, to optimize the integration and performance of LLMs across various applications, several general key strategies should be prioritized:

- Enhance Retrieval Mechanisms: Implement advanced retrieval strategies to ensure that LLMs can access the most relevant and accurate information from external knowledge sources. Hybrid retrieval techniques, which combine sparse and dense retrieval, as well as multi-stage architectures such as re-ranking, can significantly improve the quality of retrieved data.

- Improve Text Generation Augmentation: Focus on optimizing how LLMs integrate retrieved information into their generation process. Experimenting with integration techniques at various layers of the model—such as input-layer and intermediate-layer integration—can improve the model's ability to generate more contextually appropriate and precise responses.

- Adapt to Domain-Specific Needs: Tailor LLMs to meet the specific requirements of different domains by adjusting their retrieval strategies and knowledge bases accordingly. Ensuring the model is well-suited to its domain enhances the relevance and accuracy of its outputs, making it more effective for specialized applications.

- Mitigate Hallucinations and Enhance Trustworthiness: Incorporate methods to reduce hallucinations and ensure that the outputs of LLMs are reliable. Strategies such as verifying the relevance and quality of retrieved information, using authoritative external sources, and filtering out irrelevant or noisy content can enhance the overall trustworthiness of LLM responses.

6. Conclusion

This paper has explored the integration of Large Language Models (LLMs) with knowledge bases, beginning with an overview of LLMs to understand their key capabilities and real-world applications. It summarizes the challenges faced in implementing LLMs for real-world scenarios. We also examined existing solutions and innovations aimed at overcoming these challenges, emphasizing the value of hybrid approaches that leverage the strengths of both LLMs and knowledge bases.

Finally, we comprehensively explored the potential for enhancing AI capabilities by integrating LLMs with knowledge-based systems. By reviewing the current state of integration and examining techniques and typical case studies, we have identified key benefits, challenges, and future recommendations. Our findings demonstrate that this integration can significantly improve data contextualization, enhance model accuracy, and facilitate more reliable knowledge retrieval across various domains. Despite the promising advancements, challenges remain, particularly in dynamic knowledge management and model flexibility. Moving forward, future research should focus on refining integration techniques, optimizing retrieval processes, and ensuring that knowledge bases remain up-to-date and relevant.

In conclusion, integrating LLMs with knowledge bases offers significant potential to advance AI technology and improve its application across diverse sectors. The continued evolution of this field promises to result in more intelligent, accurate, and context-aware AI systems, with substantial benefits for various industries and organizations.

References

[1] Shervin Minaee, Tomas Mikolov, Narjes Nikzad, Meysam Chenaghlu, Richard Socher, Xavier Amatriain, and Jianfeng Gao. Large language models: A survey, 2024.

[2] Zichong Wang, Zhibo Chu, Thang Viet Doan, Shiwen Ni, Min Yang, and Wenbin Zhang. History, development, and principles of large language models: an introductory survey, 2024.

[3] Sanjay Kukreja, Tarun Kumar, Amit Purohit, Abhijit Dasgupta, and Debashis Guha. A literature survey on open source large language models, 2024.

[4] Teo Susnjak, Peter Hwang, Napoleon H Reyes, Andre LC Barczak, Timothy R McIntosh, and Surangika Ranathunga. Automating research synthesis with domain-specific large language model fine-tuning, 2024.

[5] Timothy R McIntosh, Teo Susnjak, Tong Liu, Paul Watters, and Malka N Halgamuge. The inadequacy of reinforcement learning from

human feedback-radicalizing large language models via semantic vulnerabilities, 2024.

[6] Nuraini Sulaiman and Farizal Hamzah. Evaluation of transfer learning and adaptability in large language models with the glue benchmark, 2024.

[7] Mohaimenul Azam Khan Raiaan, Md. Saddam Hossain Mukta, Kaniz Fatema, Nur Mohammad Fahad, Sadman Sakib, Most Marufatul Jannat Mim, Jubaer Ahmad, Mohammed Eunus Ali, and Sami Azam. A Review on Large Language Models: Architectures, Applications, Taxonomies, Open Issues and Challenges. *IEEE Access*, 12:26839–26874, 2024.

[8] Amanda Kau, Xuzeng He, Aishwarya Nambissan, Aland Astudillo, Hui Yin, and Amir Aryani. Combining Knowledge Graphs and Large Language Models, July 2024. arXiv:2407.06564 [cs].

[9] Huy Quoc To, Ming Liu, and Guangyan Huang. Towards Efficient Large Language Models for Scientific Text: A Review, August 2024. arXiv:2408.10729 [cs].

[10] Muhammad Usman Hadi, Qasem Al Tashi, Rizwan Qureshi, Abbas Shah, Amgad Muneer, Muhammad Irfan, Anas Zafar, Muhammad Bilal Shaikh, Naveed Akhtar, Jia Wu, and Seyedali Mirjalili. A Survey on Large Language Models: Applications, Challenges, Limitations, and Practical Usage, July 2023.

[11] Bonan Min, Hayley Ross, Elior Sulem, Amir Pouran Ben Veyseh, Thien Huu Nguyen, Oscar Sainz, Eneko Agirre, Ilana Heinz, and Dan Roth. Recent Advances in Natural Language Processing via Large Pre-Trained Language Models: A Survey, November 2021. arXiv:2111.01243 [cs].

[12] Katikapalli Subramanyam Kalyan. A survey of GPT-3 family large language models including ChatGPT and GPT-4. *Natural Language Processing Journal*, 6:100048, March 2023.

[13] Jean Lee, Nicholas Stevens, Soyeon Caren Han, and Minseok Song. A Survey of Large Language Models in Finance (FinLLMs), February 2024. arXiv:2402.02315 [cs, q-fin].

[14] Desta Haileselassie Hagos, Rick Battle, and Danda B. Rawat. Recent Advances in Generative AI and Large Language Models: Current Status, Challenges, and Perspectives, August 2024. arXiv:2407.14962 [cs].

[15] Jinyuan Fang, Zaiqiao Meng, and Craig Macdonald. TRACE the Evidence: Constructing Knowledge-Grounded Reasoning Chains for Retrieval-Augmented Generation, June 2024. arXiv:2406.11460 [cs].

[16] Amin Beheshti. Empowering generative ai with knowledge base 4.0: Towards linking analytical, cognitive, and generative intelligence, Jul 2023.

[17] Ilyas Aden, Christopher HT Child, and Constantino Carlos Reyes-Aldasoro. International classification of diseases prediction from mimiic-iii clinical text using pre-trained clinicalbert and nlp deep learning models achieving state of the art. *Big Data and Cognitive Computing*, 8(5):47, 2024.

[18] Siyu Ding, Junyuan Shang, Shuohuan Wang, Yu Sun, Hao Tian, Hua Wu, and Haifeng Wang. Ernie-doc: A retrospective long-document modeling transformer. *arXiv preprint arXiv:2012.15688*, 2020.

[19] Quanjun Zhang, Chunrong Fang, Yang Xie, YuXiang Ma, Weisong Sun, Yun Yang, and Zhenyu Chen. A Systematic Literature Review on Large Language Models for Automated Program Repair, May 2024. arXiv:2405.01466 [cs].

[20] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding, 2018.

[21] Enkelejda Kasneci, Kathrin Sessler, Stefan Küchemann, Maria Bannert, Daryna Dementieva, Frank Fischer, Urs Gasser, Georg Groh, Stephan Günnemann, Eyke Hüllermeier, Stephan Krusche, Gitta Kutyniok, Tilman Michaeli, Claudia Nerdel, Jürgen Pfeffer, Oleksandra Poquet, Michael Sailer, Albrecht Schmidt, Tina Seidel, Matthias Stadler, Jochen Weller, Jochen Kuhn, and Gjergji Kasneci. ChatGPT for good? On opportunities and challenges of large language models for education. *Learning and Individual Differences*, 103:102274, April 2023.

[22] Wenqi Fan, Yujuan Ding, Liangbo Ning, Shijie Wang, Hengyun Li, Dawei Yin, Tat-Seng Chua, and Qing Li. A Survey on RAG Meeting LLMs: Towards Retrieval-Augmented Large Language Models, June 2024. arXiv:2405.06211 [cs].

[23] Alaa Abd-alrazaq, Rawan AlSaad, Dari Alhuwail, Arfan Ahmed, Padraig Mark Healy, Syed Latifi, Sarah Aziz, Rafat Damseh, Sadam Alabed Alrazak, and Javaid Sheikh. Large Language Models in Medical Education: Opportunities, Challenges, and Future Directions. *JMIR Medical Education*, 9:e48291, June 2023.

[24] Xiaokai Wei, Shen Wang, Dejiao Zhang, Parminder Bhatia, and Andrew Arnold. Knowledge Enhanced Pretrained Language Models: A Compreshensive Survey, October 2021. arXiv:2110.08455 [cs].

[25] Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Rémi Louf, Morgan Funtowicz, Joe Davison, Sam Shleifer, Patrick von Platen, Clara Ma, Yacine Jernite, Julien Plu, Canwen Xu, Teven Le Scao, Sylvain Gugger, Mariama Drame, Quentin Lhoest, and Alexander M. Rush. HuggingFace's Transformers: State-of-the-art Natural Language Processing, 2019.

[26] Erik Keanius. Domain adaptation of LLMs: A study of content generation, RAG, and fine-tuning, 2024.

[27] Chengrui Wang, Qingqing Long, Meng Xiao, Xunxin Cai, Chengjun Wu, Zhen Meng, Xuezhi Wang, and Yuanchun Zhou. BioRAG: A RAG-LLM Framework for Biological Question Reasoning, August 2024. arXiv:2408.01107 [cs].

[28] Junjielong Xu, Ziang Cui, Yuan Zhao, Xu Zhang, Shilin He, Pinjia He, Liqun Li, Yu Kang, Qingwei Lin, Yingnong Dang, Saravan Rajmohan, and Dongmei Zhang. Unilog: Automatic logging via llm and in-context learning, 02 2024.

[29] Patrick Lewis, Ethan Perez, Aleksandra Piktus, Fabio Petroni, Vladimir Karpukhin, Naman Goyal, Heinrich Küttler, Mike Lewis, Wen-tau Yih, Tim Rocktäschel, et al. Retrieval-augmented generation for knowledge-intensive nlp tasks. *Advances in Neural Information Processing Systems*, 33:9459–9474, 2020.

[30] Tyna Eloundou, Sam Manning, Pamela Mishkin, and Daniel Rock. Gpts are gpts: An early look at the labor market impact potential of large language models, 2023.

[31] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need, Jun 2017. MAG ID: 2963403868.

[32] Xiaonan Li, Changtai Zhu, Linyang Li, Zhangyue Yin, Tianxiang Sun, and Xipeng Qiu. LLatrieval: LLM-Verified Retrieval for Verifiable Generation, March 2024. arXiv:2311.07838 [cs].

[33] Shirui Pan, Linhao Luo, Yufei Wang, Chen Chen, Jiapu Wang, and Xindong Wu. Unifying Large Language Models and Knowledge Graphs: A Roadmap. *IEEE Transactions on Knowledge and Data Engineering*, 36(7):3580–3599, July 2024. arXiv:2306.08302 [cs].

[34] N. Kitaev et al. Reformer: The efficient transformer, 2020.

[35] Yunfan Gao, Yun Xiong, Xinyu Gao, Kangxiang Jia, Jinliu Pan, Yuxi Bi, Yi Dai, Jiawei Sun, Meng Wang, and Haofen Wang. Retrieval-Augmented Generation for Large Language Models: A Survey, March 2024. arXiv:2312.10997 [cs].

[36] Taiyu Zhang, Xuesong Zhang, Robbe Cools, and Adalberto L. Simeone. Focus Agent: LLM-Powered Virtual Focus Group, September 2024. arXiv:2409.01907 [cs].

[37] Zabir Al Nazi and Wei Peng. Large language models in healthcare and medical domain: A review. In *Informatics*, volume 11, page 57. MDPI, 2024.

[38] Yingqing He, Zhaoyang Liu, Jingye Chen, Zeyue Tian, Hongyu Liu, Xiaowei Chi, Runtao Liu, Ruibin Yuan, Yazhou Xing, Wenhai Wang, Jifeng Dai, Yong Zhang, Wei Xue, Qifeng Liu, Yike Guo, and Qifeng Chen. LLMs Meet Multimodal Generation and Editing: A Survey, June 2024. arXiv:2405.19334 [cs].

[39] Ruiyao Xu and Kaize Ding. Large Language Models for Anomaly and Out-of-Distribution Detection: A Survey, September 2024. arXiv:2409.01980 [cs].

[40] Linhao Luo, Yuan-Fang Li, Gholamreza Haffari, and Shirui Pan. Reasoning on Graphs: Faithful and Interpretable Large Language Model Reasoning, February 2024. arXiv:2310.01061 [cs].

[41] Richard Fang, Rohan Bindu, Akul Gupta, and Daniel Kang. LLM Agents can Autonomously Exploit One-day Vulnerabilities, April 2024. arXiv:2404.08144 [cs].

[42] Yuqi Nie, Yaxuan Kong, Xiaowen Dong, John M. Mulvey, H. Vincent Poor, Qingsong Wen, and Stefan Zohren. A Survey of Large Language Models for Financial Applications: Progress, Prospects and Challenges, June 2024. arXiv:2406.11903 [q-fin].

[43] Rui Yang, Edison Marrese-Taylor, Yuhe Ke, Lechao Cheng, Qingyu Chen, and Irene Li. Integrating umls knowledge into large language models for medical question answering, 2023.

[44] Seyma Handan Akyon, Fatih Cagatay Akyon, Ahmet Sefa Camyar, Fatih Hızlı, Talha Sari, and Şamil Hızlı. Evaluating the capabilities of generative ai tools in understanding medical papers: Qualitative study. *JMIR Medical Informatics*, 12(1):e59258, 2024.

[45] Majid Afshar, Yanjun Gao, Deepak Gupta, Emma Croxford, and Dina Demner-Fushman. On the role of the umls in supporting diagnosis generation proposed by large language models. *Journal of Biomedical Informatics*, 157:104707, 2024.

[46] Karan Singhal, Tao Tu, Juraj Gottweis, Rory Sayres, Ellery Wulczyn, Le Hou, Kevin Clark, Stephen Pfohl, Heather Cole-Lewis, Darlene Neal, et al. Towards expert-level medical question answering with large language models. *arXiv preprint arXiv:2305.09617*, 2023.

[47] M. Karabacak and K. Margetis. Embracing large language models for medical applications: Opportunities and challenges. *Cureus*, 15(5):e39305, May 2023.

[48] Zhibin Gou, Zhihong Shao, Yeyun Gong, Yelong Shen, Yujiu Yang, Minlie Huang, Nan Duan, and Weizhu Chen. ToRA: A Tool-Integrated Reasoning Agent for Mathematical Problem Solving, February 2024. arXiv:2309.17452 [cs].

[49] Hossein Saiedian. Leveraging Large Language Models in Education: Enhancing Learning and Teaching. In *2023 ASEE Midwest Section Conference Proceedings*, page 46353, University of Nebraska-Lincoln, Lincoln, Nebraska, July 2024. ASEE Conferences.

[50] Adina Williams, Nikita Nangia, and Samuel Bowman. A Broad-Coverage Challenge Corpus for Sentence Understanding through Inference. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pages 1112–1122, New Orleans, Louisiana, 2018. Association for Computational Linguistics.

[51] Karthik Soman, Peter W Rose, John H Morris, Rabia E Akbas, Brett Smith, Braian Peetoom, Catalina Villouta-Reyes, Gabriel Cerono, Yongmei Shi, Angela Rizk-Jackson, Sharat Israni, Charlotte A Nelson, Sui Huang, and Sergio E Baranzini. Biomedical knowledge graph-optimized prompt generation for large language models. *Bioinformatics*, 40(9):btae560, September 2024.

[52] Long Ouyang, Jeff Wu, Xu Jiang, Diogo Almeida, Carroll L Wainwright, Pamela Mishkin, Chong Zhang, Sandhini Agarwal, Katarina Slama, Alex Ray, John Schulman, Jacob Hilton, Fraser Kelton, Luke Miller, Maddie Simens, Amanda Askell, Peter Welinder, Paul Christiano, Jan Leike, and Ryan Lowe. Training language models to follow instructions with human feedback, 2022.

[53] Junde Wu, Jiayuan Zhu, and Yunli Qi. Medical Graph RAG: Towards Safe Medical Large Language Model via Graph Retrieval-Augmented Generation, August 2024. arXiv:2408.04187 [cs].

[54] Xin Cheng, Di Luo, Xiuying Chen, Lemao Liu, Dongyan Zhao, and Rui Yan. Lift yourself up: Retrieval-augmented text generation with self memory, 2023.

[55] Hongzhi Zhang and M. Omair Shafiq. Triple-Aware Reasoning: A Retrieval-Augmented GenerationApproach for Enhancing Question-Answering Tasks withKnowledge Graphs and Large Language Models, may 27 2024. https://caiac.pubpub.org/pub/bytcy6lo.

[56] Shengjie Ma, Chengjin Xu, Xuhui Jiang, Muzhi Li, Huaren Qu, and Jian Guo. Think-on-Graph 2.0: Deep and Interpretable Large Language Model Reasoning with Knowledge Graph-guided Retrieval, August 2024. arXiv:2407.10805 [cs].

[57] Yinghao Zhu, Changyu Ren, Shiyun Xie, Shukai Liu, Hangyuan Ji, Zixiang Wang, Tao Sun, Long He, Zhoujun Li, Xi Zhu, and Chengwei Pan. REALM: RAG-Driven Enhancement of Multimodal Electronic Health Records Analysis via Large Language Models, February 2024. arXiv:2402.07016 [cs].

[58] Xinyang Hu, Fengzhuo Zhang, Siyu Chen, and Zhuoran Yang. Unveiling the Statistical Foundations of Chain-of-Thought Prompting Methods, August 2024. arXiv:2408.14511 [cs, math, stat].

[59] Yun-Tong Yang and Hong-Gang Luo. Topological or not? A unified pattern description in the one-dimensional anisotropic quantum XY model with a transverse field, February 2023. arXiv:2302.13866 [cond-mat].

[60] Tailin Liang, John Glossner, Lei Wang, Shaobo Shi, and Xiaotong Zhang. Pruning and Quantization for Deep Neural Network Acceleration: A Survey, 2021.

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

[62] Jordan Hoffmann, Sebastian Borgeaud, Arthur Mensch, Elena Buchatskaya, Trevor Cai, Eliza Rutherford, Diego de Las Casas, Lisa Anne Hendricks, Johannes Welbl, Aidan Clark, et al. Training compute-optimal large language models. *arXiv preprint arXiv:2203.15556*, 2022.

[63] Jiawei Chen, Hongyu Lin, Xianpei Han, and Le Sun. Benchmarking Large Language Models in Retrieval-Augmented Generation. *Proceedings of the AAAI Conference on Artificial Intelligence*, 38(16):17754–17762, March 2024.

[64] Nicholas Matsumoto, Jay Moran, Hyunjun Choi, Miguel E Hernandez, Mythreye Venkatesan, Paul Wang, and Jason H Moore. KRAGEN: a knowledge graph-enhanced RAG framework for biomedical problem solving using large language models. *Bioinformatics*, 40(6):btae353, June 2024.

[65] Bodong Chen, Xinran Zhu, and Fernando Díaz Del Castillo H. Integrating generative AI in knowledge building. *Computers and Education: Artificial Intelligence*, 5:100184, November 2023. MAG ID: 4388599367.

[66] Ahmed Menshawy, Zeeshan Nawaz, and Mahmoud Fahmy. Navigating challenges and technical debt in large language models deployment. In *Proceedings of the 4th Workshop on Machine Learning and Systems*, pages 192–199, 2024.

[67] Xin Su, Tiep Le, Steven Bethard, and Phillip Howard. Semi-Structured Chain-of-Thought: Integrating Multiple Sources of Knowledge for Improved Language Model Reasoning, April 2024. arXiv:2311.08505 [cs].

[68] Tilmann Bruckhaus. Rag does not work for enterprises, 2024.

[69] Miao Zheng, Hao Liang, Fan Yang, Haoze Sun, Tianpeng Li, Lingchu Xiong, Yan Zhang, Youzhen Wu, Kun Li, Yanjun Shen, Mingan Lin, Tao Zhang, Guosheng Dong, Yujing Qiao, Kun Fang, Weipeng Chen, Bin Cui, Wentao Zhang, and Zenan Zhou. Pas: Data-efficient plug-and-play prompt augmentation system, August 2024. arXiv:2407.06027 [cs].

[70] Wentao Zhang, Lingxuan Zhao, Haochong Xia, Shuo Sun, Jiaze Sun, Molei Qin, Xinyi Li, Yuqing Zhao, Yilei Zhao, Xinyu Cai, Longtao Zheng, Xinrun Wang, and Bo An. A Multimodal Foundation Agent for Financial Trading: Tool-Augmented, Diversified, and Generalist, June 2024. arXiv:2402.18485 [q-fin].

[71] Zhengbao Jiang, Frank F. Xu, Luyu Gao, Zhiqing Sun, Qian Liu, Jane Dwivedi-Yu, Yiming Yang, Jamie Callan, and Graham Neubig. Active Retrieval Augmented Generation, October 2023. arXiv:2305.06983 [cs].

[72] Xijun Wang, Dongshan Ye, Chenyuan Feng, Howard H. Yang, Xiang Chen, and Tony Q. S. Quek. Trustworthy Image

Semantic Communication with GenAI: Explainablity, Controllability, and Efficiency, 2024. ARXIV_ID: 2408.03806 S2ID: 3e5473ecb44e13e6bb08b477623ab39da551943b.

[73] Fiona Fui-Hoon Nah, Ruilin Zheng, Jingyuan Cai, Keng Siau, and Langtao Chen. Generative AI and ChatGPT: Applications, challenges, and AI-human collaboration. *Journal of Information Technology Case and Application Research*, 25(3):277–304, July 2023.

[74] Ian J. Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative Adversarial Networks, 2014.

[75] D. P. Kingma and M. Welling. Auto-encoding variational bayes, 2013.

[76] Aidan Hogan, Eva Blomqvist, Michael Cochez, Claudia d'Amato, Gerard de Melo, Claudio Gutierrez, José Emilio Labra Gayo, Sabrina Kirrane, Sebastian Neumaier, Axel Polleres, Roberto Navigli, Axel-Cyrille Ngonga Ngomo, Sabbir M. Rashid, Anisa Rula, Lukas Schmelzeisen, Juan Sequeda, Steffen Staab, and Antoine Zimmermann. Knowledge Graphs. *ACM Computing Surveys*, 54(4):1–37, May 2022. arXiv:2003.02320 [cs].

[77] Minbyul Jeong, Jiwoong Sohn, Mujeen Sung, and Jaewoo Kang. Improving Medical Reasoning through Retrieval and Self-Reflection with Retrieval-Augmented Large Language Models, June 2024. arXiv:2401.15269 [cs].

[78] Emma Yann Zhang, Adrian David Cheok, Zhigeng Pan, Jun Cai, and Ying Yan. From Turing to Transformers: A Comprehensive Review and Tutorial on the Evolution and Applications of Generative Transformer Models. *Sci*, 5(4):46, December 2023.

[79] Peizhong Gao, Ao Xie, Shaoguang Mao, Wenshan Wu, Yan Xia, Haipeng Mi, and Furu Wei. Meta Reasoning for Large Language Models, June 2024. arXiv:2406.11698 [cs].

[80] Dr Kasnesis. Augmentation of Large Language Model capabilities with Knowledge Graphs, 2024.

[81] Weijian Xie, Xuefeng Liang, Yuhui Liu, Kaihua Ni, Hong Cheng, and Zetian Hu. WeKnow-RAG: An Adaptive Approach for Retrieval-Augmented Generation Integrating Web Search and Knowledge Graphs, August 2024. arXiv:2408.07611 [cs].

[82] Sijia Chen, Baochun Li, and Di Niu. Boosting of Thoughts: Trial-and-Error Problem Solving with Large Language Models, February 2024. arXiv:2402.11140 [cs].

[83] Ilia Stepin, Muhammad Suffian, Alejandro Catalá, and J. Alonso-Moral. How to Build Self-Explaining Fuzzy Systems: From Interpretability to Explainability [AI-eXplained], 2024. S2ID: 1d5127686d01fe1e66805c10f19284f865484632.

[84] Ling Yang, Zhaochen Yu, Tianjun Zhang, Shiyi Cao, Minkai Xu, Wentao Zhang, Joseph E. Gonzalez, and Bin Cui. Buffer of Thoughts: Thought-Augmented Reasoning with Large Language Models, June 2024. arXiv:2406.04271 [cs].

[85] Majid Afshar, Yanjun Gao, Deepak Gupta, Emma Croxford, and Dina Demner-Fushman. On the role of the umls in supporting diagnosis generation proposed by large language models. *Journal of Biomedical Informatics*, 157:104707, 2024.

[86] Lukas Bahr, Christoph Wehner, Judith Wewerka, José Bittencourt, Ute Schmid, and Rüdiger Daub. Knowledge Graph Enhanced Retrieval-Augmented Generation for Failure Mode and Effects Analysis, July 2024. arXiv:2406.18114 [cs].

[87] Garima Agrawal, Tharindu Kumarage, Zeyad Alghamdi, and Huan Liu. Can Knowledge Graphs Reduce Hallucinations in LLMs? : A Survey, March 2024. arXiv:2311.07914 [cs].

[88] Hasan Abu-Rasheed, Christian Weber, and Madjid Fathi. Knowledge Graphs as Context Sources for LLM-Based Explanations of Learning Recommendations, March 2024. arXiv:2403.03008 [cs].

[89] Julien Delile, Srayanta Mukherjee, Anton Van Pamel, and Leonid Zhukov. Graph-Based Retriever Captures the Long Tail of Biomedical Knowledge, February 2024. arXiv:2402.12352 [cs].

[90] 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, January 2024. arXiv:2401.07883 [cs].

[91] Yu Wang, Shiwan Zhao, Zhihu Wang, Heyuan Huang, Ming Fan, Yubo Zhang, Zhixing Wang, Haijun Wang, and Ting Liu. Strategic Chain-of-Thought: Guiding Accurate Reasoning in LLMs through Strategy Elicitation, September 2024. arXiv:2409.03271 [cs].

[92] Davit Janezashvili. Rag at large enterprises. `https://modulai.io/blog/rag-at-large-enterprises/`, Oct 2024. Accessed: 2024-10-25.

[93] Haoyu Wang, Ruirui Li, Haoming Jiang, Jinjin Tian, Zhengyang Wang, Chen Luo, Xianfeng Tang, Monica Cheng, Tuo Zhao, and Jing Gao. BlendFilter: Advancing Retrieval-Augmented Large Language Models via Query Generation Blending and Knowledge Filtering, July 2024. arXiv:2402.11129 [cs].

[94] Yujia Zhou, Zheng Liu, Jiajie Jin, Jian-Yun Nie, and Zhicheng Dou. Metacognitive Retrieval-Augmented Large Language Models, February 2024. arXiv:2402.11626 [cs].

[95] Masoomali Fatehkia, Ji Kim Lucas, and Sanjay Chawla. T-RAG: Lessons from the LLM Trenches, June 2024. arXiv:2402.07483 [cs].

[96] 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, April 2024. arXiv:2404.10384 [cs].

[97] Shunyu Yao, Jeffrey Zhao, Dian Yu, Nan Du, Izhak Shafran, Karthik Narasimhan, and Yuan Cao. ReAct: Synergizing Reasoning and Acting in Language Models, March 2023. arXiv:2210.03629 [cs].

[98] Jiejun Tan, Zhicheng Dou, Yutao Zhu, Peidong Guo, Kun Fang, and Ji-Rong Wen. Small Models, Big Insights: Leveraging Slim Proxy Models To Decide When and What to Retrieve for LLMs, May 2024. arXiv:2402.12052 [cs].

[99] Zhihong Shao, Yeyun Gong, Yelong Shen, Minlie Huang, Nan Duan, and Weizhu Chen. Enhancing Retrieval-Augmented Large Language Models with Iterative Retrieval-Generation Synergy, October 2023. arXiv:2305.15294 [cs].

[100] Yanming Liu, Xinyue Peng, Xuhong Zhang, Weihao Liu, Jianwei Yin, Jiannan Cao, and Tianyu Du. RA-ISF: Learning to Answer and Understand from Retrieval Augmentation via Iterative Self-Feedback, June 2024. arXiv:2403.06840 [cs].

[101] 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, Jul 2024. arXiv:2404.17723 [cs].

[102] Jialin Dong, Bahare Fatemi, Bryan Perozzi, Lin F. Yang, and Anton Tsitsulin. Don't Forget to Connect! Improving RAG with Graph-based Reranking, May 2024. arXiv:2405.18414 [cs].

[103] Taicheng Guo, Xiuying Chen, Yaqi Wang, Ruidi Chang, Shichao Pei, Nitesh V. Chawla, Olaf Wiest, and Xiangliang Zhang. Large language model based multi-agents: A survey of progress and challenges, 2024.

[104] Xiaoxi Li, Jiajie Jin, Yujia Zhou, Yuyao Zhang, Peitian Zhang, Yutao Zhu, and Zhicheng Dou. From Matching to Generation: A Survey on Generative Information Retrieval, May 2024. arXiv:2404.14851 [cs].

[105] C. Zakka, R. Shad, A. Chaurasia, A. R. Dalal, J. L. Kim, M. Moor, R. Fong, C. Phillips, K. Alexander, E. Ashley, J. Boyd, K. Boyd, K. Hirsch, C. Langlotz, R. Lee, J. Melia, J. Nelson, K. Sallam, S. Tullis, M. A. Vogelsong, and W. Hiesinger. Almanac - retrieval-augmented language models for clinical medicine, 2024.

[106] Olivier Bodenreider. The unified medical language system (umls): integrating biomedical terminology. *Nucleic acids research*, 32(suppl_1):D267–D270, 2004.

[107] Tristan Coignion, Clément Quinton, and Romain Rouvoy. A performance study of llm-generated code on leetcode, June 2024.

[108] Juan Altmayer Pizzorno and Emery D. Berger. Coverup: Coverage-guided llm-based test generation, 2024.

[109] Danie Smit, Hanlie Smuts, Paul Louw, Julia Pielmeier, and Christina Eidelloth. The impact of github copilot on developer productivity from a software engineering body of knowledge perspective, 2024.

[110] Roberto Gozalo-Brizuela and Eduardo C. Garrido-Merchan. ChatGPT is not all you need. A State of the Art Review of large Generative AI models, 2023.

Feature	Knowledge Bases (KB)	Knowledge Graphs (KGs)	Retrieval-Augmented Generation (RAG)	Prompting Techniques
Primary Benefit	Provides static, factual data with high precision and reliability.	Enables complex reasoning and inference, emphasizing entity relationships.	Allows real-time retrieval and contextual response generation.	Enhances model response quality through structured prompts, improving clarity and relevance.
Data Structure	Structured relational format storing facts, rules, and relationships.	Graph-based structure with nodes (entities) and edges (relationships) for interconnected knowledge.	Flexible, utilizes external data sources for real-time retrieval and integration into outputs.	No structured data; relies on pre-trained model knowledge, augmented by prompt patterns and templates.
Applications	Best suited for fields requiring reliable, factual information (e.g., scientific data retrieval, expert systems, question answering, semantic search).	Ideal for applications requiring dynamic knowledge updates, such as recommendation systems, drug discovery, knowledge discovery, fraud detection.	Useful in dynamic scenarios needing up-to-date information, like customer support, content generation, question answering (single-hop/multi-hop), information extraction, dialogue generation, code search, text generation, text summarization, text classification, sentiment analysis, and math problem solving.	Effective in tasks requiring nuanced language, creativity, or simulated problem-solving, such as text generation, code generation, translation, summarization, question answering, reasoning, and chatbots.
Computational Efficiency	Relatively high efficiency; queries are fast due to structured data.	Moderate; graph traversal is efficient but can be complex in large KGs. Querying for complex relationships can be computationally intensive.	Varies; efficiency depends on retrieval model complexity and data volume.	Highly efficient; relies on model's internal processing without external retrievals, minimizing latency.
Response Accuracy	High accuracy for factual data; may struggle with queries requiring inference or reasoning beyond the explicitly stored knowledge.	High for structured and relationship-driven data, depending on KG completeness and correctness.	Moderate to high; enhanced by real-time retrieval from verified sources.	Variable; accuracy depends on prompt design and model's internal knowledge, which may be outdated.
Adaptability to Different Data Types	Low adaptability; generally static data and hard to update.	Moderate; adaptable within domain-specific contexts and relationships.	High; adapts well across domains by retrieving context-specific information.	Very adaptable for structured text tasks but limited to model's pre-trained knowledge without updates.
Domain-Specific Knowledge Suitability	Effective for fields with specialized, static knowledge requirements (e.g., medical terminology databases).	Beneficial for domains needing advanced contextual relationships (e.g., legal databases, scientific research).	Highly adaptable; ideal for any domain needing real-time, responsive data integration.	Limited by the scope of model training data; requires carefully crafted prompts for niche topics.
Strength in Reasoning	Limited; relies on static data and predefined relationships.	Strong, supports multi-hop reasoning and can infer relationships between concepts.	Limited to contextual retrieval; depends on external sources for expanded reasoning.	Provides indirect reasoning via prompt chaining and contextual prompts but lacks true inference ability.
Update Capability	Static, requiring manual updates for new data.	Flexible, can be updated frequently for real-time data (e.g., live news, ongoing medical developments).	Real-time access to external sources ensures content is current; continuous updates required for relevance.	No direct update capability; prompts use model's training data, which may be outdated.
Challenges	Limited scalability without regular updates; static data may impact performance in dynamic applications.	Data consistency issues in large-scale graphs, with a need for frequent updates.	Latency and complexity in real-time retrieval, leading to performance trade-offs under high load.	Risk of generating incorrect or irrelevant responses if prompts are ambiguous or complex; lacks external grounding.

Table 4: Comparative Analysis of Integrating LLMs with Knowledge Bases