Chapter 2
Foundations of Generative Information Retrieval

Qingyao Ai, Jingtao Zhan, and Yiqun Liu

Abstract The chapter discusses the foundational impact of modern generative Artificial Intelligence (AI) models on Information Access (IA) systems. In contrast to traditional AI, the large-scale training and superior data modeling of generative AI models enable them to produce high-quality, human-like responses, which bring brand new opportunities for the development of IA paradigms. In this chapter, we identify and introduce two of them in detail, i.e., information generation and information synthesis. Information generation allows AI to create tailored content addressing user needs directly, enhancing user experience with immediate, relevant outputs. Information synthesis leverages the ability of generative AI to integrate and reorganize existing information, providing grounded responses and mitigating issues like model hallucination, which is particularly valuable in scenarios requiring precision and external knowledge. This chapter delves into the foundational aspects of generative models, including architecture, scaling, and training, and discusses their applications in multi-modal scenarios. Additionally, it examines the retrieval-augmented generation paradigm and other methods for corpus modeling and understanding, demonstrating how generative AI can enhance information access systems. It also summarizes potential challenges and fruitful directions for future studies.

The primary distinction between modern generative models and traditional AI techniques lies in their capability to generate complicated and high-quality output based on human instructions. As shown by many studies [1–3], modern generative AI models possess remarkable abilities to generate responses that closely mimic human interaction. Generally speaking, such impressive performance comes from

Qingyao Ai and Jingtao Zhan contributed equally to this work.

Q. Ai (✉) · J. Zhan · Y. Liu
Department of Computer Science and Technology, Tsinghua University, Beijing, China
e-mail: aiqy@tsinghua.edu.cn; zhanjt20@mails.tsinghua.edu.cn; yiqunliu@tsinghua.edu.cn

R. W. White, C. Shah (eds.), *Information Access in the Era of Generative AI*,
The Information Retrieval Series 51, https://doi.org/10.1007/978-3-031-73147-1_2

their large-scale training collections and their advanced data modeling algorithms. Their superior data understanding ability can benefit almost every components of existing information access systems, from document encoding and index construction to query processing and relevance analysis, etc. However, when talking about new opportunities or paradigms that are uniquely brought by the generative AI to information access, they can be broadly categorized in two directions. The first one is to create content that directly addresses the user's information needs. By understanding and taking user queries as input instructions, generative AI models are able to generate specific answers or products tailored to the individual's request. This direct approach to information generation can significantly enhance user experience by providing immediate and relevant responses. The second direction is to leverage the advanced instruction-following capabilities of generative AI models to synthesize and recombine existing information in innovative ways. Generative AI such as Large Language Models (LLMs) can take existing data and transform it into new, coherent pieces of information that may not have been explicitly outlined before. This ability to reinterpret and organize information opens up new possibilities for retrieval system design and applications. Therefore, in this chapter, we discuss how generative AI models could help information access from two perspectives, namely, *information generation* and *information synthesis*.

2.1 Information Generation

Information needs are diverse and typically long-tail. Traditional information retrieval systems, such as search engines and recommendation platforms, are designed to present information that already exists. However, these systems often fall short when it comes to fulfilling the less common information needs. This is particularly evident in scenarios requiring creative creation, where users seek not just information but inspiration and novel ideas. The limitations of traditional information systems in addressing these unique demands have paved the way for the emergence of generative models, which hold the promise of creating new information that aligns closely with long-tail information needs.

In recent years, generative models have made significant developments. For instance, ChatGPT can respond to user questions, Bing enhances its responses with retrieval-augmented generation, Midjourney generates images based on user prompts, and recommendation systems generate personal contents for different users. The development is mainly driven by the capable model architectures, computational resources, and large-scale Internet data. These elements have facilitated the performance of generative models to new heights. With the continuous efforts on scaling up these elements, the model performance is still rapidly improving. Nowadays, generative models have gradually been integrated into various workflows and everyday life activities.

In this section, we present the foundation of generative models. This section is organized as follows: Sect. 2.1.1 shows the efforts on designing the model archi-

tectures for LLMs. Section 2.1.2 discusses how scaling facilitates the development of generative models and its potential future. Section 2.1.3 presents the different training stages of LLMs. Finally, Sect. 2.1.4 introduces how LLMs are used in multimodal scenarios.

2.1.1 Model Architecture

In different generation scenarios like ChatGPT or SoRA, the transformer [4] has emerged as the predominant model structure. It starts with an embedding layer, followed by multiple neural layers. Within each layer, an attention mechanism models the interactions between words, creating contextualized embeddings. The final decision on word generation probabilities is derived by comparing the output embedding with the vocabulary embeddings. We illustrate the model architecture in Fig. 2.1. Unlike traditional recurrent neural networks [5], Transformers are capable of modeling long-distance interactions between words directly, which provides a more powerful representational capability. Numerous enhancements to the transformer architecture have been proposed. In the following, we will explore various modifications to each component of the Transformer, highlighting the advancements that have further improved its efficacy and efficiency.

2.1.1.1 Word Embedding

Word embedding module is at the bottom of the Transformer architecture. Initially, a tokenizer breaks down a sentence into tokens, which the Word embedding module then maps into embeddings. These are combined with position embeddings and fed into subsequent neural layers. Recent research on large-scale language models has

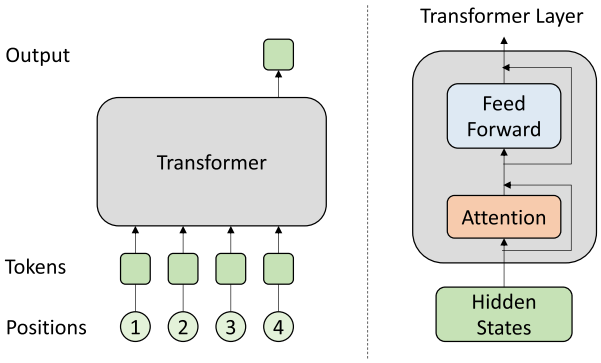

Fig. 2.1 Transformer architecture: overview on the left and the illustration of one layer on the right [4]

identified word embeddings as one of the main sources to training instability [6]. Particularly in the early stages of training, the gradients of word embeddings are often orders of magnitude larger than those of other parameters. To address this issue, [7] introduced a Layer Normalization (LN) immediately after the word embedding layer, stabilizing the distribution effectively. Besides, [6] opted to scale down the gradients of the word embeddings by an order of magnitude to prevent substantial updates. Both approaches have been proven effective in stabilizing the training of language models at the 100 billion parameter scale. Yet, whether they are still effective for larger models remains to be investigated.

2.1.1.2 Position Embedding

Position embedding is essential for Transformers. Unlike RNNs, which inherently process sequences in order, the vanilla attention mechanism disregards the positional distances between words, and the Transformer has to rely on position embeddings for position modeling. Initially, the Transformer [4] utilized sinusoidal embeddings, a non-trainable form of position embedding that is added directly to word embeddings. Later, [8] introduced trainable position embeddings, which is initialized randomly and are updated through gradient descent during training. Subsequently, [9] and [10] proposed relative positioning, where the attention mechanism incorporates biases based on the relative positions of words to better model varying distances. Recently, [11] introduced the concept of rope position embedding, based on the principle that the dot product of vectors correlates with their magnitudes and the angles between them. By rotating vectors in space proportionally to their positions, this method naturally integrates positional information into attention scores. [12] has found that this approach outperforms trainable position embeddings. Yet, these approaches may not work well when extrapolated to long sequences, and more effective methods need to be explored.

2.1.1.3 Attention

The attention mechanism models interactions between words and is a significant component of the Transformer architecture. Enhancements to the attention module have predominantly focused on two aspects: modeling long texts and optimizing the Key-Value (KV) cache. (1) Modeling Long Texts: The vanilla attention mechanism has a complexity of $O(n^2)$, which significantly increases computational costs for long texts. To address this, the Sparse Transformer [13] employs sparse attention, utilizing pre-designed attention patterns to avoid the computation of attention over long sequences. Another approach, Reformer [14], uses Locality-Sensitive Hashing (LSH) to reduce computational complexity. Additionally, [15] compressed context information to shorten sequences, thereby reducing overhead. Others have explored retrieval-based methods [16, 17]. This area of research continues to hold considerable potential for future advancements. (2) Optimizing KV Cache: classic

Transformers use multi-head attention (MHA), which requires storing extensive key-value caches during inference, slowing down model generation. To mitigate this, [18] proposed Multi-Query Attention (MQA), which employs multiple key heads but only a single value head, substantially reducing the key-value cache and enhancing computational speed. However, [19] found that this could degrade model performance, leading to the development of grouped query attention. This method allows multiple key heads to share a single value head, effectively serving as a hybrid between MQA and MHA, balancing computational complexity and performance more effectively. Recently, [20] introduced multi-head latent attention, which compresses keys and values into a single latent space, thereby reducing the key-value cache while maintaining robust representational capacity.

2.1.1.4 Layer Normalization

Layer normalization (LayerNorm) is important for stabilizing the distribution of hidden states, a key to train large language models. In the classical Transformer architecture, LayerNorm is positioned between residual blocks, hence termed Post-LN. Researchers [21] observed that this configuration could lead to high gradients near the output layers and very small gradients near the input layers, resulting in unstable gradients and challenging training dynamics. To address this issue, the Pre-LN configuration was proposed [21], placing LayerNorm on the residual pathways before attention or Feed-Forward Network (FFN) module. Experiments have shown that this adjustment leads to more uniform gradient distribution. Building upon Pre-LN, other researchers introduced Sandwich-LN [22], which adds an additional LayerNorm at the output of the residual pathways, further enhancing the training stability. Beyond merely adjusting the position of LayerNorm, researchers have developed DeepNorm [23], which combines a tailored parameter initialization strategy with modified residual connections to stabilize training. This approach enables the training of Transformers with depths reaching up to 1000 layers. Nevertheless, there still lacks a theoretical understanding about how layer normalization affects the training stability, and more work needs to be done for scaling the model even further.

2.1.2 Scaling

Across different information generation scenarios, scaling has been a significant factor to the performance improvement. It is largely attributed to the discovery of scaling laws [24]. Scaling laws describe how loss decreases in a log-linear manner as model size or training data volume increases. It can be formulated as follows:

$$L(x) = L_{\infty} + k \cdot x^{-\alpha}, \tag{2.1}$$

where L is the loss, x is model size or data size, and k and α are coefficients. This scaling formula has become a crucial theoretical guide in the era of large models, suggesting that performance can be enhanced at a log-linear rate simply by scaling up the model size or training data. Based on these scaling laws, researchers also derived optimal model sizes given fixed computational resources [25]. Their findings indicate that as computational capacity expands, it is beneficial not only to increase the training step but also the model size. This insight has further facilitated the pursuit of large models. The correctness of scaling laws was first proposed in language modeling field and then validated in many other areas, including data mixture scaling laws [26], multimodal scaling laws [27], and scaling laws specific to Information Retrieval (IR) [28].

Despite wide recognition of scaling laws, there remains disagreement among researchers about whether scaling is the correct path to the future. This stems from two main concerns: the uncertain relationship between loss and practical metrics and the inference costs associated with large models.

- **Loss vs. metric improvement:** The first arguing point is whether a linear reduction in loss can translate into super-linear improvements in actual metrics. If metrics could improve super-linearly with linear increases in computational effort, scaling up models would be highly advantageous. However, if the decrease in loss only results in linear or sublinear metric improvements, the diminishing improvements make scaling an inefficient option. The relationship between loss and metric performance remains an open question. Some researchers [29] believe that metrics can improve super-linearly, which is termed emergent abilities. This is further supported by Du et al. [30], who observed a jump in metrics when loss reaches a certain threshold. Additionally, [31] introduced the concept of "grokking" to explain emergence, showing that models might suddenly exhibit strong generalization capabilities when provided with sufficient computational resources. Nevertheless, some researchers [25] argued that such phenomena do not exist, showing that a well-trained smaller model can outperform a larger, undertrained one. Schaeffer et al. [32] demonstrated that emergent abilities are artifacts of discrete metric functions and found that continuous metric functions do not exhibit such behaviors. McKenzie et al. [33] even found that scaling results in worse metric scores. The existence of specific emergent abilities remains unresolved and needs to be investigated in future work.
- **Inference cost considerations:** Early studies on scaling laws did not account for the higher inference costs associated with larger models. Thus, the arguments that larger models are better [25] do not apply when the inference costs are considered. Instead, small models demonstrate potential to lower the inference costs. As shown by Fang et al. [28], the optimal model sizes become significantly smaller when accounting for inference costs. Besides, [34] show that smaller models can utilize more sampling steps during inference and thus perform better. Consequently, many recent studies focus on extensively training small models. For example, Llama [3] and MiniCPM [35] are trained with data

and steps that far exceed the guidance suggested by scaling laws. In the future, the models may be used on a phone to build up intelligent interaction with users. Thus, it is important to develop high-performing small models.

2.1.3 Training

Generative models in different scenarios are similar in training. For example, they usually use autoregressive training objectives, pretraining to Supervised Fine-Tuning (SFT) to Reinforcement Learning from Human Feedback (RLHF) training stages, and prompt tuning procedure. In this section, we focus on the text generation scenario. We first discuss the training objectives and then show the three training stages. Finally, we discuss how to design the prompts after the model is trained.

2.1.3.1 Training Objectives

For generative language models, the training objective is usually next token prediction. However, this was not widely used when Transformers first appeared. Initially, masked language modeling was the prevalent training objective during the bidirectional encoder representations from transformers (BERT) era [8]. It masks 15% of the words in a text randomly, and the model is tasked with predicting these masked words. This approach allows the model to utilize bidirectional attention, enhancing its representational capabilities. Even today, BERT models perform better than autoregressive models on tasks requiring bidirectional attention. However, a significant drawback of this method is the gap between its training setup and downstream tasks, necessitating a fine-tuning phase for adaptation to various applications. Thus, its zero-shot generalization capabilities are very limited.

Next token prediction was developed to address the inability of masked language modeling to generalize zero-shot to downstream tasks. The authors of GPT-2 [36] proposed that all Natural Language Processing (NLP) tasks could be reformulated as next token prediction tasks. By training models on this task, models could be directly applied to any downstream task without the need for specific fine-tuning. In fact, research nowadays demonstrates the effectiveness of this idea. Mathematically, next token prediction can be represented with the following formula:

$$P\left(x_{t+1} \mid x_1, \ldots, x_t\right), \tag{2.2}$$

which is to predict the probability of the next token x_{t+1} given the sequence of previous tokens.

2.1.3.2 Training Stages

The training process of language models typically unfolds in three stages: pre-training, SFT, and RLHF. Each phase presents unique challenges and methodologies.

Pre-training is the most resource-intensive stage. It is training a randomly initialized model on a large dataset to develop a robust linguistic capability. Several challenges arise during this stage: (1) Large models are especially difficult to train from random initialization. During training, there are often spikes in training loss or difficulty in converging [6, 23, 37]. We discussed various architectural improvements in Sect. 2.1.1 to address these instabilities, yet a definitive solution remains an open issue. (2) The computational demand is substantial. Pre-training requires stable and efficient use of computational resources [1]. It often involves parallel processing across multiple machines, which can lead to low utilization rates of computing resources [38]. Zeng et al. [6] reported numerous hardware failures during pre-training. (3) The quality of pre-training data is crucial [39]. Given the vast amount of data needed, efficiently filtering out low-quality data is essential. The filtering methods usually employ neural scoring models and based on the credibility of the site [40, 41].

SFT is to train the model on instruction-response pairs [42]. The model can thus learn to follow instructions or engage in dialogue [3]. To enhance dataset diversity, researchers often leverage different types of NLP tasks. The quality of the dataset is significant and requires a skilled annotation team. Besides, it is also important to label safety-related data, which helps instruct the models to learn to reject inappropriate requests [3].

RLHF focuses on aligning the model with human preferences based on human feedback [43, 44]. The process starts by sampling real human prompts to which the model generates multiple responses. These responses are then compared by users or third-party annotators. A reward model is trained based on these human preferences. Subsequently, reinforcement learning techniques utilize the reward model to guide the model updates. This approach significantly enhances the quality of model outputs, especially in creative writing tasks. However, a major challenge is the generalizability of the reward model; as the model evolves, the reward model may no longer accurately assess the quality of outputs. Continuous iterations of this process are necessary to mitigate this issue [3]. Recently, there are also some offline reinforcement learning algorithms that do not necessitate training a reward model, such as direct preference optimization (DPO) [45]. Yet studies [46] show that such offline learning methods still underperform the online learning methods.

2.1.3.3 Prompt Optimization

Generative models are highly sensitive to input prompts; an effective prompt can significantly enhance the quality of the model's output [47]. Therefore, optimizing

prompts for a generative model is a crucial area of research. Here are three main directions:

- **Designing prompt templates:** Researchers often design prompts that mimic human thought processes to guide the model effectively. This includes using structured thought patterns like chain of thought [48], tree of thought [49], and self-consistency [50], which help the model organize and process information in a logical manner.
- **Iterative optimization of prompt templates:** As with reinforcement learning, this method continuously iterates and refines the prompt templates based on the generation feedback. Given that prompt templates are typically discrete, researchers usually employ large language models to conduct prompt updates [51, 52].
- **Training prompt rewriting models using user interaction logs:** This approach harnesses the rich feedback contained within user interaction logs to tap into user insights. By analyzing how users interact with the model, researchers can train an automated model to rewrite prompts more effectively. This method leverages real-world data to better align the prompts with user intentions and improve the model's responses [53, 54].

2.1.4 Multi-Modal Applications

The rapid advancement of language models has significantly helped progress in the multimodal domain. Language models facilitate the understanding of multimodal data and developments in multimodal generation. We will discuss these two aspects separately.

2.1.4.1 Multi-Modal Understanding

Multimodal understanding involves models processing inputs from multiple modalities to produce relevant textual responses. For example, GPT-4o can process textual, visual, and auditory input. Challenges in this area include designing model structures that can handle multimodal inputs and crafting appropriate training objectives. Here, we focus on how visual signals are integrated into large language models:

In terms of aligning multimodal inputs, there are mainly three approaches:

- **Object detection-based input:** This method involves detecting objects within an image, extracting their features and associated spatial information, and then feeding this data into the language model [55, 56]. While this approach is effective, it tends to be slow due to the processing time required for object detection.
- **Visual encoding:** Another method encodes images directly using a visual encoder, which converts images into a latent vector representation before

integration with the model [57–61]. This method can sometimes result in the loss of detail.

- **Patch-based input:** The most efficient approach involves dividing images into several patches, transforming them with a simple linear layer, and directly inputting them into the model without the need for a complex visual encoder [62].

In terms of training methods, there are mainly four types of training objectives:

- **Contrastive learning or image-text matching:** These tasks require the model to correctly categorize images and their corresponding textual descriptions, aligning the representations of text and images [61, 63, 64].
- **Image captioning:** The model generates captions based on images, which helps it learn to understand the visual content [58–61].
- **Fine-grained image understanding:** The model is tasked to describe specific areas of an image or locate particular objects within an image. This helps enhance the model's detailed comprehension of visual elements [58, 65].
- **Image generation:** This task is reconstructing the original pixels of an image that has been blurred or corrupted [58, 66].

These methodologies and training objectives are crucial for advancing models' capabilities to process and interpret complex multimodal information effectively. This facilitates a more natural interaction with users.

2.1.4.2 Multi-Modal Generation

Multi-modal generation models, such as text-to-image generation, have substantially revolutionized the field of art creation. Traditionally, Generative Adversarial Networks (GAN) [67] and autoregressive methods [68] are mainstream methods. However, they are computationally expensive and cannot produce high-quality results. Recently, diffusion [69, 70] emerges as a new state-of-the-art method in multimodal generation. It perturbs the data with noise and learns to reconstruct the original data.

Language models are increasingly applied in the multimodal generation domain, such as in image [71, 72] and video generation [73, 74]. Language models are primarily utilized for processing training data and reformulating prompts.

In terms of training data, the titles associated with real-world images or videos often contain significant noise. If generative models are trained directly on these noisy titles, it could lead to inaccurate semantic understanding. To address this, language models can be used to filter and regenerate text descriptions within the training data [75, 76]. For instance, a multimodal understanding model could first be trained and then used to relabel videos or images to obtain more precise and detailed text descriptions. Experimental results have shown that this method significantly improves the fidelity of model generations to prompts.

During inference, multimodal generation models are highly sensitive to the input prompts. Many users do not know how to craft effective prompts and thus get

unsatisfying responses [77]. As a result, it is common to train a language model to rewrite user-provided prompts to enhance the quality of the generated images [75]. One of the challenges here is the difficulty in annotating such rewriting training data, as even system developers may not always know the optimal prompts, let alone crowdsourced workers [78]. To overcome this, some researchers collect a large number of user-shared effective prompts as training data [79]. Others build prompt-rewriting models based on user log data, capturing preferences, and feedback for training [53].

2.2 Information Synthesis

Other than generating information directly, another important research and application direction is to use the power of generative AI models, particularly LLMs, to integrate existing information and generate grounded responses accordingly. For simplicity, we refer to this paradigm as *information synthesis*. The key difference between information generation and information synthesis is the source of information. Information generation relies on the internal knowledge and information gathered through the training of generative AI models to create the model outputs, while information synthesis requires external sources to provide information to the models, and the models serve more as a integrator than a creator. There are multiple reasons why information synthesis is considered more reliable than generation in several IA scenarios. Here we discuss two of the most significant ones, i.e., model hallucination and external knowledge.

Hallucinating, which refers to the behavior of generative AI models that create responses and outputs that are not grounded by facts or existing supporting materials, is rooted in the foundation of most existing generative AI systems. For instance, LLMs create responses based on the next token prediction task, which formulates the generation of language as a probabilistic process and generates the next token in the output based on a probabilistic distribution (over the vocabulary) predicted by neural networks [1, 3]. The probabilistic model of LLMs allows them to capture knowledge in large-scale data efficiently and effectively, but it also introduces inevitable variance in their generation process. In other words, it is well acknowledged that it is theoretically impossible to prevent LLMs from generating data that are not seen in their training process [80]. While the ability of hallucinating is the source of creativity for LLMs (and for humans as well), it is not always desirable in practice, particularly for tasks with high requirements on result precision, reliability, and explainability. Therefore, asking the generative AI models to integrate human-created or factually grounded materials instead of generating information on their own is often considered more effective and robust to hallucination-sensitive applications.

The need for external knowledge is another key reason why we may prefer information synthesis over information generation. Despite the fact that modern generative AI models are trained with an incredibly large amount of data gathered

from the Web, there are many cases where we still need to retrieve and find support from external knowledge collections to finish certain tasks. Examples include the use of private datasets, vertical domain applications that require special knowledge, tasks that involve time-sensitive data, etc. It is usually inefficient or prohibitive to update large-scale generative AI models such as LLMs with task-oriented external data through model pre-training or SFT [81–83]. Even if possible, such paradigm is not preferred because the internal knowledge structures of most generative AI models are still a mystery (at least of today), and there is no guarantee that the models could behave and use the external information as we expect. In contrast, using generative AI models as information synthesizer gives us not only more flexibility but also more transparency and control over system outputs.

In this section, we discuss how generative AI models, particularly LLMs, can serve as effective information synthesizers for IA. We start with introducing one of the most popular information synthesis paradigm, i.e., Retrieval-Augmented Generation (RAG), and then discuss several other directions that utilize LLMs for corpus modeling and understanding.

2.2.1 Retrieval-Augmented Generation

RAG refers to the process of augmenting LLMs with data retrieved from external collections or synthesizing multiple retrieval results with LLMs for downstream applications [84, 85]. While the popularity of RAG rose after the release of large-scale pre-trained language models such as GPT [1] and BART [86], relevant topics and techniques have already been studied for at least more than two decades in both the IR and NLP communities, e.g., extractive and abstractive summarization that generates summary based on retrieved sentences [87, 88] or answer extraction from top retrieved document [89]. A major reason why RAG-like techniques were not as attractive as they are today is the limited performance of generative models before the era of LLMs. After ChatGPT [1] demonstrated superior ability text generation at the end of 2022, there have been many studies and surveys on RAG and its applications in LLMs [84, 90, 91]. As the intent of this chapter is not to provide yet another survey on existing RAG papers, we focus the following discussions on several present and future directions for RAG and their relations underneath.

2.2.1.1 Naive RAG

Naive RAG refers to the paradigm that directly feeds documents or other types of information retrieved by a retrieval system to the input (e.g., prompts) of a generative AI model and hope that the model can generate better output with or without a specific target task [92]. It is also referred to as the "Retrieve-then-Read" framework that has been used in reading comprehension and text summarization before LLMs hit the world [93]. Given an input (could be a query or a specific

task instruction), we first retrieve relevant information (usually entities, passages, or documents) from an external corpus or previous inputs (e.g., the memory of an agent [94, 95]) with a retrieval system. Then, we craft a input prompt with the retrieval results and feed it to the LLM. The LLM will generate the final response based on the input request and the retrieved information. This paradigm has already been proven to be effective in multiple IA tasks such as question answering [85].

Since LLMs are purely used as black-box tools to process the retrieved documents and input request in naive RAG, existing studies in this direction mainly focus on the development of better retrieval systems and prompt design for RAG. The studies on retrieval systems, unsurprisingly, are highly similar to those in IR, which involve indexing, query processing, first-stage retrieval, re-ranking, etc. These topics and system components have already been studied in the IR community for more than five decades. Perhaps the most notable difference is that recent studies on naive RAG often prefer the use of neural retrieval models (e.g., dense retrieval models [96]) over traditional term-matching models (e.g., BM25 [97]). An important reason behind this is that neural retrieval models share similar theoretical background and model structures with LLMs. This makes joint optimization possible in modern RAG systems, which we discuss in Sect. 2.2.1.3.

The design of input prompts with retrieval results, on the other hand, is relatively more under-explored before the rise of LLMs. It has been well recognized that prompt formats, even when the contents are same, could significantly affect the performance of LLMs. How to feed retrieval results effectively into the prompts of LLMs for RAG has thus attracted a lot of attention recently [93, 98, 99]. Studies have found that LLMs exhibit significant position bias over the input result sequences [100, 101] and has different perspectives on relevance with human experts [102]. Since prompts are the main interaction interface between retrieval and generation, their design principles and downstream effects on naive RAG are of great value both in research and real-world applications. Particularly, how to craft effective RAG prompts automatically could be a fruitful direction to explore. Existing studies have shown that high-quality prompt writers can be automatically learned based on downstream task performance and user logs in image generation [53], and it is widely believed that similar techniques have also been used in popular LLM chatbots [103]. Yet how to do this for RAG remains to be a question to be answered.

2.2.1.2 Modular RAG

In contrast to naive RAG methods, modular RAG treats retrieval systems as functional modules to support LLMs [104]. While some works view this retrieval module as one type of many tools that can be learned and used by LLMs [105], it is widely acknowledged that retrieval systems possess an irreplaceable position in modern LLM applications due to its diverse nature and significant importance [84]. Broadly speaking, existing studies on using retrieval systems as functional modules

for LLM generation mainly focus on the three "W" questions, namely, *when to retrieve*, *what to retrieve*, and *where to retrieve*.

The question of *when to retrieve* refers to the timing of functional call for retrieval systems. In contrast to LLMs that directly create responses based on their internal parameter space without explicit evidence grounding, retrieval systems produce reliable and explainable information directly by searching external corpus. From this perspective, the best timing to call the retrieval system is when LLMs start to hallucinate or produce wrong results. Yet identifying such timing is difficult because we neither know the correct answers in advance or understand the internal mechanisms of LLMs (at least as of today) [106]. One naive yet effective method is to retrieve supporting evidence for LLM inference with a fixed time interval, such as every fixed number of generated tokens [107, 108] or every sentence [109]. More advanced paradigms involve the analysis of knowledge boundary [110] and the estimation of prediction uncertainty in LLMs [106, 111]. Theoretically speaking, since the study of *when to retrieve* shares similar motivations and foundations with the study of hallucination detection, existing studies on LLM hallucination [112, 113] could provide important inspiration for research on this topic. Promising directions include better fact-checking systems for LLMs [114] and more investigations on how to characterize the confidence and uncertainty of LLM predictions based on both external behavior and internal state analysis [111].

The question of *what to retrieve* focuses on analyzing the intents and information needs of LLMs in inference. LLMs often need the help of different tools and systems to finish different tasks [105]. However, in contrast to other tools widely studied in tool learning, retrieval itself is a complicated systems with dynamic and free-form inputs, data collections, and outputs. Therefore, understanding what exactly is needed by LLMs and how to formulate it in the language of retrieval systems is an important problem. Most existing studies on RAG naively use the whole or local context of LLM inference as the queries to retrieval systems and assume that these context contain enough information to guide retrieval [90]. A slightly better solution is to use the terms that LLMs have low confidence to formulate queries since uncertain tokens represent cases where LLMs have limited knowledge to generate responses and thus need more information [106]. As long studied in the IR community, the formulation of an effective query requires deep understanding of the user's intent, and many of the important context information behind a user intent is not explicitly expressed in the words they wrote [115]. Therefore, a more theoretically principled method to answer *what to retrieve* in RAG is to analyze the internal state of LLMs and infer their information needs directly. For example, Su et al. [111] directly formulate queries based on the internal attention distribution of LLMs (Fig. 2.2) and improve the performance of RAG for nearly 20% on several benchmark datasets without changing the retrieval system. This demonstrates the potential of future studies in this direction.

Where to retrieve refers to the question of how to identify the correct information sources for RAG. Studies in this direction are particularly related to the research on multi-source retrieval [116] and tool learning [105]. To answer different requests related to the use of information collected from different databases or data

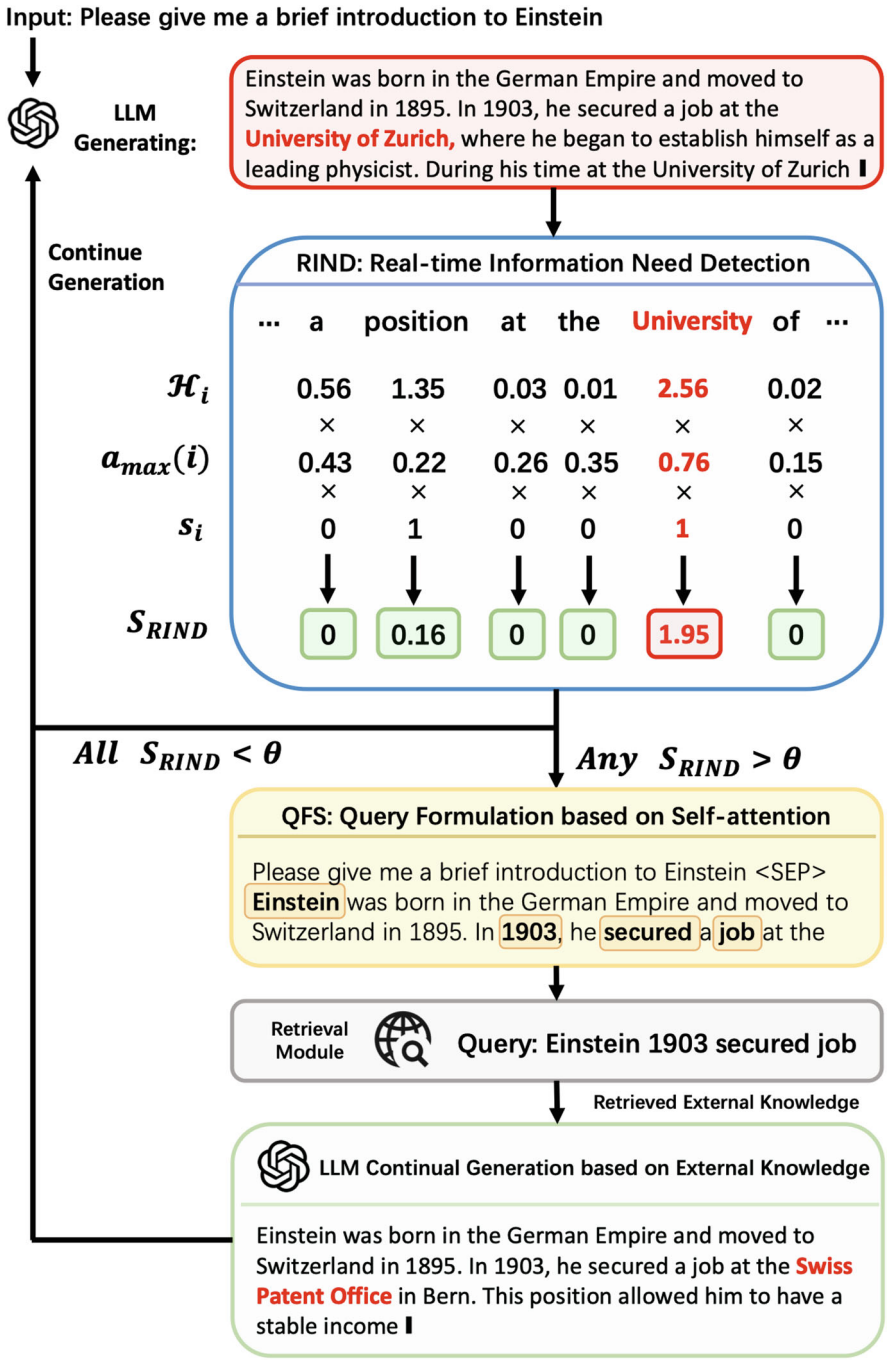

Fig. 2.2 Su et al. [111] generate queries for RAG based on the internal attention distribution of LLMs

collections, LLMs need to learn how to interact with each information sources effectively and efficiently. Studies of tool learning focus on teaching LLMs to use tools according to the context, and retrieval systems are usually considered as one type of tools to use. However, retrieval itself could be a complicated problem when we possess multiple data collections with different characteristics. In search engines, information sources are broadly categorized based on their modality, and we usually build separate systems for each of them (e.g., the "Images," "News," "Videos" tabs on Google). While commercial search engines may aggregate results from different sources into a single page, the ultimate Search Engine Result Page (SERP) shown to users are just a list of results, and it is up to the users to decide which they want to see and how to use these results for downstream applications. In contrast, when using LLMs, users often request LLMs to directly answer their question instead of listing a couple of candidates [117, 118], so it is the job of LLMs to decide where to retrieve the information given the current context. While studies of how to navigate user queries to search indexes built from different information sources have been widely studied in the IR community [119–122], how to do it for RAG with modern generative AI models is, to the best of our knowledge, still underexplored. Existing literature on RAG mostly works on a single retrieval collection (usually a text corpus), but it is obvious that no single collection can satisfy the needs of LLMs in different tasks. For instance, when writing a legal case document, the judge needs to collect and organize information from evidences, complaints, counterclaims, court records, as well as legal articles and previous cases. How to navigate the generation model to retrieve and integrate information from different sources jointly for downstream applications is a practical and potentially fruitful research question for RAG.

2.2.1.3 Optimization of Retrieval and Generation

As discussed in several RAG surveys [84, 90], the optimization of RAG systems usually involves the optimization of three components, i.e., the retriever, the generator, and the augmentation method. If we further step back and look at the high-level goals of RAG optimization, we could also categorize it based on how we evaluate the RAG system, namely, the evaluation from the perspectives of retrievers, generators, or the joint systems. The evaluation from the retriever perspectives is not particularly different from existing studies on ranking evaluation. The underlining assumption of this is that once the LLMs are fed with the passages or documents that contain the correct information, they should be able to produce the correct answers directly. Therefore, the evaluation and optimization of a RAG system could downgrade to the evaluation and optimization of a classic retrieval/ranking systems, to where most existing works on dense retrieval and Learning to Rank (LTR) could be applied [123, 124]. Yet there are still differences between RAG and traditional retrieval tasks as the queries are no longer issued by users. How to formulate queries efficiently and effectively from LLMs for the retriever is a worthy research question,

and studies in this direction have already shown potential in improving the overall quality of RAG systems [111].

From the perspective of generators, RAG evaluation and optimization focus more on improving the robustness and effectiveness of LLM generation based on a fixed set of retrieval results [108]. This often means extra training or fine-tuning on LLMs to improve their fundamental ability in information processing. For example, retrieved documents could be lengthy, and LLMs are usually not good at processing long input context [101]. Therefore, how to design efficient LLMs that can take long context inputs efficiently and effectively has been a popular research problem that have been widely studied by researchers from both academia and industry [100]. We have seen many companies show off their models based on how many input tokens they can process in one request. In addition, since retrieval results are fed as a part of the LLM inputs, whether the LLMs can generate the response based on the retrieved documents instead of their internal knowledge could be seen as a special type of instruction-following ability. Studies have been conducted to teach LLMs to utilize retrieval results faithfully and constantly in RAG systems [125] On the other hand, factors such as irrelevant results and ranking perturbations are well acknowledged to be harmful for the performance of generators in RAG, so there are also studies that try to improve the robustness of LLMs from the perspective of RAG. For example, [126] proposes to fine-tune LLMs with the presence of retrieval results (i.e., retrieval-augmented fine tuning) so that LLMs can learn the domain-specific knowledge introduced by the retriever and improve their robustness against potential distracting information from retrieval.

From the perspective of augmentation methods, existing research mostly focuses on the joint optimization of the RAG system as a whole. In other words, the loss functions of RAG optimization should be built from the performance metrics of downstream tasks directly. While this paradigm is appealing, it often has strict requirements on the design of RAG systems. Particularly, it is difficult to apply such joint optimization algorithms on a RAG system in which retrievers and generators are loosely connected through prompts constructed from discrete retrieval results. While reinforcement learning could solve the problem in theory, its empirical performance when being used as the solo optimization algorithms for ranking systems is still not satisfying at this point [127]. If you already have a good retriever and only conduct fine-tuning with a fixed LLM, then it may work [128], but this still does not look like a perfect solution because reinforcement learning is usually subject to large variance in practice. To the best of our knowledge, how to directly connect the training of retrievers with the auto-regressive loss of the generators in RAG is still an open question. Answering this question requires us to go deep into the structure of generative AI models and retrieval models and develop new model structures that can take advantages from studies on both sides.

2.2.1.4 Retrieval Planning and Composite Information Needs

As discussed above, the initial motivation behind the studies of RAG mostly focuses on using the power of retrieval systems to improve the quality of responses generated by LLMs in terms of reliability and informativeness. While it is widely acknowledged that problems such as hallucination and high computation cost in supervised fine-tuning will continue to be significant for generative AI models in a short period of time, there are also concerns, especially from the IR community, that retrieval could become less important with the rapid evolution of LLMs [129]. In fact, ChatGPT has already shown similar accuracy and better user satisfaction on factoid question answering than traditional Web search engines [1]. However, the rise of generative AI models also brings brand new opportunities for IR. One of them is the possibility of moving from SERPs that simply list result candidates to a real information agent that solve complicated tasks with composite information needs.

Today, most people treat IR systems as *unit information solvers*. Despite their actual task characteristics, users first decompose their goals into a couple of unit information needs (usually expressed with separate queries) and then issue them one by one to search engines or recommendation systems to find the corresponding answers. An important reason behind the popularity of this paradigm is that, at least of today, IR systems are not capable of doing complicated information tasks with composite needs and multi-step planning. For example, we can use a search engine to find a survey on RAG by searching "survey of RAG," but cannot write such a survey directly by retrieving and analyzing papers from publication collections. The job of information need decomposition and retrieval planning has always been human's.

Fortunately, with the help of generative AI models such as LLMs, it is now possible to push the boundary of IR systems and tackle such advanced information tasks for users. Composite retrieval is not a new concept in IR [130], but previous studies refer to the phrase as retrieval paradigms that cluster results from multiple sources and show them in groups for specific user queries [131]. While this represents one type of composite needs, it is relatively simple as the target user queries usually are mostly topic specific and keyword based. Complicated information tasks such as survey generation and professional document writing often involve multi-step planning and multi-round interactions between the retrieval results and response generation. To build powerful IR systems or agents that can solve such composite information tasks, we need to construct collaborative systems that deeply connect the retrieval, planning, and generation. For instance, we need to conduct generation-oriented retrieval optimization to build retrieval framework and model interfaces for downstream task planner and response generators; we also need to design retrieval-oriented generation models that can decompose information needs, navigate the retrieval process, and gather information from multiple sources to generate the final results. Research on these directions could be fruitful and significantly extend the scope of IR in the era of generative AI.

2.2.2 *Corpus Modeling and Understanding*

In contrast to using RAG, another line of studies try to use generative AI models to replace traditional retrieval systems. Directly answering a user's information need instead of showing ten blue links has long been an important goal for the development of intelligent IR systems [132]. With the rise of LLMs, such vision is now achievable in a significant extent. For example, LLM-based chatbots such as ChatGPT can answer multiple types of user queries with direct answers [118]. Metzler et al. [133] has discussed several paradigms in which pre-trained language models can help IR systems answer a user's information needs directly without listing references. The intuition is to use neural network-based language models to store the corpus knowledge in parameter space and pull relevant answers or information directly from it based on user's queries. Depending on how the problem is formulated, several research directions have emerged. Specifically, in this section, we discuss two of them, namely, Generative Retrieval (GR) and domain-specific modeling.

2.2.2.1 Generative Retrieval

The idea of *generative retrieval* comes from the idea of differentiable index proposed by Metzler et al. [133]. The original name used in the paper was *model-based IR*, but after the rise of generative AI models, some researchers start to refer to studies in this direction as generative retrieval (GR). The core idea of GR is twofold, i.e., the differentiable index and the generation of doc IDs.

Inspired by the superior performance of pre-trained language models, particularly BERT [8] and GPT [1], generative retrieval wants to explore the possibility of replacing traditional term-based index (e.g., inverted index) in retrieval systems with large-scale neural networks. In contrast to dense retrieval models that build neural encoders to project documents to latent semantic spaces and build explicit indexes based on document vectors, GR tries to build implicit indexes in the parameter space of neural networks. For instance, Differentiable Search Indexing (DSI) and its variations [134–137] have tried to train pretrained language models on the target corpus directly and then treat the model's parameter as an "index" of the corpus. Studies in this direction argue that by training the neural models to encode the whole corpus, documents and information would be implicitly stored in the parameters of the models, and these parameter-based indexes have better storage efficiency than traditional term-based or vector-based indexes [134]. They also argue that such paradigm can unify the multi-stage retrieval pipeline so that indexes can be trained directly for the final retrieval objectives. However, storing raw document content directly in limited parameter spaces often lead to significant information loss (which is reflected in the suboptimal retrieval performance of GR models [138]), and using model parameters as indexes makes the whole system uncontrollable by both system developers and users. While the former could be alleviated by using large-

scale models, the latter is still an unresolved problem for GR. For example, it is difficult, if not impossible, to remove or update a document indexed in the parameter space when we do not know what exactly each parameter do in the neural models. Considering that dense retrieval models built with product quantization and inverted file systems can achieve state-of-the-art retrieval performance with similar latency and less storage than term-based models with inverted indexes [139], whether the idea of differentiable indexes in GR is worth its price is still a controversial question.

Another important characteristic of GR models is to retrieve documents by generating sequences of doc IDs through autoregression. Since documents are stored implicitly in model parameters, to actually retrieve a real document, GR models use user's queries as prompts to generate document IDs, which usually consist of a couple of special tokens that exclusively identify each relevant document. Since the birth of GR, a variety of document IDs have been proposed, which can be broadly categorized as IDs with explicit tokens [134, 135, 137] and IDs with implicit tokens [136, 140, 141]. GR models with explicit ID tokens try to label each document with sequences of real terms that have semantic or numerical meanings. Examples include keyword-based doc IDs and tree-based doc IDs [134]. Compared to vectors in dense retrieval, these methods have less flexibility and capability in document modeling as they discretize document semantic meanings with a limited number of tokens, and their retrieval performance is usually poor [140]. However, they have better explainability than other neural retrieval models because their doc ID tokens are constructed from real words or document clusters. To avoid the theoretical limitation of explicit token IDs and grant GR models with the same modeling capacity of dense retrieval models, several studies have proposed to build implicit token IDs with latent vectors [136, 140, 141]. The idea is to represent each document with a sequence of latent vectors so that fine-grained semantic information would not be lost. These types of GR models are highly similar to existing dense retrieval models since both of them represent each document with latent vectors. The major difference is that the former uses a sequence of vectors from a learned codebook constructed in training, while the latter builds separate vectors for each document directly from their raw content. [142] have proved that GR models with implicit tokens are equal to a multi-vector dense retrieval models in theory. Also, the use of a learned codebook for implicit token vectors is theoretically the same with a dense retrieval system that uses cluster-based product quantization [139, 143]. Therefore, the performance upper bound of GR (with implicit tokens) and dense retrieval is the same in theory. While some believe that GR models could have lower latency as they don't need to search among millions of documents on the fly, this is a questionable argument because the inference of a large-scale neural model is usually much slower than a vector-based search on distributed systems. Also, the maintenance of data in a neural model is much more complex than it is in a vector-based database. Perhaps the future potential of GR does not lay in retrieval effectiveness or efficiency but some other perspectives such as explainability.

2.2.2.2 Domain-Specific Modeling

LLMs, particularly those with instruction tuning, can respond to user's queries directly. This exactly matches the initiative of a long-standing vision of IR systems to directly answer user's need without listing a couple of documents [133]. Therefore, ever since the rise of ChatGPT, there has been a serious discussion on whether LLMs are future search engines in practice [129]. Yet apart from the hallucination problem discussed in previous sections, there are other challenges that prevent generative AI models like LLMs to serve as a major information accessing tool for modern users. One of them is how to teach LLMs to understand and use knowledge from external corpus not included in their initial training process. If we treat each external corpus as a domain-specific dataset, then the studies in this direction are essentially the same with the construction of domain-specific LLMs. While RAG can help LLMs adapt to new domains quickly, their performance is limited when the understanding of input documents from the external corpus requires domain knowledge that the LLMs do not possess in advance [83].

To solve the above problem and build usable IA systems with LLMs on domain-specific data, one of the most popular method is to conduct continued pre-training or supervised fine-tuning of LLMs on the target domain corpus. The idea is to apply similar training strategies used in model pre-training on the new corpus so that LLMs can better capture knowledge in the new domain. Example studies in this direction include techniques on data selection [82] and tokenizers adaptation [144] that directly use the target corpus to train LLMs. Many domain-specific LLMs have been developed, including legal LLMs, financial LLMs, etc. [145–147] The continued pre-training of LLMs on external corpus has been shown to be effective on many domain-specific tasks such as domain QA and text generation. However, modeling external corpus through this method may not be preferred in practice when we do not have enough computation resources to train LLMs or cannot access the parameters of them. Also, till the end of the today, the internal knowledge structure and learning mechanism of LLMs are still unknown, and applying naive continued pre-training algorithms on external corpus could hurt the performance of LLMs in unexpected way. Therefore, researchers have designed several knowledge editing techniques on LLMs to explore the possibility of injecting knowledge with no or low cost on the general effectiveness of LLMs [148, 149]. Studies in this direction are still in an early stage as most existing methods only work on fixed and limited updating rules and knowledge entity triples [150], but it could be fruitful in the future since domain adaptation and external corpus modeling is a wide need of LLM applications in practice.

Besides continued pre-training, another paradigm to model external corpus and domain knowledge is to build separate language models for each corpus and combine them with the large general LLMs to form a collaborative system. The intuition behind this is relevant to the idea of LLM agents where each LLM could serve different roles in the system to accomplish tasks together. It is widely acknowledged that the emergence of abilities only present in large-scale models [29], but training models with such large scale (e.g., GPT-4 [1]) is

usually prohibitive, even with parameter efficient algorithms [151]. Inspired by the superior instruction following ability of LLMs, researchers have explored the possibility of building small models for external corpus modeling and use them to communicate domain-specific knowledge to large general LMs [83]. In other words, the small models can serve as domain knowledge "consultants," while large general models can serve as the decision-makers that finish domain-specific tasks based on the guidance of the small models. Experiments have shown that such a paradigm can improve black-box LLMs' performance on domain-specific tasks with low cost and high flexibility. While the overall idea of prompt general LLMs with domain-specific prompts is similar to the framework of RAG, building an actual LM for corpus modeling enables us to capture implicit domain knowledge (e.g., the fine-grained differences between law articles [152]) and potentially save tokens in prompts. There are concerns on whether this paradigm is still worthy when we have more powerful LLMs that include more domain-specific data in training. However, since many users prefer to keep their data private to themselves due to multiple safety and privacy concerns, this paradigm and RAG could continue to be appealing in practice.

2.3 Summary and Future Directions

In this chapter, we introduce the foundations and applications of generative AI models in information accessing. Instead of analyzing how generative AI models like LLMs could improve the existing modules of search engines and recommendation systems, we focus on how they could revolutionize information access with new methodologies and system design. Particularly, we discuss two new paradigms brought by generative AI models, namely, information generation and information synthesis.

Information generation refers to scenarios where users can use generative AI models to create information that directly satisfies their information needs. Here, we delved into the core components of generative models, including model architectures (with a focus on Transformers and their improvements), scaling laws, and training methodologies. We examined the debates surrounding continual model scaling, the importance of prompt optimization, and the extension of these models to multi-modal applications for information access.

Information synthesis refers to the paradigm that utilizes the superior instruction-following and logic-reasoning ability of LLMs to aggregate and synthesize existing information. We extensively discuss one of the most representative techniques, i.e., RAG, on this direction, and introduce various approaches from naive implementations to more sophisticated modular systems. We describe the challenges and opportunities in optimizing RAG systems, highlighting the need for joint retrieval-generation optimization and the potential of several relevant research directions such as composite retrieval with planning. Besides RAG, we also discuss some alternative paradigms that use generative AI models to model corpus knowledge directly,

such as generative retrieval, which aims to replace traditional indexing methods with neural network-based approaches, and domain-specific model training, which conducts continued pre-training or fine-tuning on LLMs with the target corpus. We discussed the potential and limitations of these approaches, including issues of system controllability and cost efficiency.

Overall, research on how generative AI models could reshape modern information access systems is still at an early stage today. As discussed above, existing studies on information generation and information synthesis either focus on simple information tasks (such as writing a poem, answering a factoid question, etc.) or reply on simple system design (e.g., feeding all documents to LLMs as prompts) that obviously cannot fully exploit the power of modern retrieval and generation models. Therefore, we believe that there are two major directions worth exploring in the next couple of years (at least). The first one is to move from simple and unit information retrieval tasks (e.g., factoid question answering) to more complicated information tasks that used to be "impossible" for modern IR systems. Examples include retrieval with composite needs (e.g., "help me plan a wedding in Amherst, MA") or tasks that require planning and multiple rounds of retrieval and generations (e.g., "write a survey on RAG"). These tasks are used to require human experts to decompose the needs and conduct retrieval, analysis, and result aggregations. With the help of generative AI, accomplishing them automatically with machines is now possible. The second direction is to explore better techniques to communicate, collaborate, or even unify retrieval and generation systems for information access. While studies of RAG have attracted considerable attention, existing works mostly use retrieval systems as plug-in tools for LLMs without digging into their internal connections and differences. Examples such as how to understand the information needs of LLMs, how to communicate the retrieved results to LLMs, and how to optimize generators for retrieval and retriever for generation are all important yet underexplored research topics. There are many questions related to each of these topics that are worthy of detailed investigation, including the design of new training paradigms, the development of agent-like system frameworks, potential problems and bias introduced by off-policy and on-policy training for the joint system, etc.

When ChatGPT first arrived, people from the IR community were worried that such generative AI models could overthrow all existing IR systems and crush everything in the field [129], as it has almost happened in NLP. Interestingly, in simulated social experiments on human-AI competitions, [153] found that if human producers do not extend their capacities with the help of generative AI, they will eventually be "replaced" by AI. From this perspective, the future of IR research in the era of generative AI lies in how to extend the scope of IR with generative AI models to finish more complicated information tasks and develop more general system architectures that do not just retrieve a list of documents but also perform more sophisticated information processing and planning.

References

1. OpenAI: GPT-4 technical report. CoRR abs/2303.08774 (2023). https://doi.org/10.48550/ARXIV.2303.087742303.08774
2. Zhao, W.X., Zhou, K., Li, J., Tang, T., Wang, X., Hou, Y., Min, Y., Zhang, B., Zhang, J., Dong, Z., Du, Y., Yang, C., Chen, Y., Chen, Z., Jiang, J., Ren, R., Li, Y., Tang, X., Liu, Z., Liu, P., Nie, J., Wen, J.: A survey of large language models. CoRR abs/2303.18223 (2023). https://doi.org/10.48550/ARXIV.2303.182232303.18223
3. Touvron, H., Lavril, T., Izacard, G., Martinet, X., Lachaux, M.-A., Lacroix, T., Rozière, B., Goyal, N., Hambro, E., Azhar, F., et al.: Llama: Open and efficient foundation language models. arXiv preprint arXiv:2302.13971 (2023)
4. Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A.N., Kaiser, L.U., Polosukhin, I.: Attention is all you need. In: Guyon, I., Luxburg, U.V., Bengio, S., Wallach, H., Fergus, R., Vishwanathan, S., Garnett, R. (eds.) Advances in Neural Information Processing Systems, vol. 30. Curran Associates, Inc., New York (2017)
5. Schuster, M., Paliwal, K.K.: Bidirectional recurrent neural networks. IEEE Trans. Signal Process. **45**(11), 2673–2681 (1997)
6. Zeng, A., Liu, X., Du, Z., Wang, Z., Lai, H., Ding, M., Yang, Z., Xu, Y., Zheng, W., Xia, X., et al.: Glm-130b: An open bilingual pre-trained model. arXiv preprint arXiv:2210.02414 (2022)
7. Le Scao, T., Fan, A., Akiki, C., Pavlick, E., Ilić, S., Hesslow, D., Castagné, R., Luccioni, A.S., Yvon, F., Gallé, M., et al.: Bloom: A 176b-parameter open-access multilingual language model (2023)
8. Devlin, J., Chang, M., Lee, K., Toutanova, K.: BERT: pre-training of deep bidirectional transformers for language understanding. In: NAACL-HLT (1), pp. 4171–4186. Association for Computational Linguistics, New York (2019)
9. Raffel, C., Shazeer, N., Roberts, A., Lee, K., Narang, S., Matena, M., Zhou, Y., Li, W., Liu, P.J.: Exploring the limits of transfer learning with a unified text-to-text transformer. J. Mach. Learn. Res. **21**, 140–114067 (2020)
10. Press, O., Smith, N.A., Lewis, M.: Train short, test long: Attention with linear biases enables input length extrapolation. arXiv preprint arXiv:2108.12409 (2021)
11. Su, J., Lu, Y., Pan, S., Wen, B., RoFormer, Y.L.: Enhanced transformer with rotary position embedding (2021). https://doi.org/10.1016/j. neucom (2023)
12. Black, S., Biderman, S., Hallahan, E., Anthony, Q., Gao, L., Golding, L., He, H., Leahy, C., McDonell, K., Phang, J., et al.: Gpt-neox-20b: an open-source autoregressive language model. arXiv preprint arXiv:2204.06745 (2022)
13. Child, R., Gray, S., Radford, A., Sutskever, I.: Generating long sequences with sparse transformers. arXiv preprint arXiv:1904.10509 (2019)
14. Kitaev, N., Kaiser, L., Levskaya, A.: Reformer: the efficient transformer. arXiv preprint arXiv:2001.04451 (2020)
15. Munkhdalai, T., Faruqui, M., Gopal, S.: Leave no context behind: Efficient infinite context transformers with infini-attention. arXiv preprint arXiv:2404.07143 (2024)
16. Grave, E., Joulin, A., Usunier, N.: Improving neural language models with a continuous cache. arXiv preprint arXiv:1612.04426 (2016)
17. Izacard, G., Grave, E.: Leveraging Passage Retrieval with Generative Models for Open Domain Question Answering. arXiv (2020). https://arxiv.org/abs/2007.0128
18. Shazeer, N.: Fast transformer decoding: One write-head is all you need. arXiv preprint arXiv:1911.02150 (2019)
19. Ainslie, J., Lee-Thorp, J., Jong, M., Zemlyanskiy, Y., Lebrón, F., Sanghai, S.: GQA: training generalized multi-query transformer models from multi-head checkpoints. arXiv preprint arXiv:2305.13245 (2023)
20. DeepSeek-AI: DeepSeek-V2: A Strong, Economical, and Efficient Mixture-of-Experts Language Model (2024)

21. Xiong, R., Yang, Y., He, D., Zheng, K., Zheng, S., Xing, C., Zhang, H., Lan, Y., Wang, L., Liu, T.: On layer normalization in the transformer architecture. In: International Conference on Machine Learning, pp. 10524–10533. PMLR, New York (2020)
22. Ding, M., Yang, Z., Hong, W., Zheng, W., Zhou, C., Yin, D., Lin, J., Zou, X., Shao, Z., Yang, H., et al.: CogView: Mastering text-to-image generation via transformers. Adv. Neural Inf. Proces. Syst. **34**, 19822–19835 (2021)
23. Wang, H., Ma, S., Dong, L., Huang, S., Zhang, D., Wei, F.: Deepnet: Scaling transformers to 1,000 layers. In: IEEE Transactions on Pattern Analysis and Machine Intelligence (2024)
24. Kaplan, J., McCandlish, S., Henighan, T., Brown, T.B., Chess, B., Child, R., Gray, S., Radford, A., Wu, J., Amodei, D.: Scaling laws for neural language models. arXiv preprint arXiv:2001.08361 (2020)
25. Hoffmann, J., Borgeaud, S., Mensch, A., Buchatskaya, E., Cai, T., Rutherford, E., Casas, D.d.L., Hendricks, L.A.,Welbl, J., Clark, A., et al.: Training compute-optimal large language models. arXiv preprint arXiv:2203.15556 (2022)
26. Ye, J., Liu, P., Sun, T., Zhou, Y., Zhan, J., Qiu, X.: Data mixing laws: optimizing data mixtures by predicting language modeling performance. arXiv preprint arXiv:2403.16952 (2024)
27. Henighan, T., Kaplan, J., Katz, M., Chen, M., Hesse, C., Jackson, J., Jun, H., Brown, T.B., Dhariwal, P., Gray, S., et al.: Scaling laws for autoregressive generative modeling. arXiv preprint arXiv:2010.14701 (2020)
28. Fang, Y., Zhan, J., Ai, Q., Mao, J., Su, W., Chen, J., Liu, Y.: Scaling laws for dense retrieval. arXiv preprint arXiv:2403.18684 (2024)
29. Wei, J., Tay, Y., Bommasani, R., Raffel, C., Zoph, B., Borgeaud, S., Yogatama, D., Bosma, M., Zhou, D., Metzler, D., et al.: Emergent abilities of large language models. arXiv preprint arXiv:2206.07682 (2022)
30. Du, Z., Zeng, A., Dong, Y., Tang, J.: Understanding emergent abilities of language models from the loss perspective. arXiv preprint arXiv:2403.15796 (2024)
31. Power, A., Burda, Y., Edwards, H., Babuschkin, I., Misra, V.: Grokking: generalization beyond overfitting on small algorithmic datasets. arXiv preprint arXiv:2201.02177 (2022)
32. Schaeffer, R., Miranda, B., Koyejo, S.: Are emergent abilities of large language models a mirage? Adv. Neural Inf. Proces. Syst. **36**, 1–13 (2024)
33. McKenzie, I.R., Lyzhov, A., Pieler, M., Parrish, A., Mueller, A., Prabhu, A., McLean, E., Kirtland, A., Ross, A., Liu, A., et al.: Inverse scaling: when bigger isn't better. arXiv preprint arXiv:2306.09479 (2023)
34. Mei, K., Tu, Z., Delbracio, M., Talebi, H., Patel, V.M., Milanfar, P.: Bigger is not always better: Scaling properties of latent diffusion models. arXiv preprint arXiv:2404.01367 (2024)
35. Hu, S., Tu, Y., Han, X., He, C., Cui, G., Long, X., Zheng, Z., Fang, Y., Huang, Y., Zhao, W., et al.: MiniCPM: unveiling the potential of small language models with scalable training strategies. arXiv preprint arXiv:2404.06395 (2024)
36. Radford, A., Wu, J., Child, R., Luan, D., Amodei, D., Sutskever, I., et al.: Language models are unsupervised multitask learners. OpenAI blog **1**(8), 9 (2019)
37. Zhang, S., Roller, S., Goyal, N., Artetxe, M., Chen, M., Chen, S., Dewan, C., Diab, M., Li, X., Lin, X.V., et al.: OPT: open pre-trained transformer language models. arXiv preprint arXiv:2205.01068 (2022)
38. Chowdhery, A., Narang, S., Devlin, J., Bosma, M., Mishra, G., Roberts, A., Barham, P., Chung, H.W., Sutton, C., Gehrmann, S., et al.: Palm: scaling language modeling with pathways. J. Mach. Learn. Res. **24**(240), 1–113 (2023)
39. Gunasekar, S., Zhang, Y., Aneja, J., Mendes, C.C.T., Del Giorno, A., Gopi, S., Javaheripi, M., Kauffmann, P., Rosa, G., Saarikivi, O., et al.: Textbooks are all you need. arXiv preprint arXiv:2306.11644 (2023)
40. Yang, A., Xiao, B., Wang, B., Zhang, B., Bian, C., Yin, C., Lv, C., Pan, D., Wang, D., Yan, D., et al.: Baichuan 2: open large-scale language models. arXiv preprint arXiv:2309.10305 (2023)

41. Bi, X., Chen, D., Chen, G., Chen, S., Dai, D., Deng, C., Ding, H., Dong, K., Du, Q., Fu, Z., et al.: Deepseek LLM: scaling open-source language models with longtermism. arXiv preprint arXiv:2401.02954 (2024)
42. Chung, H.W., Hou, L., Longpre, S., Zoph, B., Tay, Y., Fedus, W., Li, E., Wang, X., Dehghani, M., Brahma, S., Webson, A., Gu, S.S., Dai, Z., Suzgun, M., Chen, X., Chowdhery, A., Valter, D., Narang, S., Mishra, G., Yu, A.W., Zhao, V., Huang, Y., Dai, A.M., Yu, H., Petrov, S., Chi, E.H.-h., Dean, J., Devlin, J., Roberts, A., Zhou, D., Le, Q.V., Wei, J.: Scaling instruction-finetuned language models. ArXiv abs/2210.11416 (2022)
43. Ouyang, L., Wu, J., Jiang, X., Almeida, D., Wainwright, C., Mishkin, P., Zhang, C., Agarwal, S., Slama, K., Ray, A., et al.: Training language models to follow instructions with human feedback. Adv. Neural Inf. Proces. Syst. **35**, 27730–27744 (2022)
44. Schulman, J.,Wolski, F., Dhariwal, P., Radford, A., Klimov, O.: Proximal Policy Optimization Algorithms (2017)
45. Rafailov, R., Sharma, A., Mitchell, E., Manning, C.D., Ermon, S., Finn, C.: Direct preference optimization: your language model is secretly a reward model. In: Oh, A., Neumann, T., Globerson, A., Saenko, K., Hardt, M., Levine, S. (eds.) Advances in Neural Information Processing Systems, vol. 36, pp. 53728–53741. Curran Associates, Inc., New York (2023)
46. Xu, S., Fu, W., Gao, J., Ye, W., Liu, W., Mei, Z., Wang, G., Yu, C., Wu, Y.: Is DPO superior to PPO for LLM alignment? a comprehensive study. arXiv preprint arXiv:2404.10719 (2024)
47. Liu, P., Yuan, W., Fu, J., Jiang, Z., Hayashi, H., Neubig, G.: Pre-train, prompt, and predict: a systematic survey of prompting methods in natural language processing. ACM Comput. Surv. **55**(9), 1–35 (2023)
48. Wei, J., Wang, X., Schuurmans, D., Bosma, M., Xia, F., Chi, E., Le, Q.V., Zhou, D., et al.: Chain-of-thought prompting elicits reasoning in large language models. Adv. Neural Inf. Proces. Syst. **35**, 24824–24837 (2022)
49. Yao, S., Yu, D., Zhao, J., Shafran, I., Griffiths, T., Cao, Y., Narasimhan, K.: Tree of thoughts: Deliberate problem solving with large language models. Adv. Neural Inf. Proces. Syst. **36**, 1–14 (2024)
50. Wang, X., Wei, J., Schuurmans, D., Le, Q., Chi, E., Narang, S., Chowdhery, A., Zhou, D.: Self-consistency improves chain of thought reasoning in language models. In: 11th International Conference on Learning Representations (ICLR 2023), pp. 1–15. arXiv preprint arXiv:2203.11171 (2023)
51. Zhou, Y., Muresanu, A.I., Han, Z., Paster, K., Pitis, S., Chan, H., Ba, J.: Large language models are human-level prompt engineers. arXiv preprint arXiv:2211.01910 (2022)
52. Yang, C., Wang, X., Lu, Y., Liu, H., Le, Q.V., Zhou, D., Chen, X.: Large language models as optimizers. arXiv preprint arXiv:2309.03409 (2023)
53. Zhan, J., Ai, Q., Liu, Y., Chen, J., Ma, S.: Capability-aware prompt reformulation learning for text-to-image generation. arXiv preprint arXiv:2403.19716 (2024)
54. Zhan, J., Ai, Q., Liu, Y., Pan, Y., Yao, T., Mao, J., Ma, S., Mei, T.: Prompt refinement with image pivot for text-to-image generation. In: ACL (2024)
55. Lu, J., Batra, D., Parikh, D., Lee, S.: Vilbert: Pretraining task-agnostic visiolinguistic representations for vision-and-language tasks. In: Proceedings of the 33rd International Conference on Neural Information Processing Systems, pp. 13–23. (2019)
56. Chen, Y.-C., Li, L., Yu, L., El Kholy, A., Ahmed, F., Gan, Z., Cheng, Y., Liu, J.: Uniter: universal image-text representation learning. In: European Conference on Computer Vision, pp. 104–120. Springer, Berlin (2020)
57. Huang, Z., Zeng, Z., Liu, B., Fu, D., Fu, J.: Pixel-BERT: aligning image pixels with text by deep multi-modal transformers. arXiv preprint arXiv:2004.00849 (2020)
58. Wang, P., Yang, A., Men, R., Lin, J., Bai, S., Li, Z., Ma, J., Zhou, C., Zhou, J., Yang, H.: OFA: unifying architectures, tasks, and modalities through a simple sequence-to-sequence learning framework. In: International Conference on Machine Learning, pp. 23318–23340. PMLR, New York (2022)

59. Alayrac, J.-B., Donahue, J., Luc, P., Miech, A., Barr, I., Hasson, Y., Lenc, K., Mensch, A., Millican, K., Reynolds, M., et al.: Flamingo: a visual language model for few-shot learning. Adv. Neural Inf. Proces. Syst. **35**, 23716–23736 (2022)
60. Wang, W., Lv, Q., Yu, W., Hong, W., Qi, J., Wang, Y., Ji, J., Yang, Z., Zhao, L., Song, X., et al.: CogVLM: visual expert for pretrained language models. arXiv preprint arXiv:2311.03079 (2023)
61. Li, J., Li, D., Savarese, S., Hoi, S.: Blip-2: Bootstrapping language-image pretraining with frozen image encoders and large language models. In: International Conference on Machine Learning, pp. 19730–19742. PMLR, New York (2023)
62. Kim, W., Son, B., Kim, I.: Vilt: vision-and-language transformer without convolution or region supervision. In: International Conference on Machine Learning, pp. 5583–5594. PMLR, New York (2021)
63. Li, J., Selvaraju, R., Gotmare, A., Joty, S., Xiong, C., Hoi, S.C.H.: Align before fuse: vision and language representation learning with momentum distillation. Adv. Neural Inf. Proces. Syst. **34**, 9694–9705 (2021)
64. Radford, A., Kim, J.W., Hallacy, C., Ramesh, A., Goh, G., Agarwal, S., Sastry, G., Askell, A., Mishkin, P., Clark, J., et al.: Learning transferable visual models from natural language supervision. In: International Conference on Machine Learning, pp. 8748–8763. PMLR, New York (2021)
65. Yu, T., Yao, Y., Zhang, H., He, T., Han, Y., Cui, G., Hu, J., Liu, Z., Zheng, H.-T., Sun, M., et al.: RLHF-V: towards trustworthy MLLMs via behavior alignment from fine-grained correctional human feedback. arXiv preprint arXiv:2312.00849 (2023)
66. Bao, H., Dong, L., Piao, S., Wei, F.: Beit: Bert pre-training of image transformers. arXiv preprint arXiv:2106.08254 (2021)
67. Reed, S., Akata, Z., Yan, X., Logeswaran, L., Schiele, B., Lee, H.: Generative adversarial text to image synthesis. In: International Conference on Machine Learning, pp. 1060–1069. PMLR, New York (2016)
68. Ramesh, A., Pavlov, M., Goh, G., Gray, S., Voss, C., Radford, A., Chen, M., Sutskever, I.: Zero-shot text-to-image generation. In: International Conference on Machine Learning, pp. 8821–8831. PMLR, New York (2021)
69. Nichol, A., Dhariwal, P., Ramesh, A., Shyam, P., Mishkin, P., McGrew, B., Sutskever, I., Chen, M.: Glide: towards photorealistic image generation and editing with text-guided diffusion models. arXiv preprint arXiv:2112.10741 (2021)
70. Rombach, R., Blattmann, A., Lorenz, D., Esser, P., Ommer, B.: High-resolution image synthesis with latent diffusion models. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, pp. 10684–10695 (2022)
71. Ho, J., Jain, A., Abbeel, P.: Denoising diffusion probabilistic models. Adv. Neural Inf. Proces. Syst. **33**, 6840–6851 (2020)
72. Zhang, C., Zhang, C., Zhang, M., Kweon, I.S.: Text-to-image diffusion model in generative ai: A survey. arXiv preprint arXiv:2303.07909 (2023)
73. Peebles, W., Xie, S.: Scalable diffusion models with transformers. In: Proceedings of the IEEE/CVF International Conference on Computer Vision, pp. 4195–4205 (2023)
74. Singh, A.: A survey of ai text-to-image and ai text-to-video generators. In: 2023 4th International Conference on Artificial Intelligence, Robotics and Control (AIRC), pp. 32–36. IEEE, New York (2023)
75. Betker, J., Goh, G., Jing, L., Brooks, T., Wang, J., Li, L., Ouyang, L., Zhuang, J., Lee, J., Guo, Y., et al.: Improving image generation with better captions. Computer Science. **2**(3), 8 (2023). https://cdn.openai.com/papers/dall-e-3.pdf
76. Brooks, T., Peebles, B., Holmes, C., DePue, W., Guo, Y., Jing, L., Schnurr, D., Taylor, J., Luhman, T., Luhman, E., Ng, C., Wang, R., Ramesh, A.: Video generation models as world simulators (2024)
77. Oppenlaender, J.: A taxonomy of prompt modifiers for text-to-image generation. In: *Behaviour & Information Technology*, pp. 1–14

78. Liu, V., Chilton, L.B.: Design guidelines for prompt engineering text-to-image generative models. In: Proceedings of the 2022 CHI Conference on Human Factors in Computing Systems, pp. 1–23 (2022)
79. Hao, Y., Chi, Z., Dong, L., Wei, F.: Optimizing prompts for text-to-image generation. Adv. Neural Inf. Proces. Syst. **36**, 1–17 (2024)
80. Ji, Z., Lee, N., Frieske, R., Yu, T., Su, D., Xu, Y., Ishii, E., Bang, Y.J., Madotto, A., Fung, P.: Survey of hallucination in natural language generation. ACM Comput. Surv. **55**(12), 1–38 (2023)
81. Arefeen, M.A., Debnath, B., Chakradhar, S.: Leancontext: Cost-efficient domain-specific question answering using LLMs. Nat. Lang. Process. J. **7**, 100065 (2024)
82. Aharoni, R., Goldberg, Y.: Unsupervised domain clusters in pretrained language models. arXiv preprint arXiv:2004.02105 (2020)
83. Li, H., Ai, Q., Chen, J., Dong, Q., Wu, Z., Liu, Y., Chen, C., Tian, Q.: Blade: Enhancing black-box large language models with small domain-specific models. arXiv preprint arXiv:2403.18365 (2024)
84. Gao, Y., Xiong, Y., Gao, X., Jia, K., Pan, J., Bi, Y., Dai, Y., Sun, J., Wang, H.: Retrieval-augmented generation for large language models: A survey. arXiv preprint arXiv:2312.10997 (2023)
85. Lewis, P., Perez, E., Piktus, A., Petroni, F., Karpukhin, V., Goyal, N., Küttler, H., Lewis, M., Yih, W.-t., Rocktäschel, T., et al.: Retrieval-augmented generation for knowledge-intensive NLP tasks. Adv. Neural Inf. Proces. Syst. **33**, 9459–9474 (2020)
86. Lewis, M., Liu, Y., Goyal, N., Ghazvininejad, M., Mohamed, A., Levy, O., Stoyanov, V., Zettlemoyer, L.: Bart: denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension. In: Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics, pp. 7871–7880 (2020)
87. Moratanch, N., Chitrakala, S.: A survey on extractive text summarization. In: 2017 International Conference on Computer, Communication and Signal Processing (ICCCSP), pp. 1–6. IEEE, New York (2017)
88. Lin, H., Ng, V.: Abstractive summarization: a survey of the state of the art. In: Proceedings of the AAAI Conference on Artificial Intelligence, vol. 33, pp. 9815–9822 (2019)
89. Bajaj, P., Campos, D., Craswell, N., Deng, L., Gao, J., Liu, X., Majumder, R., McNamara, A., Mitra, B., Nguyen, T., Rosenberg, M., Song, X., Stoica, A., Tiwary, S., Wang, T.: MS MARCO: A Human Generated MAchine Reading COmprehension Dataset (2018)
90. Zhao, P., Zhang, H., Yu, Q., Wang, Z., Geng, Y., Fu, F., Yang, L., Zhang, W., Cui, B.: Retrieval-augmented generation for ai-generated content: A survey. arXiv preprint arXiv:2402.19473 (2024)
91. Asai, A., Min, S., Zhong, Z., Chen, D.: Retrieval-based language models and applications. In: Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 6: Tutorial Abstracts), pp. 41–46 (2023)
92. Guu, K., Lee, K., Tung, Z., Pasupat, P., Chang, M.: Retrieval augmented language model pre-training. In: International Conference on Machine Learning, pp. 3929–3938. PMLR, New York (2020)
93. Ma, X., Gong, Y., He, P., Zhao, H., Duan, N.: Query rewriting for retrieval augmented large language models. arXiv preprint arXiv:2305.14283 (2023)
94. Wang, L., Ma, C., Feng, X., Zhang, Z., Yang, H., Zhang, J., Chen, Z., Tang, J., Chen, X., Lin, Y., et al.: A survey on large language model based autonomous agents. Front. Comp. Sci. **18**(6), 186345 (2024)
95. Zhang, Z., Bo, X., Ma, C., Li, R., Chen, X., Dai, Q., Zhu, J., Dong, Z., Wen, J.-R.: A Survey on the Memory Mechanism of Large Language Model based Agents (2024)
96. Zhan, J., Mao, J., Liu, Y., Guo, J., Zhang, M., Ma, S.: Optimizing dense retrieval model training with hard negatives. In: Proceedings of the 44th International ACM SIGIR Conference on Research and Development in Information Retrieval, pp. 1503–1512 (2021)
97. Robertson, S., Zaragoza, H., et al.: The probabilistic relevance framework: Bm25 and beyond. Found. Trends Inf. Retr. **3**(4), 333–389 (2009)

98. Mao, S., Jiang, Y., Chen, B., Li, X., Wang, P., Wang, X., Xie, P., Huang, F., Chen, H., Zhang, N.: Rafe: ranking feedback improves query rewriting for rag. arXiv preprint arXiv:2405.14431 (2024)
99. Chan, C.-M., Xu, C., Yuan, R., Luo, H., Xue, W., Guo, Y., Fu, J.: RQ-RAG: learning to refine queries for retrieval augmented generation. arXiv preprint arXiv:2404.00610 (2024)
100. Li, T., Zhang, G., Do, Q.D., Yue, X., Chen, W.: Long-context LLMs struggle with long in-context learning. arXiv preprint arXiv:2404.02060 (2024)
101. Liu, N.F., Lin, K., Hewitt, J., Paranjape, A., Bevilacqua, M., Petroni, F., Liang, P.: Lost in the middle: how language models use long contexts. Trans. Assoc. Comput. Linguistics **12**, 157–173 (2024)
102. Faggioli, G., Dietz, L., Clarke, C.L., Demartini, G., Hagen, M., Hauff, C., Kando, N., Kanoulas, E., Potthast, M., Stein, B., et al.: Perspectives on large language models for relevance judgment. In: Proceedings of the 2023 ACM SIGIR International Conference on Theory of Information Retrieval, pp. 39–50 (2023)
103. Liu, P., Yuan, W., Fu, J., Jiang, Z., Hayashi, H., Neubig, G.: Pre-train, prompt, and predict: A systematic survey of prompting methods in natural language processing. ACM Comput. Surv. **55**(9) (2023). https://doi.org/10.1145/3560815
104. Wang, X., Yang, Q., Qiu, Y., Liang, J., He, Q., Gu, Z., Xiao, Y., Wang, W.: KnowledGPT: Enhancing large language models with retrieval and storage access on knowledge bases. arXiv preprint arXiv:2308.11761 (2023)
105. Qin, Y., Hu, S., Lin, Y., Chen, W., Ding, N., Cui, G., Zeng, Z., Huang, Y., Xiao, C., Han, C., Fung, Y.R., Su, Y., Wang, H., Qian, C., Tian, R., Zhu, K., Liang, S., Shen, X., Xu, B., Zhang, Z., Ye, Y., Li, B., Tang, Z., Yi, J., Zhu, Y., Dai, Z., Yan, L., Cong, X., Lu, Y., Zhao, W., Huang, Y., Yan, J., Han, X., Sun, X., Li, D., Phang, J., Yang, C., Wu, T., Ji, H., Liu, Z., Sun, M.: Tool Learning with Foundation Models (2023)
106. Jiang, Z., Xu, F., Gao, L., Sun, Z., Liu, Q., Dwivedi-Yu, J., Yang, Y., Callan, J., Neubig, G.: Active retrieval augmented generation. In: Bouamor, H., Pino, J., Bali, K. (eds.) Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing, pp. 7969–7992. Association for Computational Linguistics, Singapore (2023). https://doi.org/10.18653/v1/2023.emnlp-main.495. https://aclanthology.org/2023.emnlp-main.495
107. Ram, O., Levine, Y., Dalmedigos, I., Muhlgay, D., Shashua, A., Leyton-Brown, K., Shoham, Y.: In-context retrieval-augmented language models. Trans. Assoc. Comput. Linguistics **11**, 1316–1331 (2023)
108. Borgeaud, S., Mensch, A., Hoffmann, J., Cai, T., Rutherford, E., Millican, K., Van Den Driessche, G.B., Lespiau, J.-B., Damoc, B., Clark, A., et al.: Improving language models by retrieving from trillions of tokens. In: International Conference on Machine Learning, pp. 2206–2240. PMLR, New York (2022)
109. Trivedi, H., Balasubramanian, N., Khot, T., Sabharwal, A.: Interleaving retrieval with chain-of-thought reasoning for knowledge-intensive multi-step questions. In: Rogers, A., Boyd-Graber, J., Okazaki, N. (eds.) Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), pp. 10014–10037. Association for Computational Linguistics, Toronto, Canada (2023). https://doi.org/10.18653/v1/2023.acl-long.557. https://aclanthology.org/2023.acl-long.557
110. Ni, S., Bi, K., Guo, J., Cheng, X.: When do LLMs need retrieval augmentation? mitigating LLMs' overconfidence helps retrieval augmentation. arXiv preprint arXiv:2402.11457 (2024)
111. Su, W., Tang, Y., Ai, Q., Wu, Z., Liu, Y.: DRAGIN: Dynamic Retrieval Augmented Generation based on the Information Needs of Large Language Models (2024)
112. Su, W., Wang, C., Ai, Q., HU, Y., Wu, Z., Zhou, Y., Liu, Y.: Unsupervised Real-Time Hallucination Detection based on the Internal States of Large Language Models (2024)
113. Liu, T., Zhang, Y., Brockett, C., Mao, Y., Sui, Z., Chen, W., Dolan, B.: A token-level reference-free hallucination detection benchmark for free-form text generation. arXiv preprint arXiv:2104.08704 (2021)

114. Fadeeva, E., Rubashevskii, A., Shelmanov, A., Petrakov, S., Li, H., Mubarak, H., Tsymbalov, E., Kuzmin, G., Panchenko, A., Baldwin, T., et al.: Fact-checking the output of large language models via token-level uncertainty quantification. arXiv preprint arXiv:2403.04696 (2024)
115. Cronen-Townsend, S., Croft, W.B., et al.: Quantifying query ambiguity. In: Proceedings of HLT, vol. 2, pp. 94–98 (2002)
116. Arens, Y., Chee, C.Y., Hsu, C.-N., Knoblock, C.A.: Retrieving and integrating data from multiple information sources. Int. J. Cooperative Inf. Syst. **02**(02), 127–158 (1993). https://doi.org/10.1142/S0218215793000071
117. Wang, J., Mo, F., Ma, W., Sun, P., Zhang, M., Nie, J.-Y.: A User-Centric Benchmark for Evaluating Large Language Models (2024)
118. Wang, J., Ma, W., Sun, P., Zhang, M., Nie, J.-Y.: Understanding User Experience in Large Language Model Interactions (2024)
119. Beitzel, S.M., Jensen, E.C., Chowdhury, A., Grossman, D., Frieder, O., Goharian, N.: Fusion of effective retrieval strategies in the same information retrieval system. J. Am. Soc. Inf. Sci. Technol. **55**(10), 859–868 (2004)
120. Wu, S., McClean, S.: Performance prediction of data fusion for information retrieval. Inf. Process. Manag. **42**(4), 899–915 (2006)
121. Cormack, G.V., Clarke, C.L., Buettcher, S.: Reciprocal rank fusion outperforms condorcet and individual rank learning methods. In: Proceedings of the 32nd International ACM SIGIR Conference on Research and Development in Information Retrieval, pp. 758–759 (2009)
122. Lee, C.-J., Ai, Q., Croft, W.B., Sheldon, D.: An optimization framework for merging multiple result lists. In: Proceedings of the 24th ACM International on Conference on Information and Knowledge Management, pp. 303–312 (2015)
123. Liu, T.-Y., et al.: Learning to rank for information retrieval. Found. Trends Inf. Retr. **3**(3), 225–331 (2009)
124. Zhan, J., Mao, J., Liu, Y., Zhang, M., Ma, S.: Learning to retrieve: How to train a dense retrieval model effectively and efficiently. arXiv preprint arXiv:2010.10469 (2020)
125. Arora, D., Kini, A., Chowdhury, S.R., Natarajan, N., Sinha, G., Sharma, A.: Gar-meets-rag paradigm for zero-shot information retrieval. arXiv preprint arXiv:2310.20158 (2023)
126. Zhang, T., Patil, S.G., Jain, N., Shen, S., Zaharia, M., Stoica, I., Gonzalez, J.E.: RAFT: Adapting Language Model to Domain Specific RAG (2024)
127. Xu, Z., Tran, A., Yang, T., Ai, Q.: Reinforcement learning to rank with coarse-grained labels. arXiv preprint arXiv:2208.07563 (2022)
128. Shi, W., Min, S., Yasunaga, M., Seo, M., James, R., Lewis, M., Zettlemoyer, L., Yih, W.-t.: REPLUG: retrieval-augmented black-box language models. arXiv preprint arXiv:2301.12652 (2023)
129. Ai, Q., Bai, T., Cao, Z., Chang, Y., Chen, J., Chen, Z., Cheng, Z., Dong, S., Dou, Z., Feng, F., et al.: Information retrieval meets large language models: a strategic report from Chinese IR community. AI Open **4**, 80–90 (2023)
130. Bota, H., Zhou, K., Jose, J.M., Lalmas, M.: Composite retrieval of heterogeneous web search. In: Proceedings of the 23rd International Conference on World Wide Web, pp. 119–130 (2014)
131. Amer-Yahia, S., Bonchi, F., Castillo, C., Feuerstein, E., Mendez-Diaz, I., Zabala, P.: Composite retrieval of diverse and complementary bundles. IEEE Trans. Knowl. Data Eng. **26**(11), 2662–2675 (2014)
132. Kolomiyets, O., Moens, M.-F.: A survey on question answering technology from an information retrieval perspective. Inf. Sci. **181**(24), 5412–5434 (2011)
133. Metzler, D., Tay, Y., Bahri, D., Najork, M.: Rethinking search: making domain experts out of dilettantes. SIGIR Forum **55**(1) (2021) https://doi.org/10.1145/3476415.3476428
134. Tay, Y., Tran, V., Dehghani, M., Ni, J., Bahri, D., Mehta, H., Qin, Z., Hui, K., Zhao, Z., Gupta, J., et al.: Transformer memory as a differentiable search index. Adv. Neural Inf. Proces. Syst. **35**, 21831–21843 (2022)
135. Tang, Y., Zhang, R., Guo, J., Chen, J., Zhu, Z., Wang, S., Yin, D., Cheng, X.: Semantic-enhanced differentiable search index inspired by learning strategies. In: Proceedings of the

29th ACM SIGKDD Conference on Knowledge Discovery and Data Mining, pp. 4904–4913 (2023)
136. Sun, W., Yan, L., Chen, Z., Wang, S., Zhu, H., Ren, P., Chen, Z., Yin, D., Rijke, M., Ren, Z.: Learning to tokenize for generative retrieval. Adv. Neural Inf. Proces. Syst. **36**, 1–17 (2024)
137. Zhuang, S., Ren, H., Shou, L., Pei, J., Gong, M., Zuccon, G., Jiang, D.: Bridging the gap between indexing and retrieval for differentiable search index with query generation. arXiv preprint arXiv:2206.10128 (2023)
138. Nguyen, T., Yates, A.: Generative retrieval as dense retrieval. arXiv preprint arXiv:2306.11397 (2023)
139. Zhan, J., Mao, J., Liu, Y., Guo, J., Zhang, M., Ma, S.: Learning discrete representations via constrained clustering for effective and efficient dense retrieval. In: Proceedings of the Fifteenth ACM International Conference on Web Search and Data Mining. WSDM '22, pp. 1328–1336. Association for Computing Machinery, New York (2022). https://doi.org/10.1145/3488560.3498443
140. Zeng, H., Luo, C., Jin, B., Sarwar, S.M., Wei, T., Zamani, H.: Scalable and effective generative information retrieval. In: Proceedings of the ACM on Web Conference 2024. WWW'24, pp. 1441–1452. Association for Computing Machinery, New York (2024). https://doi.org/10.1145/3589334.3645477
141. Zeng, H., Luo, C., Zamani, H.: Planning Ahead in Generative Retrieval: Guiding Autoregressive Generation through Simultaneous Decoding. In: Proceedings of the 47th International ACM SIGIR Conference on Research and Development in Information Retrieval, pp. 469–480 (2024)
142. Wu, S., Wei, W., Zhang, M., Chen, Z., Ma, J., Ren, Z., de Rijke, M., Ren, P.: Generative retrieval as multi-vector dense retrieval. In: Proceedings of the 47th International ACM SIGIR Conference on Research and Development in Information Retrieval, pp. 1828–1838 (2024)
143. Zhan, J., Mao, J., Liu, Y., Guo, J., Zhang, M., Ma, S.: Jointly optimizing query encoder and product quantization to improve retrieval performance. In: Proceedings of the 30th ACM International Conference on Information & Knowledge Management. CIKM '21, pp. 2487–2496. Association for Computing Machinery, New York (2021). https://doi.org/10.1145/3459637.3482358
144. Sachidananda, V., Kessler, J.S., Lai, Y.-A.: Efficient domain adaptation of language models via adaptive tokenization. arXiv preprint arXiv:2109.07460 (2021)
145. Huang, Q., Tao, M., Zhang, C., An, Z., Jiang, C., Chen, Z., Wu, Z., Feng, Y.: Lawyer llama technical report. arXiv preprint arXiv:2305.15062 (2023)
146. Cui, J., Li, Z., Yan, Y., Chen, B., Yuan, L.: ChatLaw: open-source legal large language model with integrated external knowledge bases. arXiv preprint arXiv:2306.16092 (2023)
147. Wu, S., Irsoy, O., Lu, S., Dabravolski, V., Dredze, M., Gehrmann, S., Kambadur, P., Rosenberg, D., Mann, G.: BloombergGPT: a large language model for finance. arXiv preprint arXiv:2303.17564 (2023)
148. Dai, D., Dong, L., Hao, Y., Sui, Z., Chang, B., Wei, F.: Knowledge neurons in pretrained transformers. arXiv preprint arXiv:2104.08696 (2021)
149. Meng, K., Bau, D., Andonian, A., Belinkov, Y.: Locating and editing factual associations in GPT. Adv. Neural Inf. Proces. Syst. **35**, 17359–17372 (2022)
150. Liu, J., Yu, P., Zhang, Y., Li, S., Zhang, Z., Ji, H.: EVEDIT: event-based knowledge editing with deductive editing boundaries. arXiv preprint arXiv:2402.11324 (2024)
151. Hu, E.J., Shen, Y., Wallis, P., Allen-Zhu, Z., Li, Y., Wang, S., Wang, L., Chen, W.: Lora: low-rank adaptation of large language models. arXiv preprint arXiv:2106.09685 (2021)
152. Li, H., Ai, Q., Chen, J., Dong, Q., Wu, Y., Liu, Y., Chen, C., Tian, Q.: Sailer: structure-aware pre-trained language model for legal case retrieval. In: Proceedings of the 46th International ACM SIGIR Conference on Research and Development in Information Retrieval, pp. 1035–1044 (2023)
153. Yao, F., Li, C., Nekipelov, D., Wang, H., Xu, H.: Human vs. Generative AI in Content Creation Competition: Symbiosis or Conflict? (2024)