Data Science and Engineering

https://doi.org/10.1007/s41019-025-00335-5

Article in Press

Retrieval-Augmented Generation for AI-Generated Content: A Survey

Received: 9 November 2025

Accepted: 2 December 2025

Published online: 02 January 2026

Cite this article as: Zhao P., Zhang H., Yu Q. *et al.* Retrieval-Augmented Generation for AI-Generated Content: A Survey. *Data Sci. Eng.* (2026). https://doi.org/10.1007/s41019-025-00335-5

Penghao Zhao, Hailin Zhang, Qinhan Yu, Zhengren Wang, Yunteng Geng, Fangcheng Fu, Ling Yang, Wentao Zhang, Jie Jiang & Bin Cui

We are providing an unedited version of this manuscript to give early access to its findings. Before final publication, the manuscript will undergo further editing. Please note there may be errors present which affect the content, and all legal disclaimers apply.

If this paper is publishing under a Transparent Peer Review model then Peer Review reports will publish with the final article.

ARTICLE IN PRESS

© The Author(s) 2026. **Open Access** This article is licensed under a Creative Commons Attribution-NonCommercial-NoDerivatives 4.0 International License, which permits any non-commercial use, sharing, distribution and reproduction in any medium or format, as long as you give appropriate credit to the original author(s) and the source, provide a link to the Creative Commons licence, and indicate if you modified the licensed material. You do not have permission under this licence to share adapted material derived from this article or parts of it. The images or other third party material in this article are included in the article's Creative Commons licence, unless indicated otherwise in a credit line to the material. If material is not included in the article's Creative Commons licence and your intended use is not permitted by statutory regulation or exceeds the permitted use, you will need to obtain permission directly from the copyright holder. To view a copy of this licence, visit http://creativecommons.org/licenses/by-nc-nd/4.0/.

ARTICLE IN PRESS

Retrieval-Augmented Generation for AI-Generated Content: A Survey

This Accepted Manuscript (AM) is a PDF file of the manuscript accepted for publication after peer review, when applicable, but does not reflect post-acceptance improvements, or any corrections. Use of this AM is subject to the publisher's embargo period and AM terms of use. Under no circumstances may this AM be shared or distributed under a Creative Commons or other form of open access license, nor may it be reformatted or enhanced, whether by the Author or third parties. By using this AM (for example, by accessing or downloading) you agree to abide by Springer Nature's terms of use for AM versions of subscription articles: https://www.springernature.com/gp/open-research/policies/accepted-manuscript-terms

The Version of Record (VOR) of this article, as published and maintained by the publisher, is available online at: https://doi.org/10.1007/s41019-025-00335-5. The VOR is the version of the article after copy-editing and typesetting, and connected to open research data, open protocols, and open code where available. Any supplementary information can be found on the journal website, connected to the VOR.

For research integrity purposes it is best practice to cite the published Version of Record (VOR), where available (for example, see ICMJE's guidelines on overlapping publications). Where users do not have access to the VOR, any citation must clearly indicate that the reference is to an Accepted Manuscript (AM) version.

ARTICLE IN PRESS

ARTICLE IN PRESS

Retrieval-Augmented Generation for AI-Generated Content: A Survey

Abstract

Advancements in model algorithms, the growth of foundational models, and access to high-quality datasets have propelled the evolution of Artificial Intelligence Generated Content (AIGC). Despite its notable successes, AIGC still faces hurdles such as updating knowledge, handling long-tail data, mitigating data leakage, and managing high training and inference costs. Retrieval-Augmented Generation (RAG) has recently emerged as a paradigm to address such challenges. In particular, RAG introduces the information retrieval process, which enhances the generation process by retrieving relevant objects from available data stores, leading to higher accuracy and better robustness. In this paper, we comprehensively review existing efforts that integrate RAG techniques into AIGC scenarios. We first classify RAG foundations according to how the retriever augments the generator, distilling the fundamental abstractions of the augmentation methodologies for various retrievers and generators. This unified perspective encompasses all RAG scenarios, illuminating advancements and pivotal technologies that help with potential future progress. We also summarize additional enhancements methods for RAG, facilitating effective engineering and implementation of RAG systems. Then from another view, we survey on practical applications of RAG across different modalities and tasks, offering valuable references for researchers and practitioners. Furthermore, we introduce the benchmarks for RAG, discuss the limitations of current RAG systems, and suggest potential directions for future research.

Keywords: Retrieval-Augmented Generation, AI-Generated Content, Generative Models, Information Retrieval

ARTICLE IN PRESS

1 Introduction

1.1 Background

Recent years have witnessed the surge in interests surrounding Artificial Intelligence Generated Content (AIGC). Various content generation tools have been meticulously crafted to produce diverse outputs across various modalities, such as Large Language Models (LLMs) including the GPT series [1–3] and the LLAMA series [4–6] for texts and codes, DALL-E [7–9] and Stable Diffusion [10] for images, and Sora [11] for videos. The word "AIGC" emphasizes that the contents are produced by advanced generative

ARTICLE IN PRESS

models other than human beings or rule-based approaches. These generative models have achieved remarkable performance due to the utilization of novel model algorithms, explosive scale of foundation models, and massive high-quality datasets. Specifically, sequence-to-sequence tasks have transitioned from utilizing Long Short-Term Memory (LSTM) networks [12] to Transformer-based models [13], and image-generation tasks have shifted from Generative Adversarial Networks (GANs) [14] to Latent Diffusion Models (LDMs) [10] as well. Notably, the architecture of foundation models, initially constituted by millions of parameters [15, 16], has now grown to billions or even trillions of parameters [1, 4, 17]. These advancements are further bolstered by the availability of rich, high-quality datasets [1, 18], which provide ample training samples to fully optimize model parameters.

Information retrieval is another pivotal application within the field of computer science. Different from generation, retrieval aims to locate relevant existing objects from a vast pool of resources. The most prevalent application of retrieval lies in web search engines, which primarily focus on the task of document retrieval [19, 20]. In the present era, efficient information retrieval systems can handle document collections on the order of billions [21, 22]. Besides documents, retrieval has also been applied for many other modalities [23–26].

Despite significant advancements in generative models, AIGC still grapples with challenges like outdated knowledge, lack of long-tail knowledge [27], and risks of leaking private training data [28]. Retrieval-Augmented Generation (RAG) aims to mitigate these issues with its flexible data repository [29]. The retrievable knowledge acts as non-parametric memory, which is easily updatable, accommodates extensive long-tail knowledge, and can encode confidential data. Moreover, retrieval can lower generation costs. RAG can reduce the size of large models [30], support long contexts [31], and eliminate certain generation steps [32].

A typical RAG process is depicted in Fig. 1. Given an input query, the retriever identifies relevant data sources, and the retrieved information interacts with the generator to improve the generation process. There are several *foundational paradigms* (*foundations* in short) according to how the retrieved results augment the generation: they can serve as augmented input to the generator [33, 34]; they can join at the middle stage of generation as latent representations [35, 36]; they can contribute to the final generation results in the form of logits [37, 38]; they can even influence or omit certain generation steps [32, 39]. Additionally, researchers have proposed various *enhancements* to improve the foundational RAG process. These methods encompass specific optimizations for individual components as well as holistic enhancements aimed at the entire pipeline.

In addition, while the concept of RAG initially emerged in text-to-text generation [34], this technique has also found *applications* across various domains, including codes [40–42], audios [43, 44], images [45–47], videos [48, 49], 3D [50, 51], knowledge [52–54], and AI for science [55, 56]. In particular, the essential idea and process of RAG are largely consistent across modalities. However, it necessitates minor adjustments in augmentation techniques, and the selection of retrievers and generators varies depending on the specific modalities and applications.

ARTICLE IN PRESS

Despite the rapid growth in recent research on RAG and the booming applications, a systematic review encompassing all foundations, enhancements, and applications is notably absent, hindering the development of this field. For one thing, the absence of discussion on RAG foundations significantly undermines the practical value of the research in this domain, leaving the potential of RAG not fully explored. While the majority of research interest, particularly among LLM researchers, centers on query-based RAG in text-generation tasks, it is essential to acknowledge that other RAG foundations are also effective and with significant potential for usage and further development. For another, the lack of an overview on RAG applications causes researchers and practitioners to overlook RAG's progress across multiple modalities and remain unaware of how RAG can be effectively applied. Although text generation is typically considered as the main application of RAG, we emphasize that the development of RAG in other modalities has also begun to catch on and has yielded promising advancements. Certain modalities have a rich historical connection to retrieval techniques, infusing RAG with distinctive characteristics. Inspired by this, in this paper, our objective is to present a comprehensive survey to provide a systematic overview of RAG.

1.2 Contribution

This survey offers a comprehensive overview of RAG, covering foundations, enhancements, applications, benchmarks, limitations, and potential future directions. Despite variations in retrievers and generators across modalities and tasks, we distill the core principles of RAG foundations, viewing applications as adaptations of these principles. We aim to offer references and guidelines to researchers and practitioners, providing valuable insights for advancing RAG methodologies and related applications. In summary, we list our contributions as follows:

- We conduct a comprehensive review of RAG, and distill the abstractions of RAG foundations for various retrievers and generators.
- We investigate the enhancements in the literature of RAG, elaborating the techniques leveraged to enable more effective RAG systems.
- For various modalities and tasks, we survey existing AIGC methods that incorporate RAG techniques, exhibiting how RAG contributes to current generative models.
- We discuss the limitations and promising research directions of RAG, shedding light on its potential future development.

ARTICLE IN PRESS

1.3 Related Work

As the field of RAG advances, several surveys have emerged; yet they address only specific facets of the area. In particular, they either exclusively focus on a single RAG foundation or provide only a brief overview of RAG augmentation methodologies for limited scenarios.

Most of the existing worksfocus on text-related RAG tasks that are facilitated by LLMs, without in-depth investigation in other modalities. The survey by Li et al. [57] offers a basic overview of RAG and discusses specific applications within the scope of text generation tasks. In a similar vein, the tutorial crafted by Asai et al. [58]

ARTICLE IN PRESS

centers on retrieval-based language models, detailing their structures and training strategies. Meanwhile, a recent survey by Gao et al. [59] explores RAG in the context of LLMs, with a particular emphasis on enhancement approaches for query-based RAG. Recognizing that RAG has extended beyond the text domain, our work broadens its reach to the entire AIGC landscape, facilitating a more comprehensive coverage of RAG research.

In addition, another survey proposed by Zhao et al. [60] introduces RAG applications across multiple modalities, but ignoring the discussion on RAG foundations. Another work [61] covers only part works of other modalities. While existing research has explored various aspects of RAG, there remains a need for a comprehensive overview that covers RAG foundations, enhancements, and its applicability across different domains. In this paper, we aim to address the gap and present a more systematic survey of RAG.

1.4 Roadmap

The rest of the paper is organized as follows. Section 2 elaborates on the preliminary of RAG, introducing retrievers and generators. Section 3 presents RAG foundations and further enhancements on RAG. Section 4 reviews existing research on RAG across various applications. Section 5 investigates the benchmark frameworks for RAG. Section 6 discusses current limitations of RAG and potential future directions. Finally, Section 7 concludes this paper.

2 Preliminary

In this section, we provide an overview of the general RAG architecture and explore the generators and the retrievers in today's RAG-based AIGC.

2.1 Overview

As shown in Fig. 1, the entire RAG system consists of two core modules: the retriever and the generator, where the retriever searches for relevant information from the data store and the generator produces the required contents. The RAG process unfolds as follows: (i) the retriever initially receives the input query and searches for relevant information; (ii) then, the original query and the retrieval results are fed into the generator through a specific augmentation methodology; (iii) finally, the generator produces the desired outcomes.

ARTICLE IN PRESS

2.2 Generator

The remarkable performance of generative AI across diverse tasks has ushered in the era of AIGC. The generation module plays a crucial role within the RAG system. Different generative models are applied for different scenarios, such as transformer models for text-to-text tasks, VisualGPT [62] for image-to-text tasks, Stable Diffusion [10] for text-to-image tasks, Codex [2] for text-to-code tasks, etc. As illustrated in Fig. 2, we introduce some frequently used generators below.

ARTICLE IN PRESS

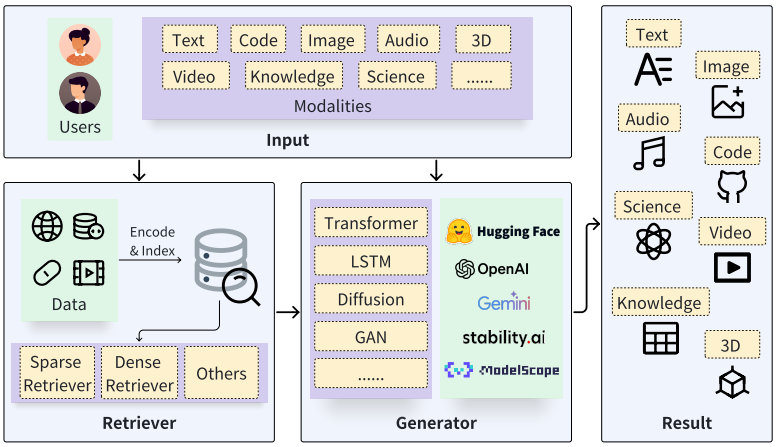

Fig. 1: A generic RAG architecture. The user queries, spanning different modalities, serve as input to both the retriever and the generator. The retriever extracts relevant information from data sources. The generator interacts with the retrieval results and ultimately produces outcomes of various modalities.

- Transformer models [63] are one of the most widely used generative models, which utilize self-attention and feed-forward networks to autoregressively generate output via vocabulary classification over latent representations.
- Long Short-Term Memory (LSTM) [64], a type of recurrent neural network, employs input, forget, and output gates along with a cell state to manage information, and similarly generates outputs autoregressively.
- Diffusion models [65] are deep generative models that create data by progressively adding noise and then reversing the process through probabilistic denoising.
- Generative Adversarial Networks (GANs) [66] comprise a generator and a discriminator trained adversarially, where the generator learns to produce realistic samples while the discriminator improves at distinguishing them from real data.

2.3 Retriever

Retrieval is to identify and obtain relevant information given an information need. Specifically, let's consider information resources that can be conceptualized as a key-value store, where each key corresponds to a value (keys and values can be identical). Given a query, the objective is to search the top-k most similar keys using a similarity function, and obtain the paired values. Based on different similarity functions, existing retrieval methods can be categorized into sparse retrieval, dense retrieval, and others. In widely used sparse and dense retrieval, the entire process can be divided into two distinct phases: (i) each object is first encoded into a specific representation; and then (ii) an index is constructed to organize the data source for efficient search.

2.3.1 Sparse Retriever

Sparse retrieval methods are commonly used in document retrieval, where the keys/-values represent the documents to be searched. These methods leverage term matching

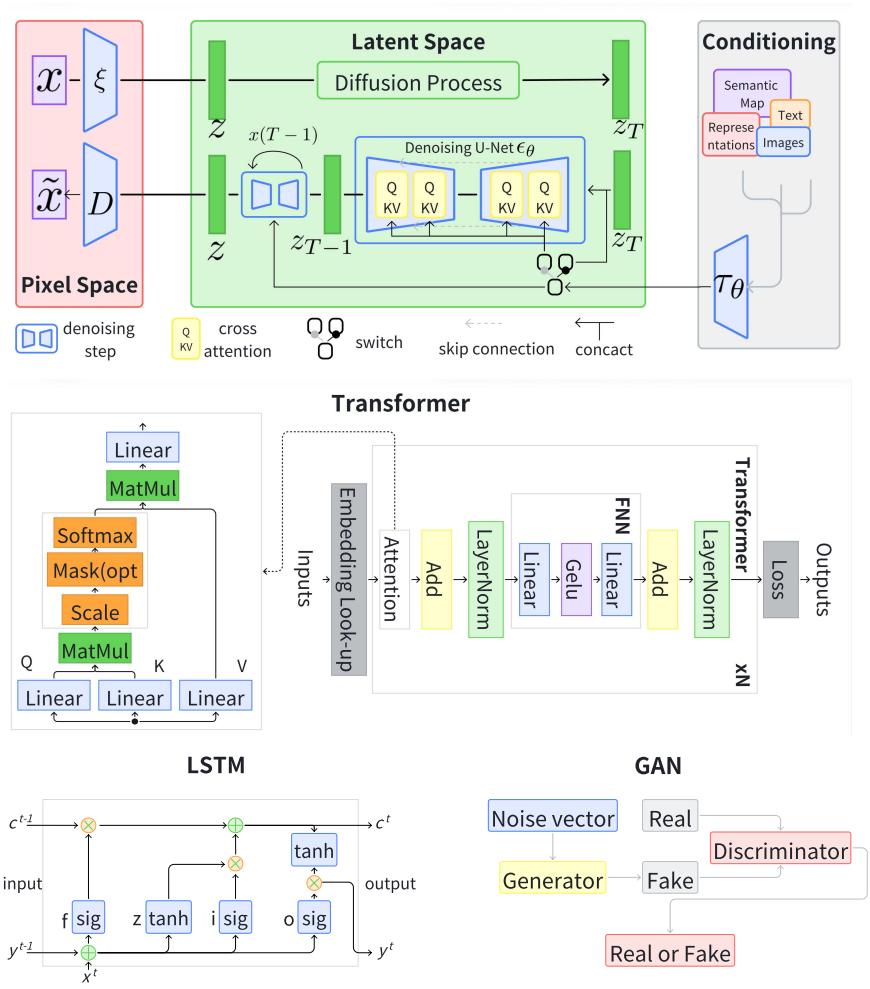

Fig. 2: General architectures of several generators.

metrics such as TF-IDF [67], query likelihood [68], and BM25 [19], which analyze word statistics from texts and construct inverted indices for efficient searching. Essentially, BM25 is a strong baseline in large-scale web search, integrating inverse document frequency weights, query token occurrences, and other pertinent metrics.

To enable efficient search, sparse retrieval typically leverages an inverted index to organize documents. Concretely, each term from the query performs a lookup to obtain a list of candidate documents, which are subsequently ranked based on their statistical scores.

ARTICLE IN PRESS

2.3.2 Dense Retriever

Unlike sparse retrieval, dense retrieval methods represent queries and keys using dense embedding vectors, and build Approximate Nearest Neighbor (ANN) index to speed up the search. This can be applied to all modalities. For text data, recent advancements in pre-trained models (such as BERT [15]) have been employed encode queries and keys individually [20]. This approach is often referred to as Dense Passage Retrieval (DPR). Similar to text, models have been proposed to encode code data [25], audio data [69], image data [24], video data [70], etc. The similarity score between dense representations are usually computed with metrics such as cosine, inner product, L2-distance.

During training, dense retrieval uses contrastive learning to increase the similarity of positive samples and decrease that of negative ones. Several hard negative techniques [71] have been proposed to further enhance model quality. For efficient searching during inference, ANN methods are employed. Various indices are developed to serve ANN search, including tree [72, 73], locality sensitive hashing [74], neighbor graph indices (e.g., HNSW [75], DiskANN [76]), and combined graph and inverted indices (e.g., SPANN [22]).

2.3.3 Others

In addition to sparse retrieval and dense retrieval, there are alternative methods for retrieving relevant objects [77, 78]. Instead of calculating representations, some research works directly use the edit distance between natural language texts [79] or abstract syntax trees (AST) of code snippets [80, 81]. In knowledge graphs, entities are connected by relations, serving as a pre-built index for retrieval. Thus, RAG methods utilizing knowledge graphs can employ k-hop neighbor searches for retrieval [82, 83]. Another retrieval method is Named Entity Recognition (NER) [84], where the query is the input and the entities act as keys.

3 Methodologies

In this section, we first introduce foundational paradigms of RAG, and then outline enhancement methods that further improve the effectiveness.

3.1 RAG Foundations

Based on how the retriever augments the generator, we categorize RAG foundations into 4 classes, as shown in Fig. 3.

3.1.1 Query-based RAG

Stemming from the idea of prompt augmentation, query-based RAG seamlessly integrates the user's query with insights from retrieved information, feeding it directly into the initial stage of the generator's input. This method is prevalent in RAG applications. Post-retrieval, the obtained content is merged with the user's original query to form a composite input, which is then processed by the generator to create a response. Query-based RAG is widely employed across various modalities.

ARTICLE IN PRESS

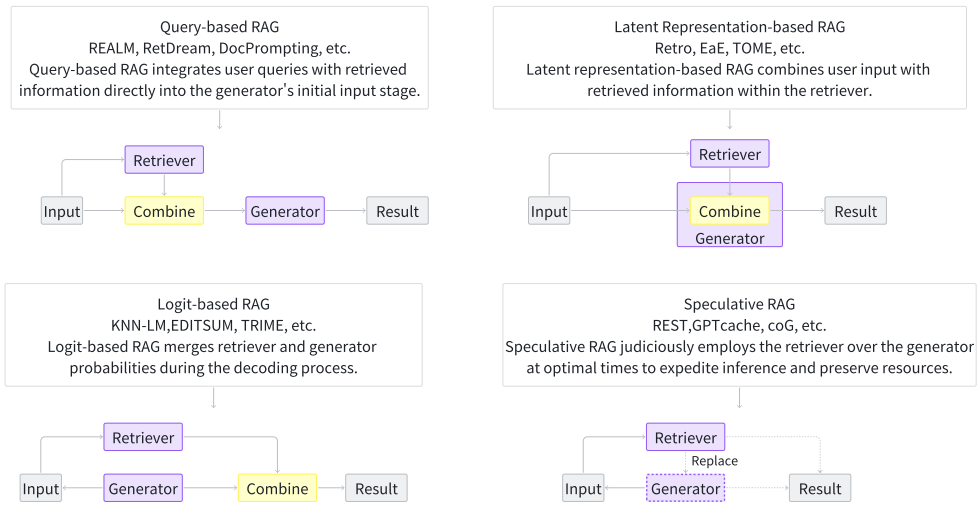

Fig. 3: Taxonomy of RAG foundations.

For text generation, REALM [33] employs a dual-BERT framework to streamline knowledge retrieval and integration, marrying pre-trained models with knowledge extractors. Lewis et al. [34] leveraged DPR for information retrieval and employs BART as the generator to effectively enhance the generation. SELF-RAG [85] utilizes a critique module to determine whether the retrieval is required. In addition to being compatible with local generators, query-based RAG is also applicable to scenarios that use LLM through API calls. REPLUG [86] follows this methodology by treating the language model as a "black box", and effectively integrates relevant external documents into the query. In-Context RALM [87] uses BM25 for document retrieval and trains a predictive reranker to reorder and integrate the top-ranked documents.

In the field of code, several works [42, 88–91] have utilized the query-based paradigm to incorporate contextual information from text or code into the prompt, resulting in improved effectiveness of downstream tasks.

Recent researches in Knowledge Base Question Answering (KBQA) has also shown significant effects of combining retrieval and language models. For instance, UniParser [92], RNG-KBQA [82], and ECBRF [93] effectively improve the performance and accuracy of QA systems by merging queries and retrieved information into prompts.

In the AI-for-Science field, Chat-Orthopedist [94] aids shared decision-making for adolescents with idiopathic scoliosis, improving LLMs' effectiveness and information precision by incorporating retrieved data into model prompts.

In the image generation task, RetrieveGAN [45] boosts the relevance and precision of generated images by incorporating retrieved data, such as selected image patches and their bounding boxes, into the generator's input stage. IC-GAN [95] modulates the specific conditions and details of the generated images by concatenating noise vectors with instance features.

ARTICLE IN PRESS

For 3D generation, RetDream [50] initially utilizes CLIP [24] to retrieve relevant 3D assets, then merges the retrieved contents with the user input during the input phase.

Query-based RAG, often paired with LLM generators, offers modular flexibility, allowing swift integration of pre-trained components for quick deployment. Prompt design is crucial for utilizing retrieved data within this setup.

3.1.2 Latent Representation-based RAG

In latent representation-based RAG framework, retrieved objects are incorporated into generative models as latent representations. This enhances the model's comprehension abilities and improves the quality of the generated content.

In the text field, FiD [35] and RETRO [36] are two classic structures of latent representation-based RAG, with many subsequent works conducting modifications based on them. FiD [35] processes each retrieved paragraph and its title alongside the query through distinct encoders, then amalgamates the resulting latent representations for decoding by a single decoder to produce the final output. RETRO [36] retrieves relevant information for each segmented sub-query, then applies a novel module termed Chunked Cross-Attention (CCA) to integrate the retrieved contents with each sub-query tokens. In addition, there are other noteworthy novel structures within the scope of latent representation-based RAG. Several studies [31, 96] have integrated k Nearest Neighbor (kNN) search within transformer blocks, allowing for input chunking and, in theory, addressing the long-criticized context length constraints of Transformer models. Kuratov et al. [97] integrated Transformer with RNN, utilizing the model's intermediate output as the content for retrieval.

In the realms of code and science, FiD has gained widespread adoption, with applications spanning various code-related fields [98–102], and AI-for-Science [55].

In the image domain, several studies [103–106] employ cross-attention mechanisms to fuse retrieval results by integrating their latent representations. Conversely, Li et al. [107] implement a text-image Affine Combination Module (ACM) that directly concatenates hidden features.

Within the knowledge domain, several studies [108–112] have adopted FiD and its derivatives for downstream tasks. EaE [113] enhances the generator's understanding through entity-specific parameterization, while TOME [114] pivots to a nuanced encoding of mentions, prioritizing the granularity of mentions over entity representations alone.

In the field of 3D generation, ReMoDiffuse [51] introduces a semantics-modulated attention mechanism which enhances the accuracy of generating corresponding 3D motions based on textual descriptions. AMD [115] achieves efficient conversion from text to 3D motion by fusing the original diffusion process with the reference diffusion process.

In the audio domain, Koizumi et al. [43] utilized an LLM, incorporating encoded dense features in the attention module to guide the generation of audio captions. Re-AudioLDM [116] utilizes distinct encoders to extract deep features from text and audio, which are then integrated into the attention mechanism of its Latent Diffusion Model (LDM).

ARTICLE IN PRESS

For video captioning, R-ConvED [48] uses a convolutional encoder-decoder network to process retrieved video-sentence pairs with an attention mechanism, generating hidden states to produce captions. CARE [117] introduces a concept detector to produce concept probabilities, and incorporates concept representations into a hybrid attention mechanism. EgoInstructor [49] uses gated-cross attention to merge text and video features, improving the relevance and coherence of captions for egocentric videos.

Latent representation-based RAG, adaptable across modalities and tasks, blends retriever and generator hidden states but requires additional training for aligning latent spaces. It enables the development of sophisticated algorithms that seamlessly incorporate retrieved information.

3.1.3 Logit-based RAG

In logit-based RAG, generative models integrate retrieval information through logits during the decoding process. Typically, the logits are combined through simple summation or models to compute the probabilities for step-wise generation.

In the text domain, kNN-LM [37] and its variant [38] blend language model probabilities with those from retrieval distances of similar prefixes at each decoding step. TRIME [118] and NPM [119] are radical evolutions of traditional kNN-LM approaches, using closely aligned tokens from a local database as output, particularly boosting performance in long-tail distribution scenarios.

Beyond text, other modalities, such as code and image, also leverage logit-based RAG. In the domain of code, several studies [80, 120] have also adopted the concept kNN to enhance final output control, thereby achieving superior performance. Furthermore, EDITSUM [98] improves the quality of code summarization by integrating prototype summaries at the logit level. For image captioning, MA [121] directly applies the kNN-LM framework to address the image caption problem, achieving favorable results.

In summary, logit-based RAG utilizes historical data to deduce current states and merges information at the logit level, ideal for sequence generation. It focuses on generator training and allows for novel methods that capitalize on probability distributions for future tasks.

3.1.4 Speculative RAG

Speculative RAG seeks opportunities to use retrieval instead of pure generation, aiming to save resources and accelerate response speed. REST [32] replaces the small models in speculative decoding [122] with retrieval, enabling the generation of drafts. GPTCache [39] addresses the issue of high latency when using the LLM APIs by building a semantic cache for storing LLM responses. COG [123] decomposes the text generation process into a series of copy-and-paste operations, retrieving words or phrases from the documents instead of generation. Cao et al. [124] proposed a new paradigm to eliminate the dependence of the final result on the quality of the first-stage retrieved content, replacing generation with directly retrieved phrase level content.

ARTICLE IN PRESS

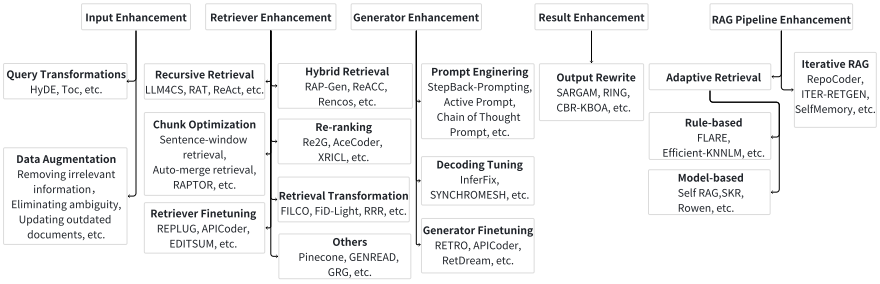

Fig. 4: Taxonomy of RAG Enhancements.

In conclusion, speculative RAG is currently primarily applicable to sequential data. It decouples the generator and the retriever, enabling the direct use of pre-trained models as components. Within this paradigm, we can explore a wider range of strategies to effectively utilize the retrieved content.

3.2 RAG Enhancements

In this section, we introduce methods which enhance the performance of a constructed RAG system. We categorize existing methods into 5 groups based on their enhancement targets: input, retriever, generator, result, and the entire pipeline.

3.2.1 Input Enhancement

The input, initially fed into the retriever, significantly impacts the final outcome of the retrieval stage. In this section, we introduce two methods for input enhancement: query transformation and data augmentation.

Query Transformation: Query transformation can enhance the result of retrieval by modifying the input query. Query2doc [125] and HyDE [126] use the original query to generate a pseudo document, which is later used as the query for retrieval. The pseudo document contains richer relevant information, which helps to retrieve more accurate results. TOC [127] leverages retrieved contents to decompose the ambiguous query into multiple clear sub-queries, which are sent to the generator and aggregated to produce the final result. For complex or ambiguous queries, RQ-RAG [128] breaks them down into clear subqueries for fine-grained retrieval and synthesizes the responses to deliver a cohesive answer to the original query. Tayal et al. [129] refined the initial query using dynamic few-shot examples and context retrieval, enhancing the generator's grasp of user intent.

Data Augmentation: Data augmentation improves data before retrieval, including techniques such as removing irrelevant information, eliminating ambiguity, updating outdated documents, synthesize new data, etc. Make-An-Audio [44] uses captioning and audio-text retrieval to generate captions for language-free audio to mitigate data sparsity, and adds random concept audio to improve the original audio. LESS [130] optimizes dataset selection for downstream tasks by analyzing gradient information, aiming to enhance model performance in response to instructional prompts. ReACC [91] employs data augmentation (including renaming and dead code insertion)

ARTICLE IN PRESS

to pre-train the code retrieval model. Telco-RAG [131] enhances the retrieve accuracy by appling a "Vocabulary for 3GPP Specifications", and match them to user queries with a router module.

3.2.2 Retriever Enhancement

In RAG systems, the quality of retrieved content determines the information fed into the generators. Lower content quality increases the risk of model hallucinations [132] or other degradation. In this section, we introduce efficient ways to enhance retrieval effectiveness.

Recursive Retrieval: Recursive retrieval is to perform multiple searches to retrieve richer and higher-quality contents. ReACT [133] uses Chain-of-Thought (CoT) [134] to break queries down for recursive retrieval and provide richer information. RATP [135] uses the Monte-Carlo Tree Search for simulations to select optimal retrieval content, which is then templated and forwarded to the generator for output.

Chunk Optimization: Chunk optimization refers to adjusting chunk size for improved retrieval results. LlamaIndex [136] incorporates a series of chunk optimization methods, one of which operates on a 'small to big' principle. The core concept here is to pinpoint finer-grained content but return richer information. For instance, Sentence-window retrieval fetches small text chunks and returns a window of relevant sentences surrounding the retrieved segment. In auto-merge retrieval, documents are arranged in a tree structure. The process retrieves the parent node, which encapsulates the content of its child nodes, by fetching the child node first. To address the lack of contextual information, RAPTOR [137] employs recursive embedding, clustering, and summarization of text chunks until further clustering becomes infeasible, thereby constructing a multi-level tree structure. Prompt-RAG [138] enhances retrieval accuracy by pre-generating a table of contents, enabling the model to autonomously select relevant chapters based on the query. Raina et al. [139] break text chunks into finer atomic statements to achieve higher recall and improved results. MoM [140] trains a model to perform document chunking, which enables finer-grained semantic overlap and facilitates subsequent retrieval.

Retriever Finetuning: The retriever, central to the RAG system, relies on a proficient embedding model [141–144] to represent related content and feed the generator, enhancing system performance. Additionally, embedding models with strong expressive power can be fine-tuned with domain-specific or task-related data to boost performance in targeted areas. REPLUG [86] treats LM as a black box and update the retriever model based on the final results. APICoder [88] finetunes the retriever with python files and api names, signature, description. EDITSUM [98] finetunes the retriever to decrease the jaccard distance between summaries after retrieval. Synchromesh [81] adds tree distance os ASTs in the loss and uses Target Similarity Tuning (TST) to finetune the retriever. R-ConvED [48] finetunes the retriever with the same data as generator. Kulkarni et al. [145] applied infoNCE loss to finetune the retriever.

Hybrid Retrieval: Hybrid retrieve denotes the concurrent employment of a diverse array of retrieval methodologies or the extraction of information from multiple distinct sources. RAP-Gen [146], BlendedRAG [147]and ReACC [91] use both dense retriever and sparse retriever to improve the quality of retrieval. Rencos [80] uses sparse

ARTICLE IN PRESS

retriever to retrieve similar code snippets on syntactic-level and uses dense retriever to retrieve similar code snippets on semantic-level. Bashexplainer [99] first uses dense retriever to capture semantic information and then uses sparse retriever to acquire lexical information. RetDream [50] first retrieves with text and then retrieves with the image embedding. CRAG [148] features a retrieval evaluator that gauges document relevance to queries, prompting three retrieval responses based on confidence: direct use of results for Knowledge Refinement if accurate, Web Search if incorrect, and a hybrid approach for ambiguous cases. Huang et al. [149] improved question-answering by introducing DKS (Dense Knowledge Similarity) and RAC (Retriever as Answer Classifier) in the retrieval phase, evaluating answer relevance and knowledge applicability. UniMS-RAG [150] introduces a novel kind of token, termed as the "acting token", which determines the source from which to retrieve information. Koley et al. [151] enhance image retrieval by integrating sketch and text for fine-grained retrieval, yielding improved results.

Re-ranking: The Rerank technique refers to reordering the retrieved content in order to achieve greater diversity and better results. Re2G [152] applies a re-ranker [153] model after the traditional retriever to reduce the impact of information loss caused by compressing text into vectors. AceCoder [154] reranks the retrieved programs with a selector to reduce redundant programs and obtain diverse retrieved programs. XRICL [155] uses a distillation-based exemplar reranker after retrieval. Rangan [156] employs the Quantized Influence Measure, assessing statistical biases between a query and a reference to evaluate the similarity of data subsets and rerank retrieval results. UDAPDR [157] uses LLMs to cost-effectively generate synthetic queries that train domain-specific rerankers, which then apply multi-teacher knowledge distillation to develop a cohesive retriever. LLM-R [158] refines its retriever iteratively by employing a static LLM for document ranking and reward model training, complemented by knowledge distillation. Each training cycle incrementally improves the retriever, enabling progressive optimization. Finardi et al. [159] integrated reciprocal rank into the retrieval process for enhanced text chunk relevance, and utilized monoT5 as a reranker to optimize the result quality. Li et al. [160] integrate a reranking module into their end-to-end RAG system, enhancing the retrieval quality and factual accuracy of LLMs. EBCAR [161] reranks passages by leveraging a transformer-based model to score them. This model goes beyond pairwise similarity by evaluating the global relationships among all passages and performing integrated reasoning to determine the final order.

Retrieval Transformation: Retrieval Transformation involves rephrasing retrieved content to better activate the generator's potential, resulting in improved output. FILCO [162] efficiently purges extraneous material from retrieved text, isolating only the pertinent supporting content to streamline the generator's task and facilitate accurate answer prediction. FiD-Light [163] initially employs an encoder to convert the retrieved content into a vector, which it then compresses, resulting in a substantial reduction of latency time. RRR [164] integrates the current query with the top-k document in each round through a template, and subsequently restructures it via a pre-trained LLMs (GPT-3.5-Turbo etc.).

ARTICLE IN PRESS

Others: In addition to the above optimization methods, there are also some other optimization methods for the retrieve process. For example, meta-data filtering [165] is a method to help processing retrieved documents which uses metadata (such as time, purpose, etc.) to filter the retrieved documents for better results. GENREAD [166] and GRG [167] introduce a novel approach where the retrieval process is supplanted or improved by prompting a LLM to generate documents in response to a given question. Multi-Head-RAG [168] employs multiple embedding models to project the same text chunk into various vector spaces and utilizes a multi-head attention layer to capture different informational aspects, thereby increasing the accuracy of the retrieval process.

3.2.3 Generator Enhancement

In RAG systems, the quality of the generator often determines the quality of the final output results. Therefore, the ability of the generator determines the upper limit of the entire RAG system's effectiveness.

Prompt Engineering: Technologies in prompt engineering [169] that focus on improving the quality of LLMs' output, such as prompt compression, Stepback Prompt [170], Active Prompt [171], Chain of Thought Prompt [134], etc., are all applicable to LLM generators in RAG systems. LLMLingua [172] applies a small model to compresses the overall length of the query to accelerate model inference, relieving the negative impact of irrelevant information on the model and alleviating the phenomenon of "Lost in the Middle"[173]. ReMoDiffuse [51] decomposes complex descriptions into anatomical text scripts by using ChatGPT. ASAP [174] incorporates exemplar tuples, consisting of input code, function definitions, analysis results, and corresponding comments, into prompts to yield better results. CEDAR [89] uses a designed prompt template to organize code demonstration, query, and natural language instructions into a prompt. XRICL [155] utilizes COT technology to add translation pairs as an intermediate step in cross linguistic semantic parsing and inference. ACTIVERAG [175] employs the Cognition Nexus mechanism to calibrate the intrinsic cognition of LLMs and applies COT prompt in answer generation. Make-An-Audio [44] is able to use other modalities as input which can provide much richer information for the following process.

Decoding Tuning: Decoding tuning involves enhancing generator control by fine-tuning hyperparameters for increased diversity and constraining the output vocabulary, among other adjustments. InferFix [90] balances the diversity and quality of results by adjusting the temperature in decoder. Synchromesh [81] limits the output vocabulary of the decoder by implementing a completion engine to eliminate implementation errors.

Generator Finetuning: The finetuning of the generator can enhance the model's ability to have more precise domain knowledge or better fit with the retriever. RETRO [36] fixes the parameters of the retriever and uses the chunked cross attention mechanism in the generator to combine the content of the query and retriever. APICoder [88] finetunes the generator CODEGEN-MONO 350M [176] with a shuffled new file combined with API information and code blocks. CARE [117] trains encoders with image, audio, and video-text pairs, then fine-tunes the decoder (generator) to simultaneously reduce caption and concept detection loss, while keeping the encoders

ARTICLE IN PRESS

and retriever fixed. Animate-A-Story [177] optimizes the video generator with image data, and then finetunes a LoRA [178] adapter to capture the appearance details of the given character. RetDream [50] finetunes a LoRA adapter [178] with the rendered images.

3.2.4 Result Enhancement

In many scenarios, the result of RAG may not achieve the expected effect, and some techniques of Result Enhancement can help alleviate this problem.

Output Rewrite: Output Rewrite refers to rewriting the content generated by the generator in certain scenarios to meet the needs of downstream tasks. SARGAM [179] refines outputs in code-related tasks by employing a special Transformer alongside Deletion, Placeholder, and Insertion Classifiers to better align with the real-world code context. Ring [180] obtains diversity results by reranking candidates based on the average of per token log probabilities produced by the generator. CBR-KBQA [54] revises the result by aligning generated relations with those presented in the local neighborhood of the query entity in knowledge graph.

3.2.5 RAG Pipeline Enhancement

RAG pipeline enhancement refers to optimizing the overall process of RAG in order to achieve better performance results.

Adaptive Retrieval: Some studies on RAG suggest that retrieval doesn't always enhance the final results. Over-retrieval can lead to resource wastage and potential confusion when the model's inherent parameterized knowledge suffices for answering relevant questions. Consequently, this chapter will delve into two methods for determining retrieval necessity: rule-based and model-based approaches.

Rule-based: FLARE [181] actively decides whether and when to search through the probability in the generation process. Efficient-KNNLM [38] combines the generation probability of KNN-LM [37] and NPM [119] with a hyperparameter λ to determine the proportion of generation and retrieval. Mallen et al. [182] used statistical analysis on questions to enable direct answers for high-frequency ones and applied RAG for low-frequency ones. Jiang et al. [183] evaluated model confidence based on Model Uncertainty, Input Uncertainty, and Input Statistics to guide retrieval decisions. Kandpal et al. [184] studied the correlation between the number of relevant documents and the model's knowledge mastery to assess the need for retrieval.

Model-based: Self-RAG [85] uses a trained generator to determine whether to perform a retrieval based on the retrieve token under different user querys. Ren et al. [185] used "Judgment Prompting" to determine whether LLMs can answer relevant questions and whether their answers are correct or not, thereby assisting in determining the necessity of a retrieval. SKR [186] uses the ability of LLMs themselves to judge in advance whether they can answer the question, and if they can answer, no retrieval is performed. Rowen [187] translates a question into multiple languages and checks for answer consistency across these languages, using the results to determine the need for information retrieval. AdaptiveRAG [188] dynamically decides whether to retrieve based on the query complexity by a classifier, which is a smaller LM.

ARTICLE IN PRESS

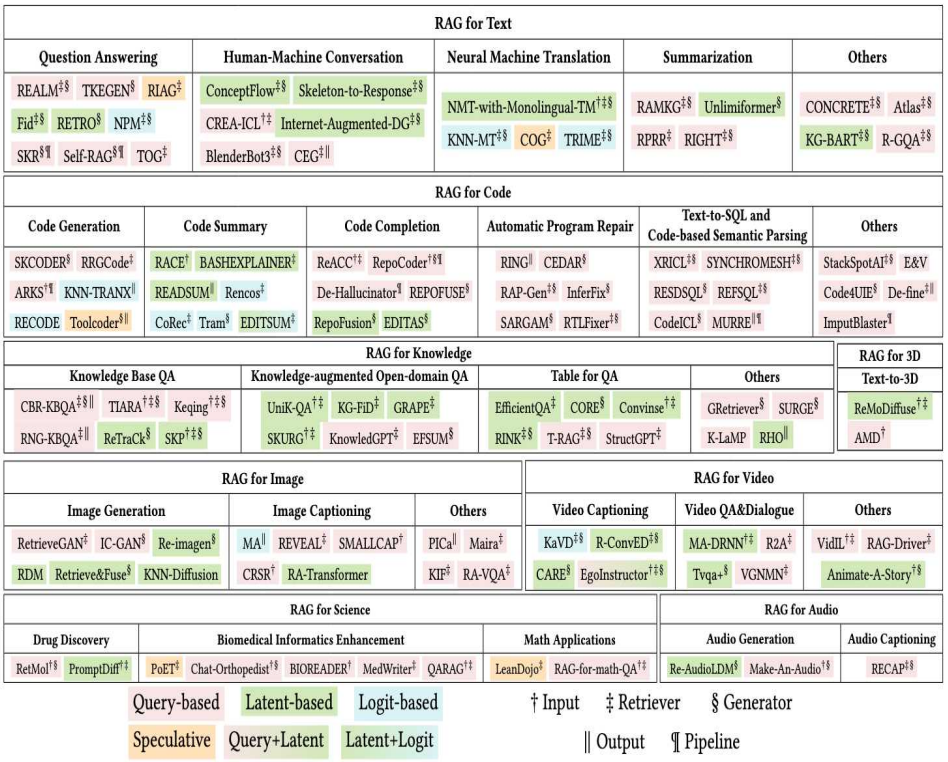

Fig. 5: Taxonomy of RAG applications across various modalities.

Iterative RAG: Iterative RAG progressively refines results by repeatedly cycling through retrieval and generation phases, rather than a single round. RepoCoder [189] uses an iterative retrieval-generation approach for code completion, refining queries with previously generated code to better utilize dispersed information and improve outcomes. ITER-RETGEN [190] iteratively enhances content quality by using the generator's output to pinpoint knowledge gaps, retrieving necessary information, and informing future generation cycles. SelfMemory [191] utilizes a retrieval-augmented generator iteratively to form an expansive memory pool, from which a memory selector picks an output to inform the next generation cycle. RAT [192] initially generates content by an LLM with a zero-shot CoT prompt, then revises each thought step by retrieving knowledge from external knowledge base.

4 Applications

In this section, we focus on RAG applications spanning various modalities. To echo with the taxonomy of RAG foundations and enhancements, we also demonstrate their utilization across different tasks in Fig. 5.

ARTICLE IN PRESS

From the perspective of RAG foundations, query-based and latent-based RAG paradigms are the most widely applied across various modalities due to their convenience and ease of use. Logit-based RAG, on the other hand, is similar to distillation and more readily facilitates the alignment of texts and other domains (e.g., image and video captioning). Speculative-based RAG is primarily utilized to boost the overall efficiency when the retrieved contents themselves can serve as the answer (e.g., question answering and code generation).

In the dimension of enhancement, Input Enhancement has been widely adopted across various modalities due to its ease of implementation(e.g., Image Genration, Video Genration and Audio Generation). Retriever Enhancement and Generator Enhancement, conversely, necessitate the selection of specific model architectures and retrieval methods tailored to the unique characteristics of different modalities and tasks. Result Enhancement is a relatively generic step designed to ensure that the final output generated by the model aligns more closely with the specific requirements of the task.(e.g., Code Summary and Image Captioning) Pipeline Enhancement focuses primarily on overall system efficiency and performance metrics, aiming for the entire system to complete user requests more effectively and at a lower operational cost(e.g., Question Answering and Code Completion).

In the following, we will gradually elaborate the specific applications within each modality.

4.1 RAG for Text

To begin with, text generation is among the most important and widely deployed applications for RAG. Here we introduce popular works for seven tasks, respectively.

4.1.1 Question Answering

Question answering involves the process of providing responses to posed questions by drawing from a vast and comprehensive collection of textual sources. FiD [35] and REALM [33] identify the top-k most pertinent article snippets based on the query and forward each snippet along with the question to LLMs to generate k responses. These responses are then synthesized into a final answer. Toutanova et al. [193] substituted the text corpus in REALM with subgraphs from a knowledge graph, yielding impressive results. RETRO [36] employs attention mechanisms to integrate the question with relevant retrieved documents within the model to produce the final answer. SKR [186] observes that using RAG does not invariably benefit question answering and thus explored guiding the model to evaluate its grasp of pertinent knowledge, subsequently adapting its use of external resources for retrieval enhancement. TOG [194] introduces an innovative knowledge graph-augmented LLM framework, which excels by fostering interactions between LLMs and the knowledge graph and by expanding the inference path space with beam search. NPM [119] pioneers the use of nonparametric data distributions in lieu of the softmax layer, enabling models with fewer parameters to perform effectively. CL-ReLKT [195] employs a language-generalized encoder to bridge the gap between question-document pairs across languages, thus better leveraging multilingual data. CORE [196] mitigates language resource disparities

ARTICLE IN PRESS

by introducing a novel dense passage retrieval algorithm and a multilingual autoregressive generation model. Lastly, EAE [113] enhances answer quality by retrieving entity embeddings for query entities and integrating these with hidden states for further processing. UR-QA [197] proposes to simultaneously retrieve QA pairs and text chunks, selecting the final answer by comparing their calibrated confidences. DISC-LawLLM [198] constructs a supervised fine-tuning dataset through a legal syllogism prompting strategy, enabling the model to receive support from the latest legal information. RAG-end2end [199] conducts simultaneous training of the retriever (DPR) and the generator (BART) to optimize performance for the end-to-end question-answering task and to facilitate domain adaptation. MultiHop-RAG [200] extracts and aggregates information from distinct documents, providing the generator with the necessary context for definitive query answers.

4.1.2 Fact Verification

Fact verification typically refers to determining whether a given natural language text and a related claim or assertion match the facts in the text. CONCRETE [201] leverages cross-lingual retrieval mechanisms to tap into a wealth of multilingual evidence, effectively bridging the gap in resources for languages that are underrepresented in fact-checking datasets. Atlas [30] shows that using RAG to support LLMs in knowledge-intensive tasks markedly improves their few-shot learning performance. Hagström et al. [202] proved on LLaMA [4] and Atlas [30] that search augmentation is more beneficial for solving inconsistency problems than increasing model size. Stochastic RAG [203] employs stochastic sampling without replacement to address the non-differentiable topk selection process in RAG retrieval, enabling end-to-end optimization and achieving excellent results in fact verification scenarios.

4.1.3 Commonsense Reasoning

Commonsense reasoning entails the capability of machines to infer or make decisions on problems or tasks in a human-like manner, drawing upon their acquired external knowledge and its application. KG-BART [204] expands the conceptual landscape by incorporating intricate interrelations among diverse concepts within a knowledge graph. It employs graph attention mechanisms to aid LLMs in crafting more nuanced and logically coherent sentences. Wan et al. [205] constructed the CONFLICTINGQA dataset with contentious questions and conflicting answers to study how textual features affect LMs' handling of controversial issues.

4.1.4 Human-Machine Conversation

Human-machine conversation encompasses the ability of machines to comprehend natural language and adeptly employ this skill to engage with humans seamlessly. ConceptFlow [206] leverages a commonsense knowledge graph to structure conversations, directing the flow of dialogue based on attention scores, and propelling the conversation forward. Cai et al. [207] reimagined the text generation task as a cloze test by retrieving and distilling the essence of past conversational history, leading to notable outcomes. Komeili et al. [208] augmented dialogue generation quality by

ARTICLE IN PRESS

harnessing advanced search engine technologies to source pertinent content from the internet. BlenderBot3 [209] broadens its search horizon, not only mining relevant internet content but also local dialogue history, and employs entity extraction among other techniques to refine the quality of the resulting dialogue. Kim et al. [210], PARC [211], and CREA-ICL [212] improve the caliber of non-English conversations by incorporating cross-lingual knowledge, effectively addressing the scarcity of non-English datasets and enhancing the quality of the generated dialogue. CEG [213] addresses hallucination issues through a post-processing mechanism, verifying LLM-generated answers through retrieval.

4.1.5 Neural Machine Translation

Neural Machine Translation (NMT) is the automated process of translating text from a source language to a target language [118, 214, 215]. It is a pivotal task in the domain of NLP and represents a significant objective in the pursuit of AI, boasting considerable scientific and practical significance. Cai et al. [214] proposed an innovative approach that utilizes monolingual corpora alongside multilingual learning techniques, challenging the traditional dependency on bilingual corpora in Neural Machine Translation. kNN-MT [215] executes translation tasks at the token level by computing vector space distances. TRIME [118] effectively minimizes the discrepancy between training and inference phases by jointly training the retrieval system and the generation model, thereby enhancing the precision of translations.

4.1.6 Event Extraction

Event extraction is a process in NLP that involves identifying and categorizing specific events within a text and associating them with relevant entities. These events are usually represented by verbs and the entities are the participants involved in the event. R-GQA [216] enhances the context of a given issue by identifying and utilizing the most closely aligned Question-Answer pair from a repository, thereby enriching the information available for processing the current query.

4.1.7 Summarization

Summarization is a task aimed at distilling the essential information from lengthy texts and producing a concise, coherent summary that encapsulates the primary themes. There are two main approaches to summarization: extractive and abstractive. Extractive summarization involves the automatic selection and compilation of key phrases directly from the source text, which refrains from creating new sentences, instead repurposing segments from the original text. Abstractive summarization, on the other hand, entails comprehending the original text's meaning and reformulating it into new sentences [96, 217–219], which can convey the source's intent more fluidly but poses greater challenges in terms of implementation due to its complexity. RAMKG [217] effectively leverages a comprehensive English corpus to bolster the performance of keyphrase generation in non-English contexts. Unlimiformer [96] addresses the issue of input length constraints in transformer-based models by retrieving and utilizing the top-k most relevant hidden states, thereby extending the model's

ARTICLE IN PRESS

capacity to handle longer inputs. RPRR [218] employs a Retrieve-Plan-Retrieve-Read approach to overcome the limited context window constraints faced by LLMs, utilizing retrieved information to generate high-quality Wikipedia documents for emerging events. RIGHT [219] chooses to use different types of retrievers in different datasets to enhance the generator. M-RAG [220] significantly enhances text summarization by segmenting documents into various databases and incorporating multi-agent reinforcement learning techniques.

4.2 RAG for Code

Separate retrieval and generation approaches have historically been employed for code-related tasks. For retrieval, similar code snippets can be identified using Abstract Syntax Trees (AST) or text edit distance. For generation, sequence-to-sequence models are employed to generate code or natural language. Recent RAG research combines both retrieval and generation techniques to enhance the overall performance.

4.2.1 Code Generation

Code generation aims to convert Natural Language (NL) descriptions into code implementations.

Query-based RAG is a common method for code generation. It builds prompts for transformer-based generative models with retrieved information, including similar examples [40, 154, 221–224], relevant API details [88, 225], documentations [42], imports [226], and global functions [227]. SKCODER [228] retrieves relevant code snippets to produce sketch template for final code generation. RRGCode [229] employs a cross-encoder to rank the retrieval results. CODEAGENT [230] designs agents for web search, documentation retrieval, program generation, and correctness testing. ARKS [231] incorporates iterative RAG to re-formulate queries and update retrieval sources.

Logit-based RAG is also applicable for code generation. RECODE [79] retrieves NL descriptions and paired codes using edit distance, then extracts n-gram action subtrees from ASTs. During LSTM-based generation, the processed subtrees are leveraged through logits at each decoding step. kNN-TRANX [120] uses a seq2tree model to convert NL to code AST. During each decoding step, hidden states are searched in the AST prefix datastore to create new probabilities, later merged with the seq2tree model's output via a confidence network.

ToolCoder [232] generates codes containing special tokens. When it encounters these tokens, ToolCoder performs online search or offline retrievals to fill in the blanks with API calls, which is a specialized form of speculative RAG.

4.2.2 Code Summarization

Code summarization tasks in turn convert the code into NL descriptions.

Many research works process retrieval results using additional encoders and then combine them for subsequent decoder, which is similar to the Fusion-in-Decoder [35].

Re2Com [101] and EditSum [98] retrieve similar codes using BM25 and generate summary using LSTM. They separately encode the input, the retrieved code, and

ARTICLE IN PRESS

the corresponding summary, then combine the hidden states or logits in the decoder. HGNN [233] instead uses code edit distance for retrieval, and substitutes the code encoder with hybrid GNN on their Code Property Graphs (CPG) [234]. RACE [102] employs separate encoders for the input code difference, the retrieved code differences through dense retrieval, and corresponding commit message to generate the final commit messages. Bashexplainer [99] applies dense retrieval, and fuses the embeddings for subsequent transformer-based decoder. READSUM [235] uses Levenshtein distance for retrieval, and employs a fusion network to combine the representations of retrieved codes and summaries.

Query-based RAG is prevalent for code summary generation. REDCODER [40], ASAP [174], and SCCLLM [236] all form prompts with retrieved contents for summarization. They employ dense retrieval, sparse retrieval, and hybrid retrieval (including semantic, syntactic, and lexical-based retrieval), respectively. The paradigm is also leveraged for pseudocode generation [237] and log statement generation [238].

Logit-based RAG also prevails in code summarization. Rencos [80] and CoRec [239] retrieve similar code snippets or code differences through AST or dense representations. They both adopt multiple LSTMs for the input and the retrieved results, and the probabilities are combined for final generation. kNN-Transformer [240] uses a transformer-based generator to obtain context vectors of input codes, then combines three parts of logits from vector search, the generator, and the copy mechanism for rare tokens in the input. Tram [241] also combines three sets of logits from the original generator, the generator for sentence-level retrieved results, and the search logits of the token-level vectors (which represent the source codes and their ASTs). CMR-Sum [242] incorporates the cross-attention probabilities between the retrieved summary and the generated summary, to the original generation logits.

4.2.3 Code Completion

Code completion is akin to the code version of the "next sentence prediction" task.

Query-based RAG is the mainstream paradigm for code completion. Drain et al. [243] retrieved template functions for function completion. ReACC [91] uses both sparse and dense retrieval. RepoCoder [189] performs iterative RAG by augmenting the retrieval input with previously generated code. De-Hallucinator [244] retrieves API references using first-time generated contents, then conducts query-based RAG for improved code completion. REPOFUSE [245] includes rationale context and retrieved codes to form prompt, and ranks the contexts to fit in the length limit.

Many works leverage latent representation-based RAG. Retrieve-and-edit [100], RepoFusion [246], and EDITAS [247] employ multiple encoders for retrieved contents or edit sequences, then fuse the information for subsequent decoder. CoCoMic [248] retrieves codes on the project context graph of the whole code project. It jointly processes the representations of source codes and retrieved contexts in the generator.

kNM-LM [249] performs logit-based RAG, combining the logits of retrieval and generation using bayes inference.

ARTICLE IN PRESS

4.2.4 Automatic Program Repair

Query-based RAG is often used in automatic program repair to help generative models fix buggy codes. RING [180], CEDAR [89], and RAP-Gen [146] all use hybrid retrieval (including both sparse and dense retrieval) for similar error messages, buggy codes, or fixes to build prompts. InferFix [90] includes the bug type, the location, relevant syntax hierarchies, and similar fixes into the prompt. SARGAM [179] utilizes prompts with similar buggy codes to generate patches; then another model is employed to refine the final result. RTLFixer [250] leverages ReAct [133] to implement an agent fixing errors in Verilog codes. It iteratively retrieves errors and paired solutions, and combines reasoning and action planning into prompts for LLMs.

4.2.5 Text-to-SQL and Code-based Semantic Parsing

Semantic parsing converts NL into clear, structured representations, like SQL or other domain-specific languages, often with the assistance of codes. All related works that employ RAG specifically utilize its query-based variant. XRICL [155] searches and reranks English utterance using non-English ones, then builds prompt to generate SQL queries. Synchromesh [81] retrieves similar NL and SQL to build prompts, then conducts constrained semantic decoding to enforce rich syntactic and semantic constraints during SQL generation. CodeICL [251] uses Python for semantic parsing, leveraging BM25 to incorporate similar training examples into prompts. Resdsql [252] includes ranked schemas into prompts to generate SQL skeleton and SQL query. Refsql [253] uses a structure-enhanced retriever with schema linking and Mahalanobis contrastive learning, which helps to make better text-to-SQL generation. To build prompts for SQL generation, ODIS [254] retrieves both in-domain and out-of-domain demonstrations, while Nan et al. [255] retrieved both similar and diverse demonstrations. MURRE [256] conducts multi-hop retrieve-rewrite on tables to generate tabularized question, then ranks the results for prompt construction. CodeS [257] retrieves relevant information from table databases in a coarse-to-fine manner to generate SQL.

4.2.6 Others

There are several other code-related tasks that adopt query-based RAG paradigm, incorporating similar examples to construct prompts. Jie et al. [258] used programs as the intermediate step in numerical reasoning. De-fine [259] uses programs to solve complex tasks. It refines the answer generated by query-based RAG, then adds the refined programs back to the retrieval source. For program static analysis, E&V [260] leverages an LLM agent to form intermediate results with AST-based source code retrieval, pseudo-code execution, execution specifications verification, and other tools. Code4UIE [261] performs information extraction through code representation. StackSpotAI [262] builds an AI coding assistant with an RAG component. InputBlaster [263] generates unusual text input that could cause mobile app crash.

4.3 RAG for Knowledge

Structured knowledge, including KGs (Knowledge Graph) and tables, is widely used in language-related tasks. It usually serves as the retrieval source to augment generation.

ARTICLE IN PRESS

In addition to regular sparse and dense retrieval, NER (Named-Entity Recognition) technique and graph-aware neighbor retrieval are applied to identify and extract relevant entities and relations.

4.3.1 Knowledge Base Question Answering

KBQA (knowledge base question answering) typically utilizes a knowledge base to determine the correct answer to a question. Many semantic parsing methods have been proposed, generating logical forms (e.g. SPARQL) based on the question.

Query-based RAG is the mainstream approach. Unseen Entity Handling [53] uses FreeBase [264] to retrieve topic entities, which are combined with query to generate SPARQL output. CBR-KBQA [54] combines the query and the retrieved (query, logical form) pairs for generation. It also revises the final result to align with the relations present in the knowledge graph. GMT-KBQA [52] re-ranks the retrieved entities and relations, and conducts relation classification and entity disambiguation before generation. RNG-KBQA [82], TIARA [83], BLLM augmentation [265], and Shu et al. [266] re-rank the candidate logical forms or entities from the knowledge graph for prompt construction. Uni-Parser [92] includes entities from mention detection, 2-hop paths extraction, and tables from databases into generator input. ECBRF [93] follows the case-based reasoning paradigm [267], retrieving similar triplet to build prompt input. FC-KBQA [268] extracts relevant classes, relations, and entities from BM25 or mention detection, StructGPT [269] extracts relevant triplets and nearest entities, and KAPING [270] extracts relevant facts through entity matching. Sen et al. [271] replaced the retrieval with a relation distribution generation model for weighted triplets. Retrieve-Rewrite-Answer [272] retrieves subgraphs into prompts using hop prediction, relation path prediction, and triplet sampling. Keqing [273] decomposes a complex question into simple sub-questions through LLM, then retrieves sub-question templates and extract candidate entities from knowledge graph, and finally generates the answer through ChatGPT. Liu et al. [274] leveraged retrieved pairs to explore the capability of formal language understanding and generation. Interactive-KBQA [275] employs the LLM as an agent, which conducts entity-linking on KG and generates current thought and action until obtaining the final answer.

Latent representation-based RAG is also employed for KBQA. ReTraCk [276] retrieves entities and schemas through mention detection and dense retrieval. It generates logical forms using LSTM, using retrieved items through knowledge-specific rules. SKP [110], DECAF [109], and KD-CoT [111] all retrieve triplets and conduct fusion-in-decoder [35] RAG. KD-CoT also follows a chain-of-thought paradigm, iteratively performing retrieval, generation, and verification.

4.3.2 Knowledge-augmented Open-domain Question Answering

Structured knowledge is often leveraged to augment ODQA (open-domain question answering).

Latent representation-based RAG, especially the fusion-in-decoder [35] technique, is prevalent for knowledge-augmented ODQA. UniK-QA [108], KG-FiD [277], Grape [278] all apply the fusion-in-decoder technique. They incorporate triplet-based

ARTICLE IN PRESS

documents, re-ranked documents through KG, and bipartite graph for pairs of question and passage, respectively.

OREOLM [279] empowers LLM with knowledge reasoning paths, integrating the entity value memory derived from contextualized random walk paths on KG into the hidden states of the LLM. SKURG [280] performs iterative retrieval and generation, using cross-attention to incorporate data sources into the input embedding. It uses a gate score to determine whether to re-start retrieval or to generate the real answer.

With the rapid development of LLMs, query-based RAG is emerging as a new standard. DIVKNOWQA [281] retrieves from multiple sources using different techniques. It iteratively retrieves and re-ranks the data before generating the final answer. KnowledGPT [282] uses generated code to retrieve from both public and personal knowledge bases. EFSUM [283] optimizes the evidence-focused summary after facts-augmented generation, so as to align the QA-specific preference for helpfulness and faithfulness. GenTKGQA [284] employs GNN (graph neural network) to integrate structural and temporal information from subgraph retrieval into virtual token representations. KnowledgeNavigator [285] performs retrieval on KG through iterative filtering of relations with respect to core entities, so as to obtain relevant triplets.

GNN-RAG [286] fuses LLMs' language understanding with GNN's reasoning prowess and employs a retrieval augmentation strategy to enhance KGQA performance.

4.3.3 Table for Question Answering

Tables, as another form of structured knowledge, also facilitate question answering.

Fusion-in-decoder [35] style RAG is often used for table QA. EfficientQA [287], a competition held in NeurIPS 2020, witnessed the proposal of numerous retrieval-reader systems that rely on textual and tabular data. Dual Reader-Parser [288] and CORE [289] both re-rank the retrieved textual and tabular data for generation. Convinse [290] retrieves information from knowledge bases, tables, and texts after question understanding. RINK [291] designs a set-level reader-inherited re-ranker to get the relevance score of table segments. TAG-QA [292] retrieves tables and texts through GNN (after table-to-graph conversion) and BM25, respectively.

Tables can be integrated into prompts for query-based RAG. Both T-RAG [293] and OmniTab [294] concatenates the retrieved tables with the query to generate the answer. CARP [295] extracts hybrid chain of retrieved tables and passages for prompt construction. StructGPT [269] retrieves from multiple sources including KGs, tables, and databases. cTBLS [296] forms prompts with ranked tables after retrieval. Min et al. [297] integrated tabular data through table-to-text techniques, then experiments on both finetuning and RAG. ERATTA [298] generates SQL code to extract table information, integrating it into the prompt to minimize model hallucination. TableRAG [299] extracts metadata from the prompt, retrieves more relevant table metadata, and feeds both into the generator for higher confidence results. THoRR [300] encodes the prompt and tables into vectors and selects the most relevant table using DPR, then employs a refinement encoder for granular data, feeding all into the generator for accurate results.

ARTICLE IN PRESS

4.3.4 Others

Prototype-KRG [301] integrates retrieved knowledge facts and dialogue prototypes into a GRU model through both hidden states and logits. SURGE [302] combines relevant subgraphs into the input for dialogue generation. RHO [303] fuses KG embedding of relevant entities and relations into textual embeddings during dialogue generation. K-LaMP [304] retrieves entities in history queries to construct prompt for query suggestion. ReSKGC [112] retrieves relevant triplets to complete triplet using Fid. G-Retriever [305] retrieves nodes and edges from textual graphs to construct subgraph and perform graph prompt tuning for QA. Hussien et al. [306] fuse the reasoning power of KG with the expressiveness of LLMs through RAG techniques. HippoRAG [307] excels in multi-hop question answering by emulating mammalian brain knowledge storage with KG triples and employing a personalized PageRank algorithm for retrieval. Going beyond traditional text-based knowledge graphs for enhanced model output, LAD-RAG [308] processes information-rich documents using a large vision model. It builds a knowledge graph from page elements where each node contains both semantic and categorical information, ultimately boosting the generator's performance.

4.4 RAG for Image

4.4.1 Image Generation

Image generation refers to the process of creating new images, typically using algorithms in the field of artificial intelligence and machine learning.

The retrieval process can not only help yield high-quality images even for rare or unseen subjects, but also reduces the parameter count and computational expense [45, 95, 103–107, 309]. For GAN-based model, RetrieveGAN [45] uses a differentiable retriever for image patch selection, facilitating end-to-end training. IC-GAN [95] models data as conditional distributions around each training instance, conditioning both the generator and discriminator on these instances. Recently, diffusion models beat GANs on image generation [310]. KNN-Diffusion [104] and RDM [105] train diffusion models conditioned on CLIP embeddings and image neighbors, enabling post-hoc conditioning on labels, prompts, and zero-shot stylization [106]. Beyond only images, Re-imagen [103] extends retrieval to image-text pairs for text-to-image generation, with interleaved guidance to balance the alignment between prompts and retrieval conditions. Retrieve&Fuse [309] prevents information loss of CLIP embeddings by concatenating retrieved and noised images before each U-Net attention block, allowing fully interaction via self-attention. RPG [311] retrieves representative images to construct in-context examples, and utilizes chain-of-thought reasoning [312] to plan out complementary subregions for compositional text-to-image diffusion.

4.4.2 Image Captioning

Image captioning is the process of generating a textual description of an image.

Retrieval-augmented image captioning typically synthesises description with a collection of retrieved captions. MA [121] augments via a memory bank, built with historical context and target word of image-text training set, and queried with inference

ARTICLE IN PRESS

context. In adversarial training, RAMP [313] takes retrieved captions as discriminator reference, and employs memory-augmented attention and copying mechanisms for better utilization of retrieved captions. The RA-Transformer [46] and EXTRA [314], both retrieval-augmented transformer-based captioning models, utilize cross-attention over encoded retrieved captions. Beyond caption retrieval, REVEAL [315] uniformly encodes and retrieves multi-modal world knowledge, integrated with retrieval score-aware attention. Directly, SMALLCAP [47] employs a CLIP vision encoder and a LLM decoder, with retrieved captions serving as input-specific in-context examples. For remote sensing images, CRSR [316] refines retrieved captions, filtering out misleading details and emphasizing visually salient content.

4.4.3 Others

There also exist many retrieval augmented works for other image-related tasks. For Visual Question Answering (VQA), PICa [317] converts images into textual descriptions, prompts GPT-3 and ensembles multi-query results. RA-VQA [318] enables an end-to-end training with differentiable retrieval for answer generation. For visually grounded dialogue, KIF [319] and Maria [320] enhances dialog generation with external knowledge like visual experiences. In multi-modal machine translation, [321] incorporates visual information at the phrase level to improve NMT with multi-modal information.

4.5 RAG for Video

4.5.1 Video Captioning

Video captioning translates the visual content into descriptive utterances. KaVD [322] generates news video caption with background knowledge in related documents like named entities and events. R-ConvED [48] retrieves relevant sentences and videos via Dual Encoding [70], and predicts the target word with a convolutional encoder-decoder network. CARE [117] combines three modalities data, i.e. frame, audio, and retrieved texts, to provide both global and local semantic guidance as augmentation. EgoInstructor [49] focuses on first-person videos, retrieves relevant exocentric videos and texts, and generates captions through LLM via cross-attention with encoded videos

4.5.2 Video QA&Dialogue

Video QA&Dialogue generates single or multiple-round responses in alignment with video content. For VideoQA, MA-DRNN [323] stores and retrieves useful information in queries and videos with external memory, therefore models the long-term visual-textual dependence. R2A [324] retrieves semantically similar texts by CLIP, and prompts LLM with both the query and the retrieved texts. For video dialogue, [325] proposes TVQA+ dataset to enable relevant moments and visual concepts retrieval, and designs corresponding spatio-temporal-aware generator. VGNMN [326] extracts visual cues from videos, while the retrieval process is parameterized by entities and actions in previous dialogues.

ARTICLE IN PRESS

4.5.3 Others

RAG also works for other video-related tasks. VidIL [327] converts video content into temporal-aware LLM prompts for tasks like video captioning, question answering, and future event prediction. For trustworthy autonomous driving, RAG-Driver [328] grounds the MLLM in retrieved expert demonstrations, to produce driving action explanations. Animate-A-Story [177] simplifies text-to-video generation by dividing it into plot-based video augmentation and video-diffusion generation conditioned on text and video inputs.

4.6 RAG for Audio

4.6.1 Audio Generation

Audio generation usually synthesises audio with natural language prompt. Given input prompt, Re-AudioLDM [116] retrieves relevant caption-audio pairs with dense retriever CLAP [26] for generation. Make-An-Audio [44] retrieves audios given text prompt, then constructs pseudo prompts for text-to-audio diffusion model training.

4.6.2 Audio Captioning

Audio captioning, basically a sequence-to-sequence task, generates natural language data for audio data. RECAP [329] and [43] leverages dense retrievers, CLAP [26] and VGGish [69] respectively, to retrieve related captions given audio data. For RECAP, captions are included into LLM prompts, while [43] uses both audio and retrieved captions in attention module. Other research studies align audio modality with text to leverage advancements in LLMs [330–332] for various downstream text generation.

4.7 RAG for 3D

4.7.1 Text-to-3D

Retrieval can be applied to augment 3D asset generation. ReMoDiffuse [51] retrieves relevant motion entities and generates motions using diffusion models, with the semantic-modulated attention and condition mixture guidance. AMD [115] designs and fuses two motion diffusion models. One branch conditions on the original prompt, while the other decomposes the prompt into anatomical scripts and retrieves similar motions. RetDream [50] retrieves 3D assets to augment the variational score distillation [333] of 2D diffusion models. These assets offer geometric and adapted 2D priors, which not only impose additional velocity on particles for initialization but also help optimize 2D diffusion models by LoRA.

4.8 RAG for Science

RAG has also emerged as a promising research direction for many interdisciplinary applications, such as spatiotemporal prediction [334–336], molecular generation, medical tasks and computational research.

ARTICLE IN PRESS

4.8.1 Drug Discovery

The goal of drug discovery is to generate molecules that concurrently fulfill diverse properties.

RetMol [55] integrates a lightweight retrieval mechanism and molecular strings into a pre-trained encoder-decoder generative model to retrieve and fuse exemplar molecules with the input. PromptDiff [337] introduces an interaction-based, retrieval-augmented 3D molecular diffusion model that retrieves a curated set of ligand references to guide the synthesis of ligands meeting specific design criteria.

4.8.2 Biomedical Informatics Enhancement

Several recent studies have improved the expressiveness of LLM by retrieving information from biomedical domain-specific databases, thereby augmenting the model's capabilities to provide valuable guidance for tasks in the medical field.

PoET [338] is an autoregressive model using a transformer variant with a retrieval mechanism for prompt augmentation, speeding up the prediction of protein variant fitness properties. Chat-Orthopedist [94] enhances ChatGPT with a retrieval-augmented mechanism focused on adolescent idiopathic scoliosis (AIS), utilizing an external knowledge base for precise responses. BIOREADER [339] is the first retrieval-enhanced text-to-text transformer-based model for biomedical natural language processing, incorporating the retrieved literature evidence into the model using a chunked-cross attention mechanism. MedWriter [340] employs a hierarchical retrieval-augmented generation method that combines report-level and sentence-level templates to produce coherent and clinically accurate medical reports from images. QA-RAG [341] employs a dual-track RAG strategy to enhance pharmaceutical compliance by effectively retrieving and integrating regulatory guidelines based on language model responses and user queries. RAG-RLRC-LaySum [342] leverages biomedical text knowledge for llms, employing reinforcement learning and re-ranking techniques to enhance content relevance and readability of the output.

4.8.3 Math Applications

Retrieval-augmented generation technology in mathematics streamlines problem-solving, boosts research innovation, and refines educational strategies.

LeanDojo [343] boosts theorem proving by using retrieval-augmented methods to choose relevant premises from extensive mathematical libraries, improving automation and theorem generalization. RAG-for-math-QA [344] improves math question-answering by integrating a high-quality math textbook with RAG, enhancing LLM-generated responses for middle-school algebra and geometry.

5 Benchmark

Given the increasing research interests and applications of RAG, there have also been several benchmarks assessing RAG from certain aspects.

Chen et al. [345] proposed an RAG benchmark that evaluates across four dimensions: (1) Noise Robustness, testing if LLMs can extract necessary information from

ARTICLE IN PRESS

noisy documents; (2) Negative Rejection, assessing if LLMs can reject to respond when retrieved content is insufficient; (3) Information Integration,checking if LLMs can acquire knowledge and respond by integrating multiple retrieved contents; (4) Counterfactual Robustness, determining if LLMs can identify counterfactual errors in retrieved content.

Three other benchmarks, RAGAS [346], ARES [347], and TruLens [348], evaluate three aspects using a separate evaluator LLM: (1) Faithfulness, assessing factual accuracy based on retrieved content; (2) Answer Relevance, determining if results address the queries; (3) Context Relevance, evaluating the relevance of retrieved content and its conciseness. CRUD-RAG [349] divides RAG tasks into four types: Create, Read, Update, and Delete, assessing them through text continuation, question answering, hallucination correction, and open-domain multi-document summary. MIRAGE [350] assesses RAG in the medical domain, focusing on the performance of medical question-answering systems. KILT [351] aligns Wikipedia snapshots to verify information accuracy, using BLEU scores to pinpoint relevant texts and filtering to uphold quality, thus providing diverse retrieval systems for evidence-backed predictions or citations.

Furthermore, a host of benchmarks for specific domains and tasks are also receiving a lot of attention. CRAG [352] incorporates a diverse set of questions across five domains and eight categories, covering the full spectrum of entity popularity (from popular to long-tail) with temporal dynamisms that range from years down to seconds. Multihop-rag [200] primarily benchmarks the accuracy of various RAG methodologies in multi-hop question answering. Legalbench-rag [353] is a benchmark specifically designed to evaluate the retrieval step in legal-domain RAG pipelines. It emphasizes precise retrieval, with a focus on extracting minimal, highly relevant text segments from legal documents. Omnieval [354] is an all-encompassing, automated RAG benchmark designed for the financial sector, covering five task categories and spanning sixteen financial topics.

6 Discussion

6.1 Limitations

Despite the widespread adoption of RAG, it suffers from several limitations by nature.

6.1.1 Noises in Retrieval Results

Information retrieval is inherently flawed due to information loss in item representations and ANN search. The inevitable noise, manifesting as irrelevant content or misleading information, can create failure points in RAG systems [355]. However, although improving retrieval accuracy seems intuitive for RAG effectiveness, recent research surprisingly finds that noisy retrieval results might enhance generation quality [356]. A possible explanation is that diverse retrieval outcomes could contribute to prompt construction [357]. Thus, the impact of retrieval noise remains unclear, leading to confusion about metric selection and retriever-generator interaction in practical uses.

ARTICLE IN PRESS

6.1.2 Extra Overhead

While retrieval can reduce generation costs in certain cases [30–32], it incurs non-negligible overhead in most cases. In other words, the retrieval and interaction processes increase latency inevitably. This is amplified when RAG is combined with complex enhancement methods, such as recursive retrieval [358] and iterative RAG [189]. Furthermore, as the scale of retrieval sources expands, the storage and access complexity will also increase [359]. Such overhead hampers the practicality of RAG in real-time services that are sensitive to latency.

6.1.3 The Gap between Retrievers and Generators

Since the objectives of retrievers and generators may not align, and their latent spaces might differ, designing their interaction requires meticulous design and optimization. Current approaches either disentangle retrieval and generation or integrate them at an intermediate stage. While the former is more modular, the latter could benefit from joint training but hamper generality. Selecting a cost-effective interaction method to bridge the gap poses a challenge and necessities deliberation in practice.

6.1.4 Increased System Complexity

The introduction of retrieval unavoidably increases the system complexity and the number of hyper-parameters to tune. For instance, a recent study found that using top-k rather than a single retrieval improves attribution but harms fluency in query-based RAG [360], while other aspects such as metric selection are still under explored. Thus, it requires more expertise to tune the generation service when RAG is involved.

6.1.5 Lengthy Context

One of the primary shortcomings of RAG, in particular the query-based RAG, is that it lengthens the context tremendously, making it infeasible for generators with limited context length. In addition, the lengthened context also slows down the generation process generally. The research advancements in prompt compression [172, 361] and long-context support [362] have partially mitigated these challenges, albeit with a slight trade-off in accuracy or costs.

6.2 Potential Future Directions

Lastly, we wish to outline several potential directions for future RAG research and applications.

6.2.1 Novel Design of Augmentation Methodologies

Existing research has explored various interaction patterns between retrievers and generators. However, due to distinct objectives in these two components, the practical augmentation process has a significant impact on the final generation results. Investigation of more advanced foundations for augmentation holds promise for fully unleashing the potential of RAG.

ARTICLE IN PRESS

6.2.2 Flexible RAG Pipelines

RAG systems are progressively embracing flexible pipelines, such as recursive, adaptive, and iterative RAG. With precise tuning and meticulous engineering, the unique blend of retrieval sources, retrievers, generators, and RAG subsystems promises to tackle complex tasks and boost overall performance. We eagerly anticipate pioneering exploration that will drive the evolution of even more innovative RAG systems.

6.2.3 Broader Applications

RAG is a general technique applied in various applications. However, some generative tasks have not yet explored RAG, and in many domains, RAG is applied naively without considering the domain's unique characteristics. We believe designing domain-specific RAG techniques will significantly benefit broader applications.

6.2.4 Efficient Deployment and Processing

There exist several deployment solutions for query-based RAG with LLMs, such as LangChain [363], LLAMA-Index [136], and PipeRAG [364]. However, for other RAG foundations and/or generation tasks, there lacks a plug-and-play solution. Besides, due to retrieval overhead and increasing complexities in retrievers and generators, achieving efficient RAG is still challenging and necessitates further system-level [365–368] optimizations.

6.2.5 Incorporating Long-tail and Real-time Knowledge

While a key motivation of RAG is to harness real-time and long-tail knowledge, few studies have explored the pipeline for knowledge updating and expansion. Many existing works use merely the generators' training data as retrieval sources, neglecting the dynamic and flexible information that retrieval could offer. As a consequence, there is a growing research on designing RAG systems with continuously updated knowledge and flexible sources. We also expect RAG to step further, adapting to personalized information in today's web service.

6.2.6 Combined with Other Techniques

RAG is orthogonal to other techniques that also aim to improve AIGC effectiveness, such as fine-tuning, reinforcement learning, chain-of-thought, and agent-based generation. The combining of these methods [369] is still in its early stages, calling for further research to fully exploit their potential through novel algorithm designs. It is worthy to note that a recent notion appears "long-context models like Gemini 1.5 will replace RAG". Nevertheless, this assertion overlooks RAG's flexibility in managing dynamic information, encompassing both up-to-date and long-tail knowledge [370]. We expect RAG to benefit from long context generation, rather than being replaced by it.

6.2.7 Agentic RAG

With the increasing integration of agents [371] into people's daily lives, a novel RAG pipeline known as Agentic RAG [372, 373] has emerged. It represents an advanced

ARTICLE IN PRESS

evolution of the traditional RAG paradigm, with its core innovation lying in the integration of AI agents endowed with autonomous decision-making and planning capabilities to overcome the limitations of conventional approaches. Traditional RAG, characterized by its linear and static workflow, struggles with complex tasks that require multi-step reasoning and cross-source information integration. It also lacks the ability to refine contextual noise and dynamically invoke external tools.

In Agentic RAG, the agent functions as an intelligent coordinator, transforming the process into a dynamic "reasoning–planning–acting–iterating" loop [374–376]. It is responsible for task decomposition, dynamic tool selection, and iterative contextual refinement. The advantages of this architecture stem from the autonomy, dynamism, modularity, and collaborative capacity provided by the agent. These attributes enable significant improvements in the accuracy and reliability of outputs through multi-step reasoning [377] and quality control, enhance the ability to handle complex tasks via dynamic planning and tool invocation, and ultimately facilitate a transition from passive response to proactive problem-solving.

7 Conclusion

In this paper, we conducted a thorough and comprehensive survey on RAG within the context of AIGC, with a particular focus on augmentation foundations, enhancements, and applications. We first systematically organized and summarized the foundation paradigms in RAG, providing insights into the interaction between retrievers and generators. Then, we reviewed the enhancements that further improve the effectiveness of RAG, including the enhancements on each component or the entire pipeline. To facilitate researchers across diverse domains, we showcased practical applications of RAG in a range of modalities and tasks. Finally, we also presented existing benchmarks for RAG, discussed current limitations of RAG, and shed light on promising future directions.

ARTICLE IN PRESS

References

[1] Brown, T.B., Mann, B., *et al.*: Language models are few-shot learners. In: NeurIPS (2020)

[2] Chen, M., Tworek, J., et al.: Evaluating large language models trained on code. arXiv:2107.03374 (2021)

[3] OpenAI: GPT-4 technical report. arXiv:2303.08774 (2023)

[4] Touvron, H., et al.: Llama: Open and efficient foundation language models. arXiv:2302.13971 (2023)

[5] Touvron, H., et al.: Llama 2: Open foundation and fine-tuned chat models. arXiv:2307.09288 (2023)

ARTICLE IN PRESS

[6] Rozière, B., Gehring, J., et al.: Code llama: Open foundation models for code. arXiv:2308.12950 (2023)

[7] Ramesh, A., Pavlov, M., Goh, G., *et al.*: Zero-shot text-to-image generation. In: ICML (2021)

[8] Ramesh, A., Dhariwal, P., Nichol, A., et al.: Hierarchical text-conditional image generation with CLIP latents. arXiv:2204.06125 (2022)

[9] Betker, J., Goh, G., Jing, L., *et al.*: Improving image generation with better captions. Computer Science **2**(3), 8 (2023)

[10] Rombach, R., Blattmann, A., Lorenz, D., *et al.*: High-resolution image synthesis with latent diffusion models. In: IEEE/CVF (2022)

[11] OpenAI: Video generation models as world simulators. https://openai.com/research/video-generation-models-as-world-simulators (2024)

[12] Hochreiter, S., Schmidhuber, J.: Long short-term memory. Neural Comput. **9**(8), 1735–1780 (1997)

[13] Vaswani, A., Shazeer, N., Parmar, N., *et al.*: Attention is all you need. In: NeurIPS (2017)

[14] Goodfellow, I., Pouget-Abadie, J., *et al.*: Generative adversarial networks. CACM **63**(11), 139–144 (2020)

[15] Devlin, J., Chang, M., *et al.*: BERT: pre-training of deep bidirectional transformers for language understanding. In: NAACL-HLT (2019)

[16] Raffel, C., Shazeer, N., Roberts, A., *et al.*: Exploring the limits of transfer learning with a unified text-to-text transformer. JMLR **21**, 140–114067 (2020)

[17] Fedus, W., Zoph, B., Shazeer, N.: Switch transformers: Scaling to trillion parameter models with simple and efficient sparsity. JMLR **23**(120), 1–39 (2022)

[18] Kaplan, J., McCandlish, S., et al.: Scaling laws for neural language models (2020)

[19] Robertson, S.E., Zaragoza, H.: The probabilistic relevance framework: BM25 and beyond. FTIR **3**(4), 333–389 (2009)

[20] Karpukhin, V., Oguz, B., Min, S., *et al.*: Dense passage retrieval for open-domain question answering. In: EMNLP (2020)

[21] Johnson, J., Douze, M., Jégou, H.: Billion-scale similarity search with gpus. IEEE Trans. Big Data **7**(3), 535–547 (2021)

ARTICLE IN PRESS

ARTICLE IN PRESS

[22] Chen, Q., Zhao, B., Wang, H., *et al.*: SPANN: highly-efficient billion-scale approximate nearest neighborhood search. In: NeurIPS (2021)

[23] Datta, R., Joshi, D., Li, J., *et al.*: Image retrieval: Ideas, influences, and trends of the new age. CSUR **40**(2), 5–1560 (2008)

[24] Radford, A., Kim, J.W., Hallacy, C., *et al.*: Learning transferable visual models from natural language supervision. In: ICML (2021)

[25] Feng, Z., Guo, D., *et al.*: Codebert: A pre-trained model for programming and natural languages. In: EMNLP Findings (2020)

[26] Wu, Y., Chen, K., Zhang, T., *et al.*: Large-scale contrastive language-audio pre-training with feature fusion and keyword-to-caption augmentation. In: ICASSP (2023)

[27] Mallen, A., Asai, A., Zhong, V., *et al.*: When not to trust language models: Investigating effectiveness of parametric and non-parametric memories. In: ACL (2023)

[28] Carlini, N., Tramèr, F., *et al.*: Extracting training data from large language models. In: USENIX (2021)

[29] Kang, M., Gürel, N.M., et al.: C-RAG: certified generation risks for retrieval-augmented language models. arXiv:2402.03181 (2024)

[30] Izacard, G., Lewis, P., et al.: Atlas: Few-shot learning with retrieval augmented language models. arXiv:2208.03299 (2022)

[31] Wu, Y., Rabe, M.N., *et al.*: Memorizing transformers. In: ICLR (2022)

[32] He, Z., Zhong, Z., et al.: REST: retrieval-based speculative decoding. arxiv:2311.08252 (2023)

[33] Guu, K., Lee, K., et al.: REALM: retrieval-augmented language model pre-training. ICML (2020)

[34] Lewis, P.S.H., Perez, E., *et al.*: Retrieval-augmented generation for knowledge-intensive NLP tasks. In: NeurIPS (2020)

[35] Izacard, G., Grave, E.: Leveraging passage retrieval with generative models for open domain question answering. In: EACL (2021)

[36] Borgeaud, S., Mensch, A., *et al.*: Improving language models by retrieving from trillions of tokens. In: ICML (2022)

[37] Khandelwal, U., Levy, O., Jurafsky, D., *et al.*: Generalization through memorization: Nearest neighbor language models. In: ICLR (2020)

ARTICLE IN PRESS

ARTICLE IN PRESS

[38] He, J., Neubig, G., Berg-Kirkpatrick, T.: Efficient nearest neighbor language models. In: EMNLP (2021)

[39] zilliztech: GPTCache. https://github.com/zilliztech/GPTCache

[40] Parvez, M.R., Ahmad, W.U., *et al.*: Retrieval augmented code generation and summarization. In: EMNLP Findings (2021)

[41] Ahmad, W.U., Chakraborty, S., Ray, B., *et al.*: Unified pre-training for program understanding and generation. In: NAACL-HLT (2021)

[42] Zhou, S., Alon, U., Xu, F.F., *et al.*: Docprompting: Generating code by retrieving the docs. In: ICLR (2023)

[43] Koizumi, Y., Ohishi, Y., et al.: Audio captioning using pre-trained large-scale language model guided by audio-based similar caption retrieval. arXiv:2012.07331 (2020)

[44] Huang, R., Huang, J., Yang, D., *et al.*: Make-an-audio: Text-to-audio generation with prompt-enhanced diffusion models. In: ICML (2023)

[45] Tseng, H.-Y., Lee, H.-Y., *et al.*: Retrievegan: Image synthesis via differentiable patch retrieval. In: ECCV (2020)

[46] Sarto, S., Cornia, M., Baraldi, L., Cucchiara, R.: Retrieval-augmented transformer for image captioning. In: CBMI (2022)

[47] Ramos, R., *et al.*: Smallcap: lightweight image captioning prompted with retrieval augmentation. In: CVPR (2023)

[48] Chen, J., Pan, Y., Li, Y., *et al.*: Retrieval augmented convolutional encoder-decoder networks for video captioning. TOMCCAP **19**(1s), 48–14824 (2023)

[49] Xu, J., Huang, Y., et al.: Retrieval-augmented egocentric video captioning. arXiv:2401.00789 (2024)

[50] Seo, J., Hong, S., et al.: Retrieval-augmented score distillation for text-to-3d generation. arXiv:2402.02972 (2024)

[51] Zhang, M., Guo, X., *et al.*: Remodiffuse: Retrieval-augmented motion diffusion model. In: ICCV (2023)

[52] Hu, X., Wu, X., Shu, Y., Qu, Y.: Logical form generation via multi-task learning for complex question answering over knowledge bases. In: COLING (2022)

[53] Huang, X., Kim, J., Zou, B.: Unseen entity handling in complex question answering over knowledge base via language generation. In: EMNLP Findings (2021)

ARTICLE IN PRESS

ARTICLE IN PRESS

[54] Das, R., Zaheer, M., Thai, D., *et al.*: Case-based reasoning for natural language queries over knowledge bases. In: EMNLP (2021)

[55] Wang, Z., Nie, W., Qiao, Z., *et al.*: Retrieval-based controllable molecule generation. In: ICLR (2022)

[56] Jin, Q., Yang, Y., Chen, Q., Lu, Z.: Genegpt: Augmenting large language models with domain tools for improved access to biomedical information. Bioinformatics **40**(2), 075 (2024)

[57] Li, H., Su, Y., et al.: A survey on retrieval-augmented text generation. arxiv:2202.01110 (2022)

[58] Asai, A., Min, S., Zhong, Z., Chen, D.: Acl 2023 tutorial: Retrieval-based language models and applications. ACL 2023 (2023)

[59] Gao, Y., Xiong, Y., et al.: Retrieval-augmented generation for large language models: A survey. arxiv:2312.10997 (2023)

[60] Zhao, R., *et al.*: Retrieving multimodal information for augmented generation: A survey. In: EMNLP (2023)

[61] Fan, W., Ding, Y., Ning, L., Wang, S., Li, H., Yin, D., Chua, T.-S., Li, Q.: A survey on rag meeting llms: Towards retrieval-augmented large language models. In: Proceedings of the 30th ACM SIGKDD Conference on Knowledge Discovery and Data Mining, pp. 6491–6501 (2024)

[62] Chen, J., Guo, H., Yi, K., *et al.*: Visualgpt: Data-efficient adaptation of pretrained language models for image captioning. In: CVPR (2022)

[63] Tay, Y., Dehghani, M., *et al.*: Efficient transformers: A survey. CSUR **55**(6), 109–110928 (2023)

[64] Houdt, G.V., *et al.*: A review on the long short-term memory model. Artif. Intell. Rev. **53**(8), 5929–5955 (2020)

[65] Yang, L., Zhang, Z., *et al.*: Diffusion models: A comprehensive survey of methods and applications. CSUR **56**(4), 1–39 (2023)

[66] Gui, J., Sun, Z., Wen, Y., *et al.*: A review on generative adversarial networks: Algorithms, theory, and applications. TKDE **35**(4), 3313–3332 (2023)

[67] Robertson, S.E., Walker, S.: On relevance weights with little relevance information. In: SIGIR (1997)

[68] Lafferty, J.D., Zhai, C.: Document language models, query models, and risk minimization for information retrieval. In: SIGIR (2001)

ARTICLE IN PRESS

ARTICLE IN PRESS

[69] Hershey, S., Chaudhuri, S., *et al.*: CNN architectures for large-scale audio classification. In: ICASSP (2017)

[70] Dong, J., Li, X., Xu, C., *et al.*: Dual encoding for zero-example video retrieval. In: CVPR (2019)

[71] Xiong, L., Xiong, C., Li, Y., *et al.*: Approximate nearest neighbor negative contrastive learning for dense text retrieval. In: ICLR (2021)

[72] Bentley, J.L.: Multidimensional binary search trees used for associative searching. CACM **18**(9), 509–517 (1975)

[73] Li, W., Feng, C., *et al.*: Learning balanced tree indexes for large-scale vector retrieval. In: SIGKDDg (2023)

[74] Datar, M., Immorlica, N., Indyk, P., *et al.*: Locality-sensitive hashing scheme based on p-stable distributions. In: SCG (2004)

[75] Malkov, Y.A., Yashunin, D.A.: Efficient and robust approximate nearest neighbor search using hierarchical navigable small world graphs. TPAMI **42**(4), 824–836 (2018)

[76] Jayaram Subramanya, S., Devvrit, F., et al.: Diskann: Fast accurate billion-point nearest neighbor search on a single node. NeurIPS (2019)

[77] Wang, Y., Hou, Y., *et al.*: A neural corpus indexer for document retrieval. In: NeurIPS (2022)

[78] Zhang, H., Wang, Y., *et al.*: Model-enhanced vector index. In: NeurIPS (2023)

[79] Hayati, S.A., Olivier, R., *et al.*: Retrieval-based neural code generation. In: EMNLP (2018)

[80] Zhang, J., Wang, X., Zhang, H., *et al.*: Retrieval-based neural source code summarization. In: ICSE (2020)

[81] Poesia, G., Polozov, A., Le, V., *et al.*: Synchromesh: Reliable code generation from pre-trained language models. In: ICLR (2022)

[82] Ye, X., Yavuz, S., *et al.*: RNG-KBQA: generation augmented iterative ranking for knowledge base question answering. In: ACL (2022)

[83] Shu, Y., et al.: TIARA: multi-grained retrieval for robust question answering over large knowledge bases. arXiv:2210.12925 (2022)

[84] Lin, X.V., Socher, R., et al.: Bridging textual and tabular data for cross-domain text-to-sql semantic parsing. arXiv:2012.12627 (2020)

ARTICLE IN PRESS

ARTICLE IN PRESS

[85] Asai, A., Wu, Z., et al.: Self-rag: Learning to retrieve, generate, and critique through self-reflection. arxiv:2310.11511 (2023)

[86] Shi, W., Min, S., et al.: Replug: Retrieval-augmented black-box language models. arXiv:2301.12652 (2023)

[87] Ram, O., Levine, Y., et al.: In-context retrieval-augmented language models. arXiv:2302.00083 (2023)

[88] Zan, D., Chen, B., Lin, Z., *et al.*: When language model meets private library. In: EMNLP Findings (2022)

[89] Nashid, N., Sintaha, M., Mesbah, A.: Retrieval-based prompt selection for code-related few-shot learning. In: ICSE (2023)

[90] Jin, M., Shahriar, S., Tufano, M., *et al.*: Inferfix: End-to-end program repair with llms. In: ESEC/FSE (2023)

[91] Lu, S., Duan, N., Han, H., *et al.*: Reacc: A retrieval-augmented code completion framework. In: ACL (2022)

[92] Liu, Y., *et al.*: Uni-parser: Unified semantic parser for question answering on knowledge base and database. In: EMNLP (2022)

[93] Yang, Z., Du, X., Cambria, E., *et al.*: End-to-end case-based reasoning for commonsense knowledge base completion. In: EACL (2023)

[94] Shi, W., Zhuang, Y., Zhu, Y., *et al.*: Retrieval-augmented large language models for adolescent idiopathic scoliosis patients in shared decision-making. In: ACM-BCB (2023)

[95] Casanova, A., Careil, M., Verbeek, J., *et al.*: Instance-conditioned gan. In: NeurIPS (2021)

[96] Bertsch, A., Alon, U., et al.: Unlimiformer: Long-range transformers with unlimited length input (2023)

[97] Kuratov, Y., Bulatov, A., et al.: In search of needles in a 10m haystack: Recurrent memory finds what llms miss. arXiv:2402.10790 (2024)

[98] Li, J., Li, Y., *et al.*: Editsum: A retrieve-and-edit framework for source code summarization. In: ASE (2021)

[99] Yu, C., Yang, G., Chen, X., *et al.*: Bashexplainer: Retrieval-augmented bash code comment generation based on fine-tuned codebert. In: ICSME (2022)

[100] Hashimoto, T.B., Guu, K., *et al.*: A retrieve-and-edit framework for predicting structured outputs. In: NeurIPS (2018)

ARTICLE IN PRESS

ARTICLE IN PRESS

[101] Wei, B., Li, Y., Li, G., *et al.*: Retrieve and refine: Exemplar-based neural comment generation. In: ASE (2020)

[102] Shi, E., Wang, Y., Tao, W., *et al.*: RACE: retrieval-augmented commit message generation. In: EMNLP (2022)

[103] Chen, W., Hu, H., Saharia, C., Cohen, W.W.: Re-imagen: Retrieval-augmented text-to-image generator. In: ICLR (2023)

[104] Sheynin, S., Ashual, O., *et al.*: Knn-diffusion: Image generation via large-scale retrieval. In: ICLR (2023)

[105] Blattmann, A., Rombach, R., Oktay, K., *et al.*: Retrieval-augmented diffusion models. In: NeurIPS (2022)

[106] Rombach, R., Blattmann, A., Ommer, B.: Text-guided synthesis of artistic images with retrieval-augmented diffusion models. arXiv:2207.13038 (2022)

[107] Li, B., Torr, P.H., et al.: Memory-driven text-to-image generation. arXiv:2208.07022 (2022)

[108] Oguz, B., Chen, X., Karpukhin, V., *et al.*: Unik-qa: Unified representations of structured and unstructured knowledge for open-domain question answering. In: NAACL Findings (2022)

[109] Yu, D., Zhang, S., *et al.*: Decaf: Joint decoding of answers and logical forms for question answering over knowledge bases. In: ICLR (2023)

[110] Dong, G., Li, R., Wang, S., *et al.*: Bridging the kb-text gap: Leveraging structured knowledge-aware pre-training for KBQA. In: CIKM (2023)

[111] Wang, K., Duan, F., Wang, S., et al.: Knowledge-driven cot: Exploring faithful reasoning in llms for knowledge-intensive question answering. arXiv:2308.13259 (2023)

[112] Yu, D., Yang, Y.: Retrieval-enhanced generative model for large-scale knowledge graph completion. In: SIGIR (2023)

[113] Févry, T., Soares, L.B., *et al.*: Entities as experts: Sparse memory access with entity supervision. In: EMNLP (2020)

[114] Jong, M., Zemlyanskiy, Y., *et al.*: Mention memory: incorporating textual knowledge into transformers through entity mention attention. In: ICLR (2021)

[115] Jing, B., Zhang, Y., Song, Z., *et al.*: Amd: Anatomical motion diffusion with interpretable motion decomposition and fusion. In: AAAI (2024)

[116] Yuan, Y., Liu, H., Liu, X., *et al.*: Retrieval-augmented text-to-audio generation.

ARTICLE IN PRESS

ARTICLE IN PRESS

In: ICASSP (2024)

[117] Yang, B., Cao, M., Zou, Y.: Concept-aware video captioning: Describing videos with effective prior information. TIP **32**, 5366–5378 (2023)

[118] Zhong, Z., Lei, T., Chen, D.: Training language models with memory augmentation. In: EMNLP (2022)

[119] Min, S., Shi, W., *et al.*: Nonparametric masked language modeling. In: ACL Findings (2023)

[120] Zhang, X., Zhou, Y., Yang, G., Chen, T.: Syntax-aware retrieval augmented code generation. In: EMNLP Findings (2023)

[121] Fei, Z.: Memory-augmented image captioning. In: AAAI (2021)

[122] Leviathan, Y., Kalman, M., Matias, Y.: Fast inference from transformers via speculative decoding. In: ICML (2023)

[123] Lan, T., Cai, D., Wang, Y., *et al.*: Copy is all you need. In: ICLR (2023)

[124] Cao, B., Cai, D., Cui, L., et al.: Retrieval is accurate generation. arXiv:2402.17532 (2024)

[125] Wang, L., Yang, N., Wei, F.: Query2doc: Query expansion with large language models. In: EMNLP (2023)

[126] Gao, L., Ma, X., Lin, J., Callan, J.: Precise zero-shot dense retrieval without relevance labels. In: ACL (2023)

[127] Kim, G., Kim, S., Jeon, B., *et al.*: Tree of clarifications: Answering ambiguous questions with retrieval-augmented large language models. In: EMNLP (2023)

[128] Chan, C.-M., Xu, C., et al.: Rq-rag: Learning to refine queries for retrieval augmented generation. arXiv:2404.00610 (2024)

[129] Tayal, A., Tyagi, A.: Dynamic contexts for generating suggestion questions in rag based conversational systems. In: WWW'24 Companion (2024)

[130] Xia, M., Malladi, S., Gururangan, S., et al.: LESS: selecting influential data for targeted instruction tuning. arXiv:2402.04333 (2024)

[131] Bornea, A.-L., Ayed, F., et al.: Telco-rag: Navigating the challenges of retrieval-augmented language models for telecommunications. arXiv:2404.15939 (2024)

[132] Tan, L., Huang, K.-W., Shi, J., Wu, K.: Interpdetect: Interpretable signals for detecting hallucinations in retrieval-augmented generation. arXiv preprint arXiv:2510.21538 (2025)

ARTICLE IN PRESS

ARTICLE IN PRESS

[133] Yao, S., Zhao, J., *et al.*: React: Synergizing reasoning and acting in language models. In: ICLR (2023)

[134] Wei, J., Wang, X., Schuurmans, D., *et al.*: Chain-of-thought prompting elicits reasoning in large language models. In: NeurIPS (2022)

[135] Pouplin, T., Sun, H., Holt, S., Schaar, M.: Retrieval-augmented thought process as sequential decision making. arXiv:2402.07812 (2024)

[136] Liu, J.: LlamaIndex. https://github.com/jerryjliu/llama_index

[137] Sarthi, P., Abdullah, S., Tuli, A., *et al.*: Raptor: Recursive abstractive processing for tree-organized retrieval. In: ICLR (2023)

[138] Kang, B., Kim, J., et al.: Prompt-rag: Pioneering vector embedding-free retrieval-augmented generation in niche domains, exemplified by korean medicine. arXiv:2401.11246 (2024)

[139] Raina, V., et al.: Question-based retrieval using atomic units for enterprise rag. arXiv:2405.12363 (2024)

[140] Zhao, J., Ji, Z., Niu, S., Wang, H., Xiong, F., Li, Z.: Mom: Mixtures of scenario-aware document memories for retrieval-augmented generation systems. arXiv preprint arXiv:2510.14252 (2025)

[141] Xiao, S., Liu, Z., Zhang, P., et al.: C-pack: Packaged resources to advance general chinese embedding. arxiv:2309.07597 (2023)

[142] Chen, J., Xiao, S., Zhang, P., et al.: Bge m3-embedding: Multi-lingual, multi-functionality, multi-granularity text embeddings through self-knowledge distillation. arxiv:2309.07597 (2023)

[143] Xiao, S., Liu, Z., Zhang, P., Xing, X.: Lm-cocktail: Resilient tuning of language models via model merging. arxiv:2311.13534 (2023)

[144] Zhang, P., Xiao, S., Liu, Z., Dou, Z., Nie, J.-Y.: Retrieve anything to augment large language models. arxiv:2310.07554 (2023)

[145] Kulkarni, M., Tangarajan, P., Kim, K., et al.: Reinforcement learning for optimizing RAG for domain chatbots. arXiv:2401.06800 (2024)

[146] Wang, W., Wang, Y., *et al.*: Rap-gen: Retrieval-augmented patch generation with codet5 for automatic program repair. In: ESEC/FSE (2023)

[147] Sawarkar, K., Mangal, A., et al.: Blended rag: Improving rag (retriever-augmented generation) accuracy with semantic search and hybrid query-based retrievers. arXiv:2404.07220 (2024)

ARTICLE IN PRESS

ARTICLE IN PRESS

[148] Yan, S.-Q., Gu, J.-C., Zhu, Y., Ling, Z.-H.: Corrective retrieval augmented generation. arXiv:2401.15884 (2024)

[149] Huang, W., Lapata, M., Vougiouklis, P., *et al.*: Retrieval augmented generation with rich answer encoding. In: IJCNLP-AACL (2023)

[150] Wang, H., Huang, W., Deng, Y., et al.: Unims-rag: A unified multi-source retrieval-augmented generation for personalized dialogue systems. arXiv:2401.13256 (2024)

[151] Koley, S., Bhunia, A.K., *et al.*: You'll never walk alone: A sketch and text duet for fine-grained image retrieval. In: CVPR (2024)

[152] Glass, M.R., Rossiello, G., Chowdhury, M.F.M., *et al.*: Re2g: Retrieve, rerank, generate. In: NAACL (2022)

[153] Nogueira, R.F., Cho, K.: Passage re-ranking with BERT. arxiv:1901.04085 (2019)

[154] Li, J., Zhao, Y., Li, Y., et al.: Acecoder: Utilizing existing code to enhance code generation. arXiv:2303.17780 (2023)

[155] Shi, P., Zhang, R., Bai, H., Lin, J.: XRICL: cross-lingual retrieval-augmented in-context learning for cross-lingual text-to-sql semantic parsing. In: EMNLP Findings (2022)

[156] Rangan, K., Yin, Y.: A fine-tuning enhanced rag system with quantized influence measure as ai judge. arXiv:2402.17081 (2024)

[157] Saad-Falcon, J., Khattab, O., Santhanam, K., *et al.*: Udapdr: Unsupervised domain adaptation via llm prompting and distillation of rerankers. In: EMNLP (2023)

[158] Wang, L., Yang, N., Wei, F.: Learning to retrieve in-context examples for large language models. arXiv:2307.07164 (2023)

[159] Finardi, P., Avila, L., et al.: The chronicles of rag: The retriever, the chunk and the generator. arXiv:2401.07883 (2024)

[160] Li, J., Yuan, Y., Zhang, Z.: Enhancing llm factual accuracy with rag to counter hallucinations: A case study on domain-specific queries in private knowledge-bases. arXiv:2403.10446 (2024)

[161] Yuan, Y., Shabani, M.A., Liu, S.: Embedding-based context-aware reranker. arXiv preprint arXiv:2510.13329 (2025)

[162] Wang, Z., Araki, J., Jiang, Z., et al.: Learning to filter context for retrieval-augmented generation. arxiv:2311.08377 (2023)

ARTICLE IN PRESS

ARTICLE IN PRESS

[163] Hofstätter, S., Chen, J., Raman, K., Zamani, H.: Fid-light: Efficient and effective retrieval-augmented text generation. In: SIGIR (2023)

[164] Arora, D., Kini, A., Chowdhury, S.R., et al.: Gar-meets-rag paradigm for zero-shot information retrieval. arXiv:2310.20158 (2023)

[165] https://www.pinecone.io

[166] Yu, W., Iter, D., et al.: Generate rather than retrieve: Large language models are strong context generators. arXiv:2209.10063 (2022)

[167] Abdallah, A., Jatowt, A.: Generator-retriever-generator: A novel approach to open-domain question answering. arXiv:2307.11278 (2023)

[168] Besta, M., Kubicek, A., et al.: Multi-head rag: Solving multi-aspect problems with llms. arXiv:2406.05085 (2024)

[169] Saravia, E.: Prompt Engineering Guide. https://github.com/dair-ai/Prompt-Engineering-Guide (2022)

[170] Zheng, H.S., Mishra, S., et al.: Take a step back: Evoking reasoning via abstraction in large language models. arxiv:2310.06117 (2023)

[171] Diao, S., Wang, P., Lin, Y., Zhang, T.: Active prompting with chain-of-thought for large language models. arxiv:2302.12246 (2023)

[172] Jiang, H., Wu, Q., Lin, C., *et al.*: Llmlingua: Compressing prompts for accelerated inference of large language models. In: EMNLP (2023)

[173] Liu, N.F., Lin, K., Hewitt, J., et al.: Lost in the middle: How language models use long contexts. arxiv:2307.03172 (2023)

[174] Ahmed, T., Pai, K.S., Devanbu, P., Barr, E.T.: Automatic semantic augmentation of language model prompts (for code summarization). arXiv:2304.06815 (2024)

[175] Xu, Z., Liu, Z., Liu, Y., et al.: Activerag: Revealing the treasures of knowledge via active learning. arXiv:2402.13547 (2024)

[176] Nijkamp, E., Pang, B., Hayashi, H., et al.: A conversational paradigm for program synthesis. arxiv:2203.13474 (2022)

[177] He, Y., Xia, M., Chen, H., et al.: Animate-a-story: Storytelling with retrieval-augmented video generation. arXiv:2307.06940 (2023)

[178] Hu, E.J., Shen, Y., *et al.*: Lora: Low-rank adaptation of large language models. In: ICLR (2022)

ARTICLE IN PRESS

ARTICLE IN PRESS

[179] Liu, C., Çetin, P., Patodia, Y., et al.: Automated code editing with search-generate-modify. arXiv:2306.06490 (2023)

[180] Joshi, H., Sánchez, J.P.C., Gulwani, S., *et al.*: Repair is nearly generation: Multilingual program repair with llms. In: AAAI (2023)

[181] Jiang, Z., Xu, F.F., et al.: Active retrieval augmented generation. arXiv:2305.06983 (2023)

[182] Mallen, A., Asai, A., Zhong, V., *et al.*: When not to trust language models: Investigating effectiveness of parametric and non-parametric memories. In: ACL (2023)

[183] Jiang, Z., Araki, J., Ding, H., Neubig, G.: How can we know *When* language models know? on the calibration of language models for question answering. TACL (2021)

[184] Kandpal, N., Deng, H., Roberts, A., *et al.*: Large language models struggle to learn long-tail knowledge. In: ICML (2023)

[185] Ren, R., Wang, Y., Qu, Y., et al.: Investigating the factual knowledge boundary of large language models with retrieval augmentation. arxiv:2307.11019 (2023)

[186] Wang, Y., Li, P., Sun, M., Liu, Y.: Self-knowledge guided retrieval augmentation for large language models. In: EMNLP Findings (2023)

[187] Ding, H., Pang, L., Wei, Z., et al.: Retrieve only when it needs: Adaptive retrieval augmentation for hallucination mitigation in large language models. arXiv:2402.10612 (2024)

[188] Jeong, S., Baek, J., Cho, S., et al.: Adaptive-rag: Learning to adapt retrieval-augmented large language models through question complexity. arXiv:2403.14403 (2024)

[189] Zhang, F., Chen, B., *et al.*: Repocoder: Repository-level code completion through iterative retrieval and generation. In: EMNLP (2023)

[190] Shao, Z., Gong, Y., Shen, Y., *et al.*: Enhancing retrieval-augmented large language models with iterative retrieval-generation synergy. In: EMNLP Findings (2023)

[191] Cheng, X., Luo, D., Chen, X., *et al.*: Lift yourself up: Retrieval-augmented text generation with self-memory. In: NeurIPS (2023)

[192] Wang, Z., Liu, A., Lin, H., et al.: Rat: Retrieval augmented thoughts elicit context-aware reasoning in long-horizon generation. arXiv:2403.05313 (2024)

ARTICLE IN PRESS

ARTICLE IN PRESS

[193] Agarwal, O., Ge, H., Shakeri, S., Al-Rfou, R.: Knowledge graph based synthetic corpus generation for knowledge-enhanced language model pre-training. In: NAACL-HLT (2021)

[194] Sun, J., Xu, C., et al.: Think-on-graph: Deep and responsible reasoning of large language model with knowledge graph. arXiv:2307.07697 (2023)

[195] Limkonchotiwat, P., Ponwitayarat, W., *et al.*: Cl-relkt: Cross-lingual language knowledge transfer for multilingual retrieval question answering. In: NAACL Findings (2022)

[196] Asai, A., Yu, X., *et al.*: One question answering model for many languages with cross-lingual dense passage retrieval. In: NeurIPS (2021)

[197] Lee, K., Han, S., *et al.*: When to read documents or QA history: On unified and selective open-domain QA. In: ACL Findings (2023)

[198] Yue, S., Chen, W., et al.: Disc-lawllm: Fine-tuning large language models for intelligent legal services. arXiv:2309.11325 (2023)

[199] Siriwardhana, S., Weerasekera, R., Kaluarachchi, T., *et al.*: Improving the domain adaptation of retrieval augmented generation (RAG) models for open domain question answering. TACL **11**, 1–17 (2023)

[200] Tang, Y., Yang, Y.: Multihop-rag: Benchmarking retrieval-augmented generation for multi-hop queries. arXiv:2401.15391 (2024)

[201] Huang, K., Zhai, C., Ji, H.: CONCRETE: improving cross-lingual fact-checking with cross-lingual retrieval. In: COLING (2022)

[202] Hagström, L., Saynova, D., Norlund, T., et al.: The effect of scaling, retrieval augmentation and form on the factual consistency of language models. arXiv:2311.01307 (2023)

[203] Zamani, H., Bendersky, M.: Stochastic rag: End-to-end retrieval-augmented generation through expected utility maximization. arXiv:2405.02816 (2024)

[204] Liu, Y., Wan, Y., *et al.*: KG-BART: knowledge graph-augmented BART for generative commonsense reasoning. In: AAAI (2021)

[205] Wan, A., Wallace, E., Klein, D.: What evidence do language models find convincing? arXiv:2402.11782 (2024)

[206] Zhang, H., Liu, Z., *et al.*: Grounded conversation generation as guided traverses in commonsense knowledge graphs. In: ACL (2020)

[207] Cai, D., Wang, Y., *et al.*: Skeleton-to-response: Dialogue generation guided by retrieval memory. In: NAACL-HLT (2019)

ARTICLE IN PRESS

ARTICLE IN PRESS

[208] Komeili, M., Shuster, K., Weston, J.: Internet-augmented dialogue generation. In: ACL (2022)

[209] Shuster, K., Xu, J., et al.: Blenderbot 3: a deployed conversational agent that continually learns to responsibly engage. arXiv:2208.03188 (2022)

[210] Kim, S., Jang, J.Y., *et al.*: A model of cross-lingual knowledge-grounded response generation for open-domain dialogue systems. In: EMNLP Findings (2021)

[211] Nie, E., Liang, S., Schmid, H., Schütze, H.: Cross-lingual retrieval augmented prompt for low-resource languages. In: ACL (2023)

[212] Li, X., Nie, E., Liang, S.: From classification to generation: Insights into crosslingual retrieval augmented icl. In: NeurIPS (2023)

[213] Li, W., Li, J., Ma, W., Liu, Y.: Citation-enhanced generation for llm-based chatbot. arXiv:2402.16063 (2024)

[214] Cai, D., Wang, Y., *et al.*: Neural machine translation with monolingual translation memory. In: ACL/IJCNLP (2021)

[215] Khandelwal, U., Fan, A., *et al.*: Nearest neighbor machine translation. In: ICLR (2021)

[216] Du, X., Ji, H.: Retrieval-augmented generative question answering for event argument extraction. In: EMNLP (2022)

[217] Gao, Y., Yin, Q., *et al.*: Retrieval-augmented multilingual keyphrase generation with retriever-generator iterative training. In: NAACL Findings (2022)

[218] Zhang, J., Yu, E.J., Chen, Q., et al.: Retrieval-based full-length wikipedia generation for emergent events. arXiv:2402.18264 (2024)

[219] Fan, R., Fan, Y., Chen, J., et al.: RIGHT: retrieval-augmented generation for mainstream hashtag recommendation. arxiv:2312.10466 (2023)

[220] Wang, Z., Teo, S.X., et al.: M-rag: Reinforcing large language model performance through retrieval-augmented generation with multiple partitions. arXiv:2405.16420 (2024)

[221] Wang, Y., Le, H., Gotmare, A.D., et al.: Codet5mix: A pretrained mixture of encoder-decoder transformers for code understanding and generation (2022)

[222] Madaan, A., Zhou, S., *et al.*: Language models of code are few-shot commonsense learners. In: EMNLP (2022)

[223] Wang, Y., Le, H., Gotmare, A., *et al.*: Codet5+: Open code large language models for code understanding and generation. In: EMNLP (2023)

ARTICLE IN PRESS

ARTICLE IN PRESS

[224] Chen, J., Hu, X., Li, Z., *et al.*: Code search is all you need? improving code suggestions with code search. In: ICSE (2024)

[225] Zan, D., Chen, B., Gong, Y., et al.: Private-library-oriented code generation with large language models. arXiv:2307.15370 (2023)

[226] Liu, M., Yang, T., Lou, Y., *et al.*: Codegen4libs: A two-stage approach for library-oriented code generation. In: ASE (2023)

[227] Liao, D., Pan, S., Huang, Q., et al.: Context-aware code generation framework for code repositories: Local, global, and third-party library awareness. arXiv:2312.05772 (2023)

[228] Li, J., Li, Y., Li, G., *et al.*: Skcoder: A sketch-based approach for automatic code generation. In: ICSE (2023)

[229] Gou, Q., Dong, Y., Wu, Y., Ke, Q.: Rrgcode: Deep hierarchical search-based code generation. Journal of Systems and Software **211**, 111982 (2024)

[230] Zhang, K., Li, J., Li, G., et al.: Codeagent: Enhancing code generation with tool-integrated agent systems for real-world repo-level coding challenges. arXiv:2401.07339 (2024)

[231] Su, H., Jiang, S., Lai, Y., et al.: Arks: Active retrieval in knowledge soup for code generation. arXiv:2402.12317 (2024)

[232] Zhang, K., Li, G., Li, J., et al.: Toolcoder: Teach code generation models to use API search tools. arXiv:2305.04032 (2023)

[233] Liu, S., Chen, Y., Xie, X., *et al.*: Retrieval-augmented generation for code summarization via hybrid GNN. In: ICLR (2021)

[234] Yamaguchi, F., Golde, N., Arp, D., Rieck, K.: Modeling and discovering vulnerabilities with code property graphs. In: S&P (2014)

[235] Choi, Y., Na, C., et al.: Readsum: Retrieval-augmented adaptive transformer for source code summarization. IEEE Access (2023)

[236] Zhao, J., Chen, X., Yang, G., Shen, Y.: Automatic smart contract comment generation via large language models and in-context learning. IST **168**, 107405 (2024)

[237] Alokla, A., Gad, W., Nazih, W., *et al.*: Retrieval-based transformer pseudocode generation. Mathematics **10**(4), 604 (2022)

[238] Xu, J., Cui, Z., *et al.*: Unilog: Automatic logging via LLM and in-context learning. In: ICSE (2024)

ARTICLE IN PRESS

ARTICLE IN PRESS

[239] Wang, H., Xia, X., *et al.*: Context-aware retrieval-based deep commit message generation. TOSEM **30**(4), 56–15630 (2021)

[240] Zhu, X., Sha, C., Niu, J.: A simple retrieval-based method for code comment generation. In: SANER (2022)

[241] Ye, T., Wu, L., Ma, T., et al.: Tram: A token-level retrieval-augmented mechanism for source code summarization. arXiv:2305.11074 (2023)

[242] Li, L., Liang, B., Chen, L., Zhang, X.: Cross-modal retrieval-enhanced code summarization based on joint learning for retrieval and generation. Available at SSRN 4724884

[243] Drain, D., Hu, C., Wu, C., et al.: Generating code with the help of retrieved template functions and stack overflow answers. arXiv:2104.05310 (2021)

[244] Eghbali, A., Pradel, M.: De-hallucinator: Iterative grounding for llm-based code completion. arXiv:2401.01701 (2024)

[245] Liang, M., Xie, X., Zhang, G., et al.: Repofuse: Repository-level code completion with fused dual context. arXiv:2402.14323 (2024)

[246] Shrivastava, D., Kocetkov, D., et al.: Repofusion: Training code models to understand your repository. arXiv:2306.10998 (2023)

[247] Sun, W., Li, H., Yan, M., *et al.*: Revisiting and improving retrieval-augmented deep assertion generation. In: ASE (2023)

[248] Ding, Y., Wang, Z., et al.: Cocomic: Code completion by jointly modeling in-file and cross-file context. arXiv:2212.10007 (2022)

[249] Tang, Z., Ge, J., Liu, S., *et al.*: Domain adaptive code completion via language models and decoupled domain databases. In: ASE (2023)

[250] Tsai, Y., Liu, M., Ren, H.: Rtlfixer: Automatically fixing RTL syntax errors with large language models. arXiv:2311.16543 (2023)

[251] Bogin, B., Gupta, S., Clark, P., et al.: Leveraging code to improve in-context learning for semantic parsing. arXiv:2311.09519 (2023)

[252] Li, H., Zhang, J., Li, C., Chen, H.: Resdsql: Decoupling schema linking and skeleton parsing for text-to-sql. In: AAAI (2023)

[253] Zhang, K., Lin, X., Wang, Y., *et al.*: Refsql: A retrieval-augmentation framework for text-to-sql generation. In: EMNLP Findings (2023)

[254] Chang, S., Fosler-Lussier, E.: Selective demonstrations for cross-domain text-to-sql. arXiv:2310.06302 (2023)

ARTICLE IN PRESS

ARTICLE IN PRESS

[255] Nan, L., Zhao, Y., Zou, W., *et al.*: Enhancing text-to-sql capabilities of large language models: A study on prompt design strategies. In: EMNLP Findings (2023)

[256] Zhang, X., Wang, D., Dou, L., et al.: Multi-hop table retrieval for open-domain text-to-sql. arXiv:2402.10666 (2024)

[257] Li, H., Zhang, J., Liu, H., et al.: Codes: Towards building open-source language models for text-to-sql. arXiv:2402.16347 (2024)

[258] Jie, Z., Lu, W.: Leveraging training data in few-shot prompting for numerical reasoning. arXiv:2305.18170 (2023)

[259] Gao, M., Li, J., Fei, H., et al.: De-fine: Decomposing and refining visual programs with auto-feedback. arXiv:2311.12890 (2023)

[260] Hao, Y., Chen, W., Zhou, Z., Cui, W.: E&v: Prompting large language models to perform static analysis by pseudo-code execution and verification. arXiv:2312.08477 (2023)

[261] Guo, Y., Li, Z., et al.: Retrieval-augmented code generation for universal information extraction. arXiv:2311.02962 (2023)

[262] Pinto, G., Souza, C., et al.: Lessons from building stackspot ai: A contextualized ai coding assistant. arXiv:2311.18450 (2024)

[263] Liu, Z., Chen, C., Wang, J., et al.: Testing the limits: Unusual text inputs generation for mobile app crash detection with large language model. arXiv:2310.15657 (2023)

[264] Bollacker, K.D., Evans, C., *et al.*: Freebase: a collaboratively created graph database for structuring human knowledge. In: SIGMOD (2008)

[265] Patidar, M., Singh, A.K., Sawhney, R., et al.: Combining transfer learning with in-context learning using blackbox llms for zero-shot knowledge base question answering. arXiv:2311.08894 (2023)

[266] Shu, Y., Yu, Z.: Data distribution bottlenecks in grounding language models to knowledge bases. arXiv:2309.08345 (2023)

[267] Leake, D., Crandall, D.J.: On bringing case-based reasoning methodology to deep learning. In: ICCBR (2020)

[268] Zhang, L., Zhang, J., *et al.*: FC-KBQA: A fine-to-coarse composition framework for knowledge base question answering. In: ACL (2023)

[269] Jiang, J., Zhou, K., *et al.*: Structgpt: A general framework for large language model to reason over structured data. In: EMNLP (2023)

ARTICLE IN PRESS

ARTICLE IN PRESS

[270] Baek, J., Aji, A.F., Saffari, A.: Knowledge-augmented language model prompting for zero-shot knowledge graph question answering. arXiv:2306.04136 (2023)

[271] Sen, P., Mavadia, S., Saffari, A.: Knowledge graph-augmented language models for complex question answering. In: NLRSE (2023)

[272] Wu, Y., Hu, N., Bi, S., et al.: Retrieve-rewrite-answer: A kg-to-text enhanced llms framework for knowledge graph question answering. arXiv:2309.11206 (2023)

[273] Wang, C., Xu, Y., Peng, Z., et al.: keqing: knowledge-based question answering is a nature chain-of-thought mentor of LLM. arXiv:2401.00426 (2024)

[274] Liu, J., Cao, S., Shi, J., et al.: Probing structured semantics understanding and generation of language models via question answering. arXiv:2401.05777 (2024)

[275] Xiong, G., Bao, J., Zhao, W.: Interactive-kbqa: Multi-turn interactions for knowledge base question answering with large language models. arXiv:2402.15131 (2024)

[276] Chen, S., Liu, Q., Yu, Z., *et al.*: Retrack: A flexible and efficient framework for knowledge base question answering. In: ACL (2021)

[277] Yu, D., Zhu, C., Fang, Y., *et al.*: Kg-fid: Infusing knowledge graph in fusion-in-decoder for open-domain question answering. In: ACL (2022)

[278] Ju, M., Yu, W., Zhao, T., *et al.*: Grape: Knowledge graph enhanced passage reader for open-domain question answering. In: EMNLP Findings (2022)

[279] Hu, Z., Xu, Y., Yu, W., *et al.*: Empowering language models with knowledge graph reasoning for open-domain question answering. In: EMNLP (2022)

[280] Yang, Q., Chen, Q., Wang, W., *et al.*: Enhancing multi-modal multi-hop question answering via structured knowledge and unified retrieval-generation. In: MM (2023)

[281] Zhao, W., Liu, Y., Niu, T., et al.: DIVKNOWQA: assessing the reasoning ability of llms via open-domain question answering over knowledge base and text. arXiv:2310.20170 (2023)

[282] Wang, X., Yang, Q., Qiu, Y., et al.: Knowledgpt: Enhancing large language models with retrieval and storage access on knowledge bases. arXiv:2308.11761 (2023)

[283] Ko, S., Cho, H., Chae, H., et al.: Evidence-focused fact summarization for knowledge-augmented zero-shot question answering. arXiv:2403.02966 (2024)

[284] Gao, Y., Qiao, L., Kan, Z., et al.: Two-stage generative question answering

ARTICLE IN PRESS

on temporal knowledge graph using large language models. arXiv:2402.16568 (2024)

[285] Guo, T., Yang, Q., Wang, C., et al.: Knowledgenavigator: Leveraging large language models for enhanced reasoning over knowledge graph. arXiv:2312.15880 (2023)

[286] Mavromatis, C., Karypis, G.: Gnn-rag: Graph neural retrieval for large language model reasoning. arXiv:2405.20139 (2024)

[287] Min, S., Boyd-Graber, J., Alberti, C., *et al.*: Neurips 2020 efficientqa competition: Systems, analyses and lessons learned. In: NeurIPS 2020 Competition and Demonstration Track (2021)

[288] Li, A.H., Ng, P., Xu, P., *et al.*: Dual reader-parser on hybrid textual and tabular evidence for open domain question answering. In: ACL/IJCNLP (2021)

[289] Ma, K., Cheng, H., Liu, X., *et al.*: Open-domain question answering via chain of reasoning over heterogeneous knowledge. In: EMNLP Findings (2022)

[290] Christmann, P., Roy, R.S., Weikum, G.: Conversational question answering on heterogeneous sources. In: SIGIR (2022)

[291] Park, E., Lee, S.-M., *et al.*: Rink: reader-inherited evidence reranker for table-and-text open domain question answering. In: AAAI (2023)

[292] Zhao, W., Liu, Y., Wan, Y., et al.: Localize, retrieve and fuse: A generalized framework for free-form question answering over tables. arXiv:2309.11049 (2023)

[293] Pan, F., Canim, M., et al.: End-to-end table question answering via retrieval-augmented generation. arXiv:2203.16714 (2022)

[294] Jiang, Z., Mao, Y., He, P., *et al.*: Omnitab: Pretraining with natural and synthetic data for few-shot table-based question answering. In: NAACL (2022)

[295] Zhong, W., Huang, J., Liu, Q., *et al.*: Reasoning over hybrid chain for table-and-text open domain question answering. In: IJCAI (2022)

[296] Sundar, A.S., Heck, L.: ctbl: Augmenting large language models for conversational tables. arXiv:2303.12024 (2023)

[297] Min, D., Hu, N., Jin, R., et al.: Exploring the impact of table-to-text methods on augmenting llm-based question answering with domain hybrid data. arXiv:2402.12869 (2024)

[298] Roychowdhury, S., Krema, M., et al.: Eratta: Extreme rag for table to answers with large language models. arXiv:2405.03963 (2024)

ARTICLE IN PRESS

ARTICLE IN PRESS

[299] Chen, S.-A., Miculicich, L., Eisenschlos, J., Wang, Z., Wang, Z., Chen, Y., Fujii, Y., Lin, H.-T., Lee, C.-Y., Pfister, T.: Tablerag: Million-token table understanding with language models. Advances in Neural Information Processing Systems **37**, 74899–74921 (2024)

[300] Kim, K., Kim, M., Lee, H., Park, S., Han, Y., Jeon, B.-K.: Thorr: Complex table retrieval and refinement for rag. In: Proceedings of the Workshop Information Retrieval's Role in RAG Systems (IR-RAG 2024) Co-located with the 47th International ACM SIGIR Conference on Research and Development in Information Retrieval, vol. 3784, pp. 50–55 (2024)

[301] Wu, S., Li, Y., Zhang, D., Wu, Z.: Improving knowledge-aware dialogue response generation by using human-written prototype dialogues. In: EMNLP Findings (2020)

[302] Kang, M., Kwak, J.M., *et al.*: Knowledge-consistent dialogue generation with knowledge graphs. In: ICML Workshop (2022)

[303] Ji, Z., Liu, Z., Lee, N., *et al.*: RHO: reducing hallucination in open-domain dialogues with knowledge grounding. In: ACL Findings (2023)

[304] Baek, J., Chandrasekaran, N., Cucerzan, S., et al.: Knowledge-augmented large language models for personalized contextual query suggestion. arXiv:2311.06318 (2023)

[305] He, X., Tian, Y., Sun, Y., et al.: G-retriever: Retrieval-augmented generation for textual graph understanding and question answering. arXiv:2402.07630 (2024)

[306] Hussien, M.M., Melo, A.N., et al.: Rag-based explainable prediction of road users behaviors for automated driving using knowledge graphs and large language models. arXiv:2405.00449 (2024)

[307] Gutiérrez, B.J., Shu, Y., et al.: Hipporag: Neurobiologically inspired long-term memory for large language models. arXiv:2405.14831 (2024)

[308] Sourati, Z., Wang, Z., Liu, M.M., Hu, Y., Guo, M., Bharadwaj, S., Han, K., Sheng, T., Ravi, S., Dehghani, M., et al.: Lad-rag: Layout-aware dynamic rag for visually-rich document understanding. arXiv preprint arXiv:2510.07233 (2025)

[309] Kirstain, Y., Levy, O., Polyak, A.: X&fuse: Fusing visual information in text-to-image generation. arXiv:2303.01000 (2023)

[310] Dhariwal, P., Nichol, A.: Diffusion models beat gans on image synthesis. NeurIPS (2021)

[311] Yang, L., Yu, Z., Meng, C., et al.: Mastering text-to-image diffusion: Recaptioning, planning, and generating with multimodal llms. arXiv:2401.11708

ARTICLE IN PRESS

ARTICLE IN PRESS

(2024)

[312] Zhang, Z., Zhang, A., Li, M., et al.: Multimodal chain-of-thought reasoning in language models. arXiv:2302.00923 (2023)

[313] Xu, C., Yang, M., Ao, X., *et al.*: Retrieval-enhanced adversarial training with dynamic memory-augmented attention for image paragraph captioning. Knowledge-Based Systems **214**, 106730 (2021)

[314] Ramos, R., Elliott, D., Martins, B.: Retrieval-augmented image captioning. In: EACL (2023)

[315] Hu, Z., Iscen, A., Sun, C., *et al.*: Reveal: Retrieval-augmented visual-language pre-training with multi-source multimodal knowledge memory. In: CVPR (2023)

[316] Li, Z., Zhao, W., Du, X., *et al.*: Cross-modal retrieval and semantic refinement for remote sensing image captioning. Remote Sensing **16**(1), 196 (2024)

[317] Yang, Z., Gan, Z., Wang, J., *et al.*: An empirical study of gpt-3 for few-shot knowledge-based vqa. In: AAAI (2022)

[318] Lin, W., Byrne, B.: Retrieval augmented visual question answering with outside knowledge. In: EMNLP (2022)

[319] Fan, A., Gardent, C., Braud, C., Bordes, A.: Augmenting transformers with knn-based composite memory for dialog. TACL **9**, 82–99 (2021)

[320] Liang, Z., Hu, H., Xu, C., *et al.*: Maria: A visual experience powered conversational agent. In: ACL-IJCNLP (2021)

[321] Fang, Q., Feng, Y.: Neural machine translation with phrase-level universal visual representations. In: ACL (2022)

[322] Whitehead, S., Ji, H., Bansal, M., *et al.*: Incorporating background knowledge into video description generation. In: EMNLP (2018)

[323] Yin, C., Tang, J., Xu, Z., Wang, Y.: Memory augmented deep recurrent neural network for video question answering. TNNLS **31**(9), 3159–3167 (2019)

[324] Pan, J., Lin, Z., Ge, Y., *et al.*: Retrieving-to-answer: Zero-shot video question answering with frozen large language models. In: ICCV (2023)

[325] Lei, J., Yu, L., Berg, T.L., Bansal, M.: Tvqa+: Spatio-temporal grounding for video question answering. In: ACL (2020)

[326] Le, H., Chen, N., Hoi, S.: Vgnmn: Video-grounded neural module networks for video-grounded dialogue systems. In: NAACL (2022)

ARTICLE IN PRESS

ARTICLE IN PRESS

[327] Wang, Z., Li, M., Xu, R., *et al.*: Language models with image descriptors are strong few-shot video-language learners. In: NeurIPS (2022)

[328] Yuan, J., Sun, S., Omeiza, D., et al.: Rag-driver: Generalisable driving explanations with retrieval-augmented in-context learning in multi-modal large language model. arXiv:2402.10828 (2024)

[329] Ghosh, S., Kumar, S., Evuru, C.K.R., *et al.*: Recap: retrieval-augmented audio captioning. In: ICASSP (2024)

[330] Elizalde, B., Deshmukh, S., Wang, H.: Natural language supervision for general-purpose audio representations. In: ICASSP (2024)

[331] Kouzelis, T., Katsouros, V.: Weakly-supervised automated audio captioning via text only training. In: DCASE Workshop (2023)

[332] Deshmukh, S., Elizalde, B., Emmanouilidou, D., *et al.*: Training audio captioning models without audio. In: ICASSP (2024)

[333] Wang, Z., Lu, C., Wang, Y., *et al.*: Prolificdreamer: High-fidelity and diverse text-to-3d generation with variational score distillation. In: NeurIPS (2024)

[334] Jia, H., Zhao, P., Wu, H., Gao, Y., Tao, Y., Cui, B.: Learning from history: A retrieval-augmented framework for spatiotemporal prediction. arXiv preprint arXiv:2510.24049 (2025)

[335] Ning, K., Pan, Z., Liu, Y., Jiang, Y., Zhang, J.Y., Rasul, K., Schneider, A., Ma, L., Nevmyvaka, Y., Song, D.: Ts-rag: Retrieval-augmented generation based time series foundation models are stronger zero-shot forecaster. arXiv preprint arXiv:2503.07649 (2025)

[336] Wu, H., Gao, Y., Shu, R., Wang, K., Gou, R., Wu, C., Liu, X., He, J., Cao, S., Fang, J., et al.: Advanced long-term earth system forecasting by learning the small-scale nature. arXiv preprint arXiv:2505.19432 (2025)

[337] Yang, L., Huang, Z., Zhou, X., et al.: Prompt-based 3d molecular diffusion models for structure-based drug design (2023)

[338] Truong Jr, T., Bepler, T.: Poet: A generative model of protein families as sequences-of-sequences. NeurIPS (2024)

[339] Frisoni, G., Mizutani, M., Moro, G., Valgimigli, L.: Bioreader: a retrieval-enhanced text-to-text transformer for biomedical literature. In: EMNLP (2022)

[340] Yang, X., Ye, M., You, Q., et al.: Writing by memorizing: Hierarchical retrieval-based medical report generation. arXiv:2106.06471 (2021)

ARTICLE IN PRESS

ARTICLE IN PRESS

[341] Kim, J., Min, M.: From rag to qa-rag: Integrating generative ai for pharmaceutical regulatory compliance process. arXiv:2402.01717 (2024)

[342] Ji, Y., Li, Z., et al.: Rag-rlrc-laysum at biolaysumm: Integrating retrieval-augmented generation and readability control for layman summarization of biomedical texts. arXiv:2405.13179 (2024)

[343] Yang, K., *et al.*: Leandojo: Theorem proving with retrieval-augmented language models. In: NeurIPS (2024)

[344] Levonian, Z., Li, C., Zhu, W., et al.: Retrieval-augmented generation to improve math question-answering: Trade-offs between groundedness and human preference. arXiv:2310.03184 (2023)

[345] Chen, J., Lin, H., Han, X., Sun, L.: Benchmarking large language models in retrieval-augmented generation. arxiv:2309.01431 (2023)

[346] ES, S., James, J., Anke, L.E., Schockaert, S.: RAGAS: automated evaluation of retrieval augmented generation. arxiv:2309.15217 (2023)

[347] Saad-Falcon, J., Khattab, O., Potts, C., et al.: ARES: an automated evaluation framework for retrieval-augmented generation systems. arxiv:2311.09476 (2023)

[348] https://github.com/truera/trulens

[349] Lyu, Y., Li, Z., Niu, S., et al.: CRUD-RAG: A comprehensive chinese benchmark for retrieval-augmented generation of large language models. arxiv:2401.17043 (2024)

[350] Xiong, G., Jin, Q., Lu, Z., Zhang, A.: Benchmarking retrieval-augmented generation for medicine. arXiv:2402.13178 (2024)

[351] Petroni, F., Piktus, A., *et al.*: Kilt: a benchmark for knowledge intensive language tasks. In: NAACL-HLT (2021)

[352] Yang, X., Sun, K., Xin, H., Sun, Y., Bhalla, N., Chen, X., Choudhary, S., Gui, R.D., Jiang, Z.W., Jiang, Z., *et al.*: Crag-comprehensive rag benchmark. Advances in Neural Information Processing Systems **37**, 10470–10490 (2024)

[353] Pipitone, N., Alami, G.H.: Legalbench-rag: A benchmark for retrieval-augmented generation in the legal domain. arXiv preprint arXiv:2408.10343 (2024)

[354] Wang, S., Tan, J., Dou, Z., Wen, J.-R.: Omnieval: An omnidirectional and automatic rag evaluation benchmark in financial domain. In: Proceedings of the 2025 Conference on Empirical Methods in Natural Language Processing, pp. 5737–5762 (2025)

ARTICLE IN PRESS

ARTICLE IN PRESS

[355] Barnett, S., Kurniawan, S., Thudumu, S., et al.: Seven failure points when engineering a retrieval augmented generation system. arXiv:2401.05856 (2024)

[356] Cuconasu, F., Trappolini, G., Siciliano, F., et al.: The power of noise: Redefining retrieval for RAG systems. arXiv:2401.14887 (2024)

[357] Qiu, L., Shaw, P., Pasupat, P., et al.: Evaluating the impact of model scale for compositional generalization in semantic parsing. arXiv:2205.12253 (2022)

[358] Jagerman, R., Zhuang, H., Qin, Z., et al.: Query expansion by prompting large language models. arxiv:2305.03653 (2023)

[359] Zhang, H., Zhao, P., Miao, X., et al.: Experimental analysis of large-scale learnable vector storage compression. VLDB (2023)

[360] Aksitov, R., Chang, C., Reitter, D., et al.: Characterizing attribution and fluency tradeoffs for retrieval-augmented large language models. arXiv:2302.05578 (2023)

[361] Zhao, Q., Wang, R., Cen, Y., Zha, D., Tan, S., Dong, Y., Tang, J.: Longrag: A dual-perspective retrieval-augmented generation paradigm for long-context question answering. arXiv preprint arXiv:2410.18050 (2024)

[362] Han, C., Wang, Q., Xiong, W., et al.: Lm-infinite: Simple on-the-fly length generalization for large language models. arXiv:2308.16137 (2023)

[363] Chase, H.: LangChain. https://github.com/langchain-ai/langchain (2022)

[364] Jiang, W., Zhang, S., Han, B., et al.: Piperag: Fast retrieval-augmented generation via algorithm-system co-design. arXiv:2403.05676 (2024)

[365] Hu, Z., Murthy, V., Pan, Z., Li, W., Fang, X., Ding, Y., Wang, Y.: Hedrarag: Co-optimizing generation and retrieval for heterogeneous rag workflows. In: Proceedings of the ACM SIGOPS 31st Symposium on Operating Systems Principles, pp. 623–638 (2025)

[366] Ray, S., Pan, R., Gu, Z., Du, K., Feng, S., Ananthanarayanan, G., Netravali, R., Jiang, J.: Metis: Fast quality-aware rag systems with configuration adaptation. In: Proceedings of the ACM SIGOPS 31st Symposium on Operating Systems Principles, pp. 606–622 (2025)

[367] Yao, J., Li, H., Liu, Y., Ray, S., Cheng, Y., Zhang, Q., Du, K., Lu, S., Jiang, J.: Cacheblend: Fast large language model serving for rag with cached knowledge fusion. In: Proceedings of the Twentieth European Conference on Computer Systems, pp. 94–109 (2025)

[368] Jin, C., Zhang, Z., Jiang, X., Liu, F., Liu, S., Liu, X., Jin, X.: Ragcache: Efficient knowledge caching for retrieval-augmented generation. ACM Transactions on

ARTICLE IN PRESS

Computer Systems (2024)

[369] Meduri, K., et al.: Efficient rag framework for large-scale knowledge bases (2024)

[370] Jindal, S.: Did Google Gemini 1.5 Really Kill RAG? https://analyticsindiamag.com/did-google-gemini-1-5-really-kill-rag/ (2024)

[371] Krishnan, N.: Ai agents: Evolution, architecture, and real-world applications. arXiv preprint arXiv:2503.12687 (2025)

[372] Singh, A., Ehtesham, A., Kumar, S., Khoei, T.T.: Agentic retrieval-augmented generation: A survey on agentic rag. arXiv preprint arXiv:2501.09136 (2025)

[373] Xu, Z., Wang, M., Wang, Y., Ye, W., Du, Y., Ma, Y., Tian, Y.: Recon: Reasoning with condensation for efficient retrieval-augmented generation. arXiv preprint arXiv:2510.10448 (2025)

[374] Madaan, A., Tandon, N., Gupta, P., Hallinan, S., Gao, L., Wiegreffe, S., Alon, U., Dziri, N., Prabhumoye, S., Yang, Y., *et al.*: Self-refine: Iterative refinement with self-feedback. Advances in Neural Information Processing Systems **36**, 46534–46594 (2023)

[375] Shinn, N., Cassano, F., Gopinath, A., Narasimhan, K., Yao, S.: Reflexion: Language agents with verbal reinforcement learning. Advances in Neural Information Processing Systems **36**, 8634–8652 (2023)

[376] Wu, P., Zhang, M., Wan, K., Zhao, W., He, K., Du, X., Chen, Z.: Hiprag: Hierarchical process rewards for efficient agentic retrieval augmented generation. arXiv preprint arXiv:2510.07794 (2025)

[377] Fu, Y., Peng, H., Sabharwal, A., Clark, P., Khot, T.: Complexity-based prompting for multi-step reasoning. arXiv preprint arXiv:2210.00720 (2022)

ARTICLE IN PRESS