Large Language Models and Future of Information Retrieval: Opportunities and Challenges

ChengXiang Zhai
University of Illinois at Urbana-Champaign
Urbana, Illinois, USA
czhai@illinois.edu

ABSTRACT

Recent years have seen great success of large language models (LLMs) in performing many natural language processing tasks with impressive performance, including tasks that directly serve users such as question answering and text summarization. They open up unprecedented opportunities for transforming information retrieval (IR) research and applications. However, concerns such as halluciation undermine their trustworthiness, limiting their actual utility when deployed in real-world applications, especially high-stake applications where trust is vital. How can we both exploit the strengths of LLMs and mitigate any risk caused by their weaknesses when applying LLMs to IR? What are the best opportunities for us to apply LLMs to IR? What are the major challenges that we will need to address in the future to fully exploit such opportunities? Given the anticipated growth of LLMs, what will future information retrieval systems look like? Will LLMs eventually replace an IR system? In this perspective paper, we examine these questions and provide provisional answers to them. We argue that LLMs will not be able to replace search engines, and future LLMs would need to learn how to use a search engine so that they can interact with a search engine on behalf of users. We conclude with a set of promising future research directions in applying LLMs to IR.

CCS CONCEPTS

• **Information systems** → *Retrieval models and ranking*; Language models;

KEYWORDS

Large Language Models, Information Retrieval Models, Search Engines, Intelligent Agent, Conversational Information Access

ACM Reference Format:
ChengXiang Zhai. 2024. Large Language Models and Future of Information Retrieval: Opportunities and Challenges. In *Proceedings of the 47th International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR '24), July 14–18, 2024, Washington, DC, USA.* ACM, New York, NY, USA, 10 pages. https://doi.org/10.1145/3626772.3657848

SIGIR '24, July 14–18, 2024, Washington, DC, USA

ACM ISBN 979-8-4007-0431-4/24/07.
https://doi.org/10.1145/3626772.3657848

1 INTRODUCTION

Recent years have seen great success of large language models (LLMs) [1, 6, 19, 21, 24] in performing many natural language processing tasks with impressive performance, including tasks that directly serve users such as question answering and text summarization [1]. While statistical language models have been applied to information retrieval (IR) since many decades ago [25], these new LLMs go far beyond traditional language models in their representation learning capacity, which enabled them to both understand natural language semantically and generate fluent meaningful natural language text. Moreover, as the model size increases, the performance continues to be further improved with GPT-4 achieving quite impressive performance on many tasks [1]. Furthermore, a foundation model can be fine-tuned with instructions or very few examples to perform many different tasks [27]. They can also perform in-context learning and learn from human feedback [6].

As a new generation of intelligent techniques, LLMs open up unprecedented new opportunities for transforming information retrieval research and applications. Indeed, there has already been a rapid growth of research on using LLMs for IR as shown in the large number of references (231 references) cited in a recent survey of this topic [26] and many influential papers in this area (e.g., [10, 16, 20]).

The existing work on applying LLMs to IR, however, has mostly focused on applying LLMs to improve the *current* search engines. The survey [26] classified most existing work into four directions, corresponding to using LLMs to improve query rewriters, retrievers, rerankers, and readers, respectively. However, the existing work has paid little attention to the important question about how the LLMs would impact the *future* of IR in the long run. An exception is a recent report [2] co-authored by a large group of authors, where the authors provided a systematic discussion of the future applications of LLMs to IR and suggested a framework including collaboration of an IR system, an LLM, and human users. However, it is a high-level framework, mostly fitting the current search paradigm without a detailed technical vision.

In this paper, we extend the previous work by taking a more forward-looking perspective to examine the long-term impact of LLMs on IR research and applications with consideration of the future growth of LLMs and attempt to develop a detailed technical vision for IR research to best exploit strengths of LLMs. Specifically, we aim to address the following questions: 1) How can we both exploit the strengths of LLMs and mitigate any risk caused by their weaknesses when applying LLMs to IR? (Concerns such as halluciation undermine the trustworthiness of LLMs, limiting their actual utility when deployed in real-world applications, especially high-stake applications where trust is vital.). 2) What are the best opportunities for us to apply LLMs to IR, both for improving the current generation search engines and for developing the next-generation search engines? 3) What are the major challenges that we will need to address in the future to fully exploit such opportunities? 4) Given the anticipated growth of LLMs, what will future information retrieval systems look like? Will LLMs eventually replace an IR system?

We examine all these questions and provide provisional answers to them. We discuss technical ideas about using LLMs to improve the current search engines and develop next-generation search engines. We argue that LLMs will not be able to replace search engines in the future; instead, the future LLMs would need to learn how to use a search engine so that they can interact with a search engine on behalf of users. The future IR systems will likely be intelligent agents powered by LLMs to provide personalized task support in an interactive way. We conclude the paper with a set of promising future research directions in applying LLMs to IR.

2 OVERVIEW OF LLMS FOR IR

In this section, we provide a high-level overview of the opportunities and challenges in applying LLMs to IR. We start with an analysis of the new opportunities brought to us by the LLMs as compared with the traditional language models (LMs).

2.1 LLMs vs. Traditional LMs

To understand the strengths and weaknesses of the LLMs and the unique opportunities they bring to IR research and applications, it is useful to make a comparison between the LLMs and the traditional language models such as n-gram language models or topic models [5, 17, 25].

Traditional LMs generally have interpretable parameters. For example, a unigram LM has parameters corresponding to word probabilities, which would allow us to interpret and understand the topic captured by such a model. For example, a model that gives words such as "sport" and "football" the highest probabilities would suggest a sports-related topic. Similarly, a mixture language model such as PLSA [13] or LDA [5] has parameters that represent word distributions for multiple topics as well as probabilities of topic coverage in each document. The interpretability of traditional LMs enables such models to be used to analyze text data to obtain an interpretable semantic representation of text, which can be potentially shown to a user for editing, improvement, or verification.

In contrast, the parameters of the neural network of an LLM are non-interpretable; at least, it would be impossible to interpret individual parameter values in any meaningful way. Thus while an LLM is able to "understand" text to generate an internal semantic representation of text, the representation itself does not have meaning and thus cannot be shown to a human user in any meaningful way, nor can a human interact with such a representation to edit, improve, or verify it.

However, the benefit of interpretability of traditional LMs is at the cost of their limited expression power. Indeed, due to sparseness of data, we cannot estimate an n-gram LM accurately if n is very large. As a result, we tend to stay with a very low-order of n-gram LM. Indeed, in IR, unigram LMs are often used and higher-order n-gram LMs have not really outperformed unigram LMs that much, presumably due to the inaccurate estimate of the higher-order LMs. In contrast, the LLMs seek a latent vector representation of words (embedding vectors) so that the latent representation can be used to predict values of parameters in a traditional language model, thus leading to a more generalizable language model.

Consider, for example, a bigram model $p(u|v)$, where u and v are two words. With a traditional LM, we would need to at least observe the pairwise combinations of all the words in our vocabulary in the training data in order to estimate such a model; indeed, to accurately estimate all those probabilities, we need to observe many times of occurrences of each combination. According to the Zipf's law, however, this generally would not happen in any naturally generated text data since most words would occur rarely, let alone combinations of two words.

Neural LMs break this limitation by attempting to model the relation between the two words u and v based on their embedding vector representation. That is, we no longer treat a word as a "black box" as we did in the traditional LM. This turns out to be quite powerful since it enables us to infer the probability of $p(u|v)$ even if we do not observe combinations of (u, v) in the data. The intuition behind this is that it could use the general context of word u and that of v to infer how likely u would follow v. For example, if u' is a synonym of u, they would occur in similar contexts and thus their embedding vector representation would be similar. Thus, even if we do not observe (u', v), as long as we observe (u, v),we could still infer $p(u'|v)$ based on $p(u|v)$.

Note that this kind of inference must be done in an ad hoc way with a traditional LM using smoothing, but can be achieved naturally in an LLM via learning multiple layers of hidden representation in a deep neural network, where similar contexts are mapped to similar latent representations, enabling extrapolation when predicting a word based on a context.

The comparison shows that LLMs are in general much more powerful than the traditional LMs from the perspective of language modeling. The multi-layer latent vector representation naturally performs all the needed smoothing in a traditional LM. The latent representation can be regarded as some level of understanding of the semantics of text data by LLMs, which also explains why they can perform so well on many tasks. However, the major limitation of LLMs as compared with traditional LMs is that their parameters are not interpretable, making it hard to directly improve them via human feedback, nor can a user examine those parameters to obtain any understanding of the content to be analyzed.

Another difference between LLMs and traditional LMs is that the LLMs are typically trained with massive amounts of data, including not just natural language text but also other semi-structured data such as software code and tables. Large Multimodality LMs can be even trained with multimodality data including text and images. Moreover, many machine learning techniques can be applied to improve LLMs, including self-supervised (unsupervised) learning [9], supervised learning (fine-tuning) [27], instruction tuning [18], reinforcement learning from human feedback (RLHF) [7], and in-context learning [6].

As a result, a single foundation LLM can be instructed/trained to perform multiple tasks, including particularly conversations with a user using natural language, thus directly enabling question answering and conversational IR.

The general opportunities provided by the LLMs include the following:

Conversational User-System Interaction: Being able to understand natural language enables conversational user-system interaction for any system. Specifically, an LLM can understand a user's natural language instruction, so a user can potentially interact with *any* system using natural language sentences (e.g., chat), which can then be translated into well defined instructions that the system can understand.

Text Generation and Conversation: Being able to generate fluent natural language enables many text-generation tasks that would otherwise be infeasible. As a specific example, an LLM can conduct natural language conversations.

Learning to Perform New Tasks: Fine-tuning (with instructions) further enlarges scope of applications since we can potentially fine-tune with any tasks that we are interested in. Code-training can often further enhance their capacity [23].

It is worth noting that as we further fine-tune and train a foundation LLM, the LLM would behave increasingly as an intelligent agent that can autonomously finish a task (e.g., answering questions of users). As an agent, however, an LLM suffers from the possibility of hallucination, i.e., generating text that is not true. This can be regarded as the inevitable cost paid while a neural LM attempts to generalize based on the learned representation of context, i.e., hallucination is a symptom of "over generalization." Note that over generalization is sometimes beneficial (e.g., when an LLM is asked to write a poem), but in many other cases such as serving users with information and knowledge, hallucination would inevitably undermine their trustworthiness. To make the situation worse, the LLMs cannot really explain the reasoning behind the generation of a certain text, making it even harder to address the problem of hallucination, significantly limiting their utility, especially for high-stake applications such as medical information when reliability of the generated information is crucial.

From IR perspective, another major limitation of LLMs is their lack of access to the most current information. For example, it would not know today's weather or any news article published five minutes ago. While theoretically speaking, an LLM can be updated frequently to address this problem, in practice, it is generally infeasible to do so, thus they alone are not sufficient to serve users and a search engine or database system is still needed to support access to real-time information.

2.2 Improvement of Current Search Engines

Despite their lack of understanding of natural language, the current search engines are already quite useful and are used daily by many users. The success of the search engine industry has much to do with the optimization of AI-human collaboration; indeed, as an assistive AI system, a search engine provides clear value to a user by helping a user find relevant documents (information) from a large collection of documents such as the Web. A search engine can augment a user's intelligence in the sense that a user with access to a search engine would be able to make better decisions than one without access to a search engine (the former would appear to be more intelligent).

Most search engine users have adapted to the intelligence level of a search engine in order to collaborate with a search engine effectively. Thus although a search engine can only respond to a keyword query without being able to answer their questions as ChatGPT can do, the users are used to entering keyword queries and reformulating their queries. Of course, if a system could answer a user's question directly (and reliably), the user would prefer such a question-answering system to a search engine. Indeed, when a user used a search engine initially, they might have the tendency to type in a question instead of a keyword query, but after realizing that a search engine does not really understand a question and treats a question as a set of keywords, the user would adapt to the search engine by using just keyword queries.

When a user's query is a popular query and the search engine has already received some feedback about the query from previous users, the search engine would generally work well and the users are generally satisfied with such a query-based interaction. Where the users are not satisfied is when the search engine could not answer their queries satisfactorily even after the user reformulates a query multiple times. Such "difficult" queries are often long-tail queries, meaning that not many users have this kind of information need. There are many such somewhat unique queries and they generally follow a distribution with a long tail.

A common reason why this happens is when the user has a vocabulary gap, i.e., the user does not know well about the relevant content and thus cannot predict what kind of words would be used in a relevant document. Unfortunately this scenario is quite common as when a user has a need for some information, it is generally because the user is not so familiar with the topic (or otherwise, they would not need to search in the first place). Specific examples include medical search or E-Com search with a need for a product with complex specification.

Unclear/vague information needs (e.g., buying a gift) may also lead to challenges for a search engine to effectively serve its users. In such a scenario, a user would benefit from receiving diversified search results and support in learning about each option so as to refine a vague information need into a more specific one.

The current search engines are also limited in the following aspects: They cannot understand a user's intent accurately. They cannot help users digest search results. They only support limited interactions (users have limited ways to express their information need).

All the limitations discussed above can be potentially addressed by leveraging LLMs: 1) LLMs can understand natural language well and thus can easily bridge vocabulary gap by suggesting alternative words to use in a query or facilitating matching of different words that are semantically related (i.e., support semantic matching). 2) LLMs can be used to clarify a user's intent or help a user clarify/refine the intent (e.g., using a dialogue with a user). 3) LLMs can be prompted to summarize search results from any perspective that is suitable for a user. 4) LLMs can be leveraged to support conversational search, thus enabling a user to express an information need more accurately and in a more informative way.

2.3 LLMs for Next-Generation Search Engines

With the conversation capacity of an LLM and their capacity of answering questions directly, the next-generation search engine can potentially provide much more effective information service to a user by supporting question answering and search together in a conversational manner.

Question answering is often the most efficient way to satisfy a user's information need, and the LLMs can answer many questions correctly and concisely based on the information in the training data. However, due to the concern of hallucination, a user would generally expect to see the original source of information that supports an answer. Unless an LLM can remember all the sources, we would likely need a search engine to find the sources that can support an answer. This means that the LLM needs to learn how to use a search engine, or otherwise, the user would have to do search in order to verify an answer. Thus even if we make an LLM the major agent to interact with a user via question answering, it would still need to collaborate with a search engine.

In other cases, a user may want to primarily use a search engine. For example, when a user's information need is exploratory, the user's need may be better satisfied by using a search engine. In such a case, an LLM can enhance a search engine in many ways, including conducting a dialogue with the user to clarify the information need.

Thus in general, the next-generation search engines would likely integrate question answering with search and support conversational search. The traditional retrieval models are virtually all designed based on the probability ranking principle, and thus all support ranking of documents. As we move toward the era of conversational search, we would need a new kind of retrieval models that can enable conversational clarification of a user's information need. Such a retrieval model needs to be interpretable so that a user can be involved in the loop of clarifying information need and optimizing ranking of documents. We will propose an interpretable and explainable probabilistic matching model for this purpose later in this paper.

3 IMPROVING CURRENT SEARCH ENGINES

There has already been much work on using LLMs to improve current search engines. The survey [26] provides an excellent review of the existing work. While the results from existing work are generally positive, they do not necessarily generate the biggest impacts that are possible with LLMs. Below we provide a slightly different framework to discuss some of the best opportunities for applying LLMs to improve the current search engines, where the LLMs can be expected to make significant impact on improving the utility of a search engine.

Our perspective is based on the *ideal query hypothesis* [14], which states that for any given information need, there exists a perfect query [15] that would allow a retrieval system to rank all the relevant documents above the non-relevant ones. While such a perfect query might not actually exist, it is generally possible to uniquely identify a relevant document by just using a highly effective query that contains a few terms that occur together in the target (relevant) document but not in other documents. Such a query would enable a user to see at least one relevant document on the top of search results. From here, a user could provide feedback or additional queries to retrieve additional relevant documents as needed.

An **important implication** of the ideal query hypothesis is that a keyword-based search engine (i.e., a current search engine with a basic retrieval function such as BM25) can be quite effective if the system can help a user construct an ideal query, and users would likely be satisfied with such a search engine; the search system would also be highly efficient since it can leverage inverted index to enable search to be done efficiently. Thus, one of the best opportunities for LLMs to improve the current search engines is for them to be used to help a user construct an ideal query, especially since LLMs can be assumed to have little vocabulary gap and thus able to formulate an effective query.

With this perspective, we now briefly discuss how LLMs can address multiple limitations of the current search engines.

3.1 LLMs for Bridging Vocabulary Gap

In general, the vocabulary gap can be bridged in three ways: 1) expand a query to include terms that can match those in a document relevant to the query; 2) expand a document to include terms that might be used in queries to which the document is relevant; 3) keep both the query and document unchanged, but allow matching of different words that are semantically associated. The three strategies are complementary and can thus be combined potentially.

LLMs can be directly used (via either prompting or fine-tuning) to perform query expansion and document expansion. This would exploit the capacity of text generation of LLMs directly. Semantic matching can be achieved by fine-tuning LLMs to assess relevance of document to a query. The general idea would be similar to many neural ranking models [12], where both the query and document would be mapped to embedding vectors, whose similarity can then be computed to assess relevance.

3.2 LLMS for Clarifying User Intent

When a user's intent is not clear (e.g., an ambiguous query word), the returned top results likely contain results matching different intents (e.g., mixed senses of the ambiguous word). In such a case, LLMs can be used to generate clarification questions about a user's query based on the top-ranked results. For example, LLMs can be asked to analyze the top results to identify multiple senses of a word and generate a question such as "Did you mean jaguar as an animal or a car?". LLMs can further be used to interpret a user's answer and improve query formulation.

When a user's intent is vague due to lack of knowledge, LLMs can teach the user relevant knowledge for refining a query during the search time. For example, the user may be provided with a link described as "learn more about X", clicking on which would trigger a conversation of the user with an LLM. In general, LLMs can be quite useful to support user learning during a search session [8].

3.3 LLMs for Search Result Summarization

Instead of having users interact with the current unstructured collection of search results, LLMs can be used to directly summarize search results with consideration of the user's query (i.e., query-focused summarization). LLMs can also support question answering in the context of search results to enable users to digest search results flexibly and gain relevant information. If the search results are summarized, the summary can serve naturally as a starting point for a conversation with a user, allowing a user to ask additional questions and be engaged with a conversation.

In general, LLMs can help a user learn during search, thus optimizing human-AI collaboration by increasing human intelligence. Once a user gains sufficient knowledge, the user would be able to formulate an effective query, which would allow the current search engine (even without improvement) to serve the user with satisfactory results.

3.4 LLMs for Conversational Feedback

From the perspective of AI-human collaboration, it makes sense to leverage human intelligence as much as we can [4], which for IR means to enable a user to naturally provide feedback and explain why the user is not satisfied with the current search results. For example, a user can be allowed to explain why they like or dislike a particular document, and use an example document to elaborate information need. LLMs can be leveraged to enable users to provide conversational feedback. For example, a user may be allowed to say "I like this document because it is about X," where 'X' is a topic description, and LLMs can be used to reformulate a query based on this feedback accordingly. In general, with LLMs, a user would be able to express the information need using complex natural language sentences as well as provide feedback in a particular search context where specific search results can be referred to provide detailed feedback.

3.5 Summary

In sum, LLMs can be exploited naturally to enhance a current search engine significantly by enabling a user to express information need using potentially complex natural language sentences and collaborating with them to interactively formulate an effective (ideal) query, which can then be executed using a regular search engine to generate satisfactory results to a user without increasing the complexity of search. They can also be used to facilitate users to efficiently digest relevant information via adaptive summarization of search results and enhance a user's knowledge by answering their questions. Finally, they can be used to understand a user's feedback in context, thus improving query reformulation. All these improvements can be mapped to the general human-IR-LLM interaction framework proposed in [2] as specific ways to realize the interactions.

However, while these applications of LLMs would lead to more interactions with users, including even natural language conversations, they cannot effectively address the key challenge in search, i.e., modeling relevance in detail and accurately; indeed, without an interpretable and explainable model for modeling relevance in detail, no conversation between a system and a user would likely lead quickly to accurate matching of a user's information need with documents in the collection. In order for this to happen, the conversation must be conducted around a more *detailed model* of relevance. This can be achieved by developing a general interpretable and explainable matching model for IR as we will discuss in the next section. Since a system built based on such a new model is likely dramatically different from the current search engines, we would view this as the next generation search engines, which will be discussed in the next section in detail.

4 NEXT-GENERATION SEARCH ENGINES

The most important and difficult challenge in IR applications is how to serve a user when the user could not formulate effective queries. In such a case, we would need to understand in detail exactly what the user is looking for. The key question is: what are exactly the criteria of relevance for this user? How to use conversations or interactions with a user to naturally clarify the relevance criteria is a key technical challenge since this requires formalization of relevance in some way. We propose an interpretable probabilistic matching model for formalization of relevance based on modeling multiple attributes of both information needs and information items (documents). The decomposition of relevance matching into attribute matching enables more detailed attribute-level feedback about the search results, which can be used to directly improve relevance matching and support conversational clarification of a user's intent.

4.1 Explainable Probabilistic Retrieval Model

The proposed model is illustrated in Figure 1. Formally, we frame the problem as one to determine whether there is a match between an information item I from source S and a query Q from user U in context C, where context C could potentially include many environmental factors that may affect search such as the time or location of the search. That is, we would like to estimate the probability $p(Match|Q,U,C,I,S)$. We intentionally use the term "Match" instead of "Relevance" to emphasize the multiple dimensions of the matching criteria and avoid the typical (narrow) interpretation of topical relevance as the main retrieval criterion. Functionally, however, we could rank items based on probability of matching in the same way as probability of relevance. We also show that similar users and similar items can both be potentially leveraged to improve estimate the matching probability, though we would not explore it in this paper.

If we estimate such a probability directly without further decomposition, we can rank items but would not be able to support any meaningful conversation with a user in order to clarify the intent or obtain detailed feedback. Our idea is thus to decompose the matching of an item with a query into matching of multiple attributes of each.

First, we assume that an item can be characterized by multiple attributes $B_1, ..., B_m$ and the attribute values of an item I from source S are characterized by a set of probabilities $\{p(B_i|I,S)\}$, where $i = 1, ..., m$. For example, if the item is a product, B_i could be the price or weight, and $p(B_i|I,S)$ could represent the actual price or weight of the particular item I (we use probability because not all attribute values are known, thus there is uncertainty). Similarly, if the item is a news article, B_i could be a topic category or date of the article, and $p(B_i|I,S)$ may represent the actual topic or date of the particular article I.

Analogously, we assume that a user U's information need N is also characterized by multiple attributes $A_1, A_2, ..., A_n$, and the preferred attribute values are modeled by a set of probabilities $\{p(A_i|N)\}$. In general, attributes A_i and B_i are different (thus causing vocabulary gap), though they may also overlap. For example, if the query is about finding a laptop, $p(A_i|N)$ could be the preferred price or weight (which likely overlap with B_i), but A_i could also be "whether a laptop is suitable for travel" (which is unlikely directly matching any B_i). Note that the information need N is not observed, and given N, the preferred attribute values are assumed to not depend on the Q, U, or C, thus N fully characterizes the user's information need as expressed in the query Q in context C. Under this assumption, we would also need to infer the information need based on the query, i.e., estimate $p(N|Q,U,C)$.

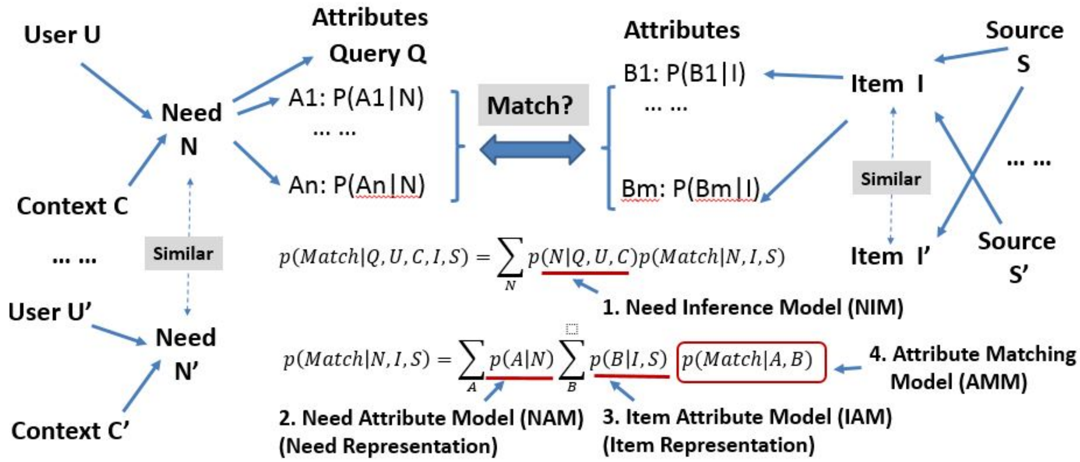

Figure 1: An Interpretable Probabilistic Information Retrieval Model

With such an attribute-based model for both information need and item representation, we can now define matching at the level of attributes as follows

$p(Match|Q,U,C,I,S) = \sum_N p(N|Q,U,C)p(Match|N,I,S)$

$p(Match|N,I,S) = \sum_A \sum_B p(A|N)p(B|I,S)p(Match|A,B)$

The first equation captures the intuition that matching with a query means matching with the information need expressed by the query and considers the uncertainty about the information need N. The second equation further defines the matching of an item with an information need as matching of their respective attributes. With this decomposition, we see that the whole retrieval model is based on the following specific models:

1) Need Inference Model (NIM): $p(N|Q,U,S)$ captures how likely the information need is N given that user U presents query Q in context C.

2) Need Attribute Model (NAM): $p(A|N)$ represents the information need in detail based on the preferred attribute values.

3) Item Attribute Model (IAM): $P(B|I,S)$ represents the item in detail based on the (inferred) attribute values of item I.

4) Attribute Matching Model (AMM): $p(Match|A,B)$ models how likely attribute B (of an item) would match with (i.e.,satisfy) attribute A of an information need.

A major advantage of this model is that it can provide a meaningful explanation of why an item is retrieved for a user. Assuming that all the attributes A_i and B_i are interpretable, we would be able to explain why an item is retrieved by identifying the matching attributes that contributed most to the matching probability (e.g., the attribute B_i of this item (e.g., laptop weight) matches with the attribute A_i of your information need (e.g., "laptop suitable for travel")).

Moreover, with such an interpretable explanation, a user could clarify any misinterpretation of the user's intent by providing feedback explicitly on $p(N|Q,U,C)$ or $P(A|N)$, enabling meaningful conversations that can lead to improvement of the component models, thus the overall search results.

Theoretically speaking, if we can estimate all these component models accurately, we would be able to match items with a user's information need perfectly. In reality, however, it remains quite challenging to estimate all these models accurately. However, LLMs could be leveraged to estimate all of them.

4.2 LLMs for Model Estimation

4.2.1 Need Inference Model (NIM): p(N|Q, U, C). The estimate of NIM first requires us to define an ontology of user information need. This can be, e.g., product catalog, or any other meaningful taxonomy. LLMs can be prompted to create such an information ontology in advance. For example, an LLM may suggest different types of products, which may be more refined than the product types maintained in an E-Com search engine.

Next, given the query, an LLM can be prompted to directly suggest the possible intent/need of the user.

4.2.2 Need Attribute Model (NAM): p(A|N). The attributes can often be obtained from the inferred need N behind a query. For example, if a query is about a product, the inferred product type (e.g., laptop) may directly suggest attributes such as weight, screen size, and price, either based on the attribute information from a product collection or by using LLMs.

LLMs can be easily prompted to suggest preferred values of an attribute. We show an example of prompting ChatGPT to estimate the attribute "screen size" for the query "laptop for travel" in Figure 2. The results from ChatGPT included multiple sizes with an explanation of each: 13.3", 14", 11-12", 15" or larger.

There are a few interesting observations here: First, the original query has a vocabulary gap since laptop descriptions (specifications) unlikely match a word such as "travel", but ChatGPT could bridge this vocabulary gap by inferring the preferred attribute values, which in this case overlap with item attributes and thus can be directly matched with item attributes. Second, the uncertainty about the preferred attribute values provide a basis for conversational clarification of a user's intent. It also suggests that the results should be diversified and use a user's feedback to further clarify the intent.

4.2.3 Item Attribute Model (IAM): p(B|I, S). IAM can generally be inferred based on the text descriptions of item I. For example, User (Prompt): A user is looking for "laptop for travel". What are the preferred screen sizes?

ChatGPT: "The preferred screen sizes for a "laptop for travel" can vary from one traveler to another based on individual preferences and needs. However, many travelers often lean towards laptops with specific screen sizes that balance portability and usability. Here are some of the preferred screen sizes for travel laptops:

1.13.3 Inches: Laptops with 13.3-inch screens are quite popular among travelers. They offer a good compromise between portability and usability. These laptops are typically compact and lightweight, making them easy to carry, and the screen size is sufficient for productivity tasks and entertainment.
2.14 Inches: Laptops with 14-inch screens are also a common choice for travelers. They provide a slightly larger display area than 13.3-inch laptops, which can be beneficial for those who need a bit more screen real estate while still maintaining a reasonable level of portability.
3.11 to 12 Inches: Some travelers prefer even smaller laptops with 11 to 12-inch screens. These ultra-portable laptops are extremely lightweight and compact, making them ideal for users who prioritize mobility above all else. However, the smaller screen size might be less suitable for extended work or multimedia tasks.
4.15 Inches and Larger: While less common, there are travelers who opt for larger laptops with 15-inch screens or larger. These laptops provide a more extensive viewing area but may sacrifice some portability. They are often chosen by travelers who prioritize a larger display for entertainment or work purposes and are willing to carry a slightly heavier laptop.

Ultimately, the preferred screen size for a travel laptop depends on the individual's specific needs and priorities. It's essential to consider how you plan to use the laptop while traveling, whether it's for work, entertainment, or a combination of both, and choose a screen size that aligns with those needs while still ensuring the laptop is easy to carry and fits within your travel setup.

Figure 2: An example of estimating NAM using ChatGPT

product specification can be used to extract specific attribute values. In the case of a news article, information extraction or text analysis can be used to extract interesting attribute values such as the major topic or date. In all these cases, LLMs can be exploited to enhance an existing method or generate attribute values directly for an item. Thus LLMs can also be used to fill in missing values or augment attribute set with additional attributes that may better reflect a user's perspective (e.g., "suitable for travel" may be added to a laptop with low weight).

4.2.4 Attribute Matching Model (AMM): p(Match|A,B). Once attributes A and B are clearly defined, LLMs can often be prompted to provide an estimate of whether A matches B as well as an explanation. As an example, consider a query/need about a laptop with two attributes values: (weight="light"; use ="travel") and some specific laptop products. We prompted ChatGPT to estimate the match of the attributes. The results show that ChatGPT indeed can provide a quite meaningful estimate with explanation.

Specifically, in Figure 3, we see that ChatGPT is quite confident to say there is a match, i.e., $P(Match = 1|A = \text{“}travel\text{”}, B = \text{“}weight : 3lb\text{”})$, whereas in Figure 4, it would be less certain when estimating $P(Match = 1|A = \text{“}travel\text{”}, B = \text{“}weight : 4lb\text{”})$. What is interesting is that it has given specific explanation regarding why it is uncertain, which provides useful context to generate clarification questions to ask the user about their preferences, thus supporting conversational search/feedback. In Figure 5, we see that ChatGPT could use its knowledge to accurately estimate $P(Match|A = \text{“}travel\text{”}, B = \text{“}battery : 1hour\text{”})$, where there is a larger vocabulary gap, thus requiring additional knowledge for estimate the matching here. Once again, ChatGPT can be leveraged to engage with users to further clarify their needs. Such conversational clarification of intent is enabled by the proposed decomposition of matching into matching at the attribute level, which is necessary in order to effectively help a user whose query does not work well.

4.3 Explainable NN Matching Model

Interestingly, the proposed probabilistic model can also be implemented efficiently as a completely interpretable neural network model as shown in Figure 6. Such an interpretable matching neural network (MNN) would take user U, context C, and query Q as input at the bottom and attempt to predict (generate) an item I as output on the top. Different from an ordinary neural network, both the

User: If a user's query contains "travel", would a product of type "laptop" and "weight=3lb" match the need of the user?

ChatGPT: If a user's query contains the term "travel" and they are specifically looking for a "laptop" with a weight of 3 pounds, it is highly likely that such a laptop would match the user's needs, especially if they prioritize portability for travel.

A laptop with a weight of 3 pounds is considered lightweight and highly portable, often falling into the category of ultraportable laptops. These laptops are specifically designed for users who travel frequently and need a compact and easy-to-carry computing device.

For users seeking a laptop suitable for travel, a 3-pound laptop is likely to meet their requirements for portability, making it a good match for their needs. Such laptops are convenient for carrying in backpacks, briefcases, or travel bags, and they are well-suited for use on the go, whether for business, leisure, or academic purposes.

Figure 3: Estimating AMM $P(Match|A = \text{“}travel\text{”}, B = \text{“}weight : 3lb\text{”})$ using ChatGPT

User: If a user's query contains "travel", would a product of type "laptop" and "weight=4lb" match the need of the user?

ChatGPT: If a user's query contains the term "travel" and they are specifically looking for a "laptop" with a weight of 4 pounds, it is possible that such a laptop could match the user's need, depending on their individual preferences and requirements.

The user's inclusion of the term "travel" in the query suggests that they are looking for a laptop suitable for travel purposes. Laptops with a weight of 4 pounds are generally more portable than heavier laptops but may not be as lightweight as some ultraportable models that weigh around 2-3 pounds.

Whether a 4-pound laptop is a good match for the user's travel needs depends on their personal comfort level with that weight. Some travelers may find a 4-pound laptop to be acceptable for their travel requirements, especially if other features of the laptop align with their needs, such as performance, battery life, and durability.

Ultimately, it's essential to consider the user's individual preferences and priorities when determining whether a laptop with a weight of 4 pounds is a suitable match for their travel needs. Users who prioritize portability above all else might prefer even lighter laptops, but for some travelers, a 4-pound laptop may be sufficiently portable for their purposes.

Figure 4: Estimating AMM $P(Match|A = \text{“}travel\text{”}, B = \text{“}weight : 4lb\text{”})$ using ChatGPT

User: If a user's query contains "travel", would a product of type "laptop" and "battery hour=1 hour" match the need of the user?

ChatGPT: If a user's query contains the term "travel" and they are specifically looking for a "laptop" with a battery life of only 1 hour, it's unlikely that such a laptop would match the needs of most travelers. A laptop with only 1 hour of battery life would have a very limited and short-lived power source, which may not be practical for most travel scenarios.

Most travelers typically require laptops with longer battery life to ensure they can work, watch movies, or browse the internet during flights, train rides, or when they don't have access to power outlets. Travelers often prioritize laptops with extended battery life to minimize the need for frequent recharging, especially during long journeys or while on the go.

While a laptop with a 1-hour battery life may exist, it's not a practical choice for most travelers. Laptops with longer battery life, typically ranging from 6 hours to 12 hours or more, are usually preferred by users looking for a travel-friendly computing experience.

Figure 5: Estimating AMM $P(Match|A = \text{“}travel\text{”}, B = \text{“}battery : 1hour\text{”})$ using ChatGPT

nodes and edges in an MNN are interpretable as clearly explained in Figure 6. If we pass information from bottom to top using such an MNN, we would be able to obtain a score for each I_i w.r.t. U, Q, and C equivalent to our probabilistic model, i.e., the score would be equivalent to $p(Match|Q, U, C, I_i, S)$ (we omitted the source S in the MNN illustration). This means that we can train the proposed probabilistic model in an end-to-end manner similarly to training a neural network by using search log data as training data.

However, one concern of this kind of training is that the data may be sparse, thus we would suffer the same problem as the traditional LM. To address this issue, there are two strategies. One is to leverage LLMs to estimate or improve the estimate of all the component models as we have already discussed, thus not necessarily or not solely relying on search log data. The other is to use the MNN as a "backbone" of a neural network and expand it by replacing each node with potentially a set of neurons and each edge with a a sub-neural network (i.e., an edge connecting X to Y can be

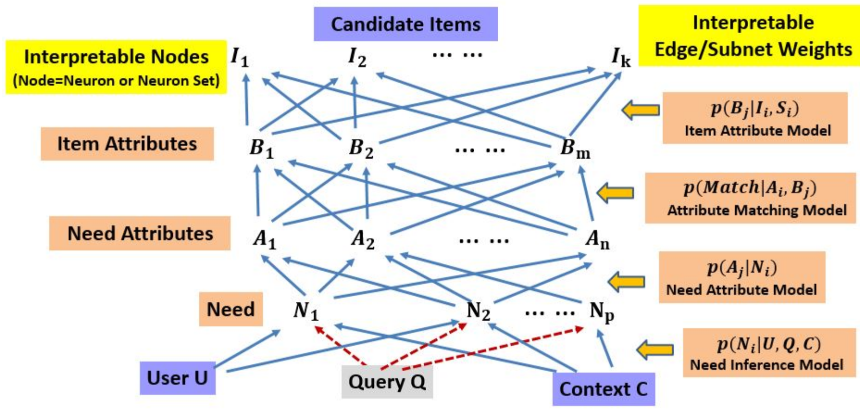

Figure 6: Probabilistic Model as an Interpretable Matching Neural Network (MNN)

replaced by a sub-neural network with X as input and Y as output). This way, we would have an interesting neural network with better interpretability than an ordinary neural network while also allowing us to take advantage of the capacity of representation learning with a deep neural network (it may be related to neurosymbolic architecture [11]).

5 LLMS AS USERS OF SEARCH ENGINES: TOWARDS INTELLIGENT AGENTS

It is reasonable to expect the LLMs to become even stronger as more data and human users/tasks are used for training them and newer human brain-like architectures may be developed to improve their reasoning capacity. How would this impact IR research and applications in the long run? What would future IR systems look like? Will LLMs eventually replace IR systems (completely)? In this section, we attempt to address these questions with a vision for future IR systems, which we believe would be personalized task agents that would leverage search engines as well as LLMs as supporting technologies to help a user finish a task [22].

In Figure 7, we illustrate this vision. Along the X-axis, we show that the current generation LLMs are mostly non-explainable, but in the future, we will likely see more explainable LLMs with human-like neurosymoblic architectures, and finally, we may have more autonomous LLMs that do not rely on humans to design their reward (objective) functions as they can potentially learn them from interacting with users and the task environment. Based on this vision, it would be reasonable to expect that the IR systems would also evolve accordingly, which we show on Y-axis.

Specifically, since the current LLMs are non-explainable, their trustworthiness would limit their utility in directly serving users (e.g., hallucination problem), thus in general, a good strategy for using the current LLMs is to use them as component or support technologies for an existing system such as a search engine (thus LLMs are "below" existing technologies). As the LLMs become explainable and more trustworthy, we can anticipate LLMs to serve users more directly. At that point, the existing IR technologies would likely become support technologies for LLMs (thus LLMs are "above" existing technologies). However, the integration of them would likely be loose and humans would have to design a framework for them to interact since the LLMs are not autonomous. Finally, as we have more autonomous LLMs, they will likely be able to learn how to use existing tools such as a search engine or any other tools automatically, thus we will likely have autonomous intelligent agents powered by LLMs that can "absorb" all the technologies including search engines, database systems, and other tools in the task environment.

On the surface, this analysis might appear to suggest that IR systems (search engines) would be eventually replaced by LLMs. This, however, is not true. What we propose is that IR systems would evolve into more autonomous and more intelligent task agents that would be able to use search engines on behalf of users.

Indeed, we argue that LLMs would unlikely be able to replace search engines. This is because they generally cannot have easy access to the newest information available unless they can use a search engine. For example, ChatGPT would not be able to answer a question about any event that has happened today, simply because it has not been trained/provided with such information. While theoretically, it might be possible for ChatGPT to be updated every minute, in practice, it is infeasible. Given this limitation, the only way to allow ChatGPT to serve users with the most recent information is for it to use a search engine. Thus search engines will always be needed. However, in the future, it is possible that users would interact with LLMs (more accurately, agents powered by LLMs), which then would use a search engine as needed on behalf of the user.

Since the future LLMs would be used in such an environment that search engines are assumed to be available, they would need to be trained with a different objective than what has been used today so that they would be discouraged from "wasting" parameters to memorize the data or information that can be easily retrieved by using a search engine and encouraged to use their parameters to learn more sophisticated knowledge/information that cannot be easily provided by a search engine.

The scenario is similar to a closed-book exam vs. open-book exam. When we use a current LLM to solve a problem, it is similar to a closed-book exam since the model must solve the problem without consulting any external source. In order to serve users with the most up-to-date information, however, those LLMs must

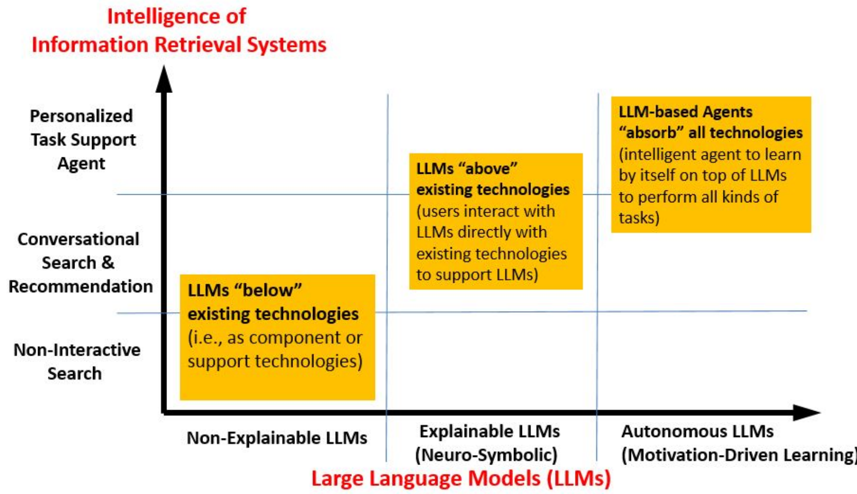

Figure 7: Future of LLMs and IR

adopt an open-book model, i.e., invoking a search engine as needed in real-time when helping a user finish a task. In general, future intelligent task agents would need to learn how to use various tools effectively, particularly a search engine. Just as students would not memorize data when preparing for an open-book exam (since they can easily look at the data as needed), future LLMs would also need to be trained more intelligently to not waste parameters to memorize the data and information that they can easily obtain by using a search engine (at running time). This could be achieved by minimizing the errors given by a hybrid system where an LLM would be combined with other tools such as a search engine to collaboratively minimize the loss function; for example, the agent can use reinforcement learning to learn a policy about when to use a search engine or another tool.

From IR research perspective, we need to address special challenges associated with building search engines to serve AI agent users instead of human users (e.g., the sequential browsing assumption made about human users is unlikely valid for an AI user). Also, AI agents, powered by LLMs, generally have much less vocabulary gap than human users, thus they can be expected to formulate effective keyword queries, which means that it might be sufficient for a search engine to just support keyword search and that complex ranking algorithms including neural ranking algorithms [12] might become less critical or even unnecessary considering their computational complexity.

6 SUMMARY AND FUTURE RESEARCH

In this paper, we addressed various questions about how the LLMs can be applied to IR and how they will impact IR research and applications in the long run. We suggested multiple ways for LLMs to improve the current generation search engines and proposed an interpretable probabilistic retrieval model to enable conversational search in the next generation search engines, which can be implemented as an interpretable neural network. Finally, we argue that search engines will not be replaced by LLMs, but future LLMs would need to learn how to use search engines in real time when serving users.

We conclude the paper with a list of interesting questions for future research: **1)** How can we leverage LLMs for bridging vocabulary gap? How can we use LLMs to effectively and efficiently support query reformulation and document expansion/annotation? **2)** How can we leverage LLMs for summarizing search results in an interactive way (e.g., generating navigable summary)? **3)** How can we leverage LLMs for interactive clarification of query intent and user task? **4)** How can we fine-tune LLMs with search log data to effectively learn representation of relevance (reasoning with relevance)? How can we systematically leverage LLMs to estimate the component models in the proposed explainable probabilistic model? **5)** How can we leverage LLMs to support question answering with knowledge provenance? **6)** How can we support question answering with up-to-date information and knowledge? **7)** How can we support question answering without hallucination or with support for interactive verification of answers? **8)** How can we use LLMs to build realistic user simulators for both evaluating and training interactive task agents [3]? **9)** How can we use LLMs to model user tasks? **10)** How can we use LLMs to build personalized conversational task agents that can continuously learn from task environment (e.g.,increase its skills of using a search engine) over time? Future research progress in these directions would enable effective use of LLMs for both improving the current search engines and developing next generation search engines.

7 ACKNOWLEDGMENTS

This work is supported in part by the US National Science Foundation and the Institute of Education Sciences through award number 2229612 (National AI Institute for Inclusive Intelligent Technologies for Education), and by the IIDAI and Jump ARCHES Program at UIUC. The author has benefited from discussions of the topic with many other researchers and wants to thank all of them, especially Alessandro Magnani, Hongning Wang, Jiafeng Guo, Min Zhang, Krisztian Balog, Ryen White, Heng Ji, Jiawei Han, Edward Hovy, Hang Li, Jie Tang, Daniel Campos, Kevin Ros, Ke Yang, Adam Davies, Liliang Ren, and Bhavya.

REFERENCES

[1] Josh Achiam, Steven Adler, Sandhini Agarwal, Lama Ahmad, Ilge Akkaya, Florencia Leoni Aleman, Diogo Almeida, Janko Altenschmidt, Sam Altman, Shyamal Anadkat, et al. 2023. Gpt-4 technical report. *arXiv preprint arXiv:2303.08774* (2023).

[2] Qingyao Ai, Ting Bai, Zhao Cao, Yi Chang, ..., Weinan Zhang, Min Zhang, and Xiaofei Zhu. 2023. Information Retrieval meets Large Language Models: A strategic report from Chinese IR community. *AI Open* 4 (2023), 80–90. https://doi.org/10.1016/j.aiopen.2023.08.001

[3] Krisztian Balog and ChengXiang Zhai. 2024. User Simulation for Evaluating Information Access Systems. *Foundations and Trends in Information Retrieval* (2024). forthcoming.

[4] Nicholas J Belkin. 1996. Intelligent information retrieval: Whose intelligence? *ISI* 96 (1996), 25–31.

[5] David M Blei. 2012. Probabilistic topic models. *Commun. ACM* 55, 4 (2012), 77–84.

[6] Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, et al. 2020. Language models are few-shot learners. *Advances in neural information processing systems* 33 (2020), 1877–1901.

[7] Paul F Christiano, Jan Leike, Tom Brown, Miljan Martic, Shane Legg, and Dario Amodei. 2017. Deep reinforcement learning from human preferences. *Advances in neural information processing systems* 30 (2017).

[8] Kevyn Collins-Thompson, Preben Hansen, and Claudia Hauff. 2017. Search as learning (dagstuhl seminar 17092). In *Dagstuhl reports*, Vol. 7. Schloss Dagstuhl-Leibniz-Zentrum fuer Informatik.

[9] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, NAACL-HLT 2019, Minneapolis, MN, USA, June 2-7, 2019, Volume 1 (Long and Short Papers)*, Jill Burstein, Christy Doran, and Thamar Solorio (Eds.). Association for Computational Linguistics, 4171–4186. https://doi.org/10.18653/V1/N19-1423

[10] Guglielmo Faggioli, Laura Dietz, Charles LA Clarke, Gianluca Demartini, Matthias Hagen, Claudia Hauff, Noriko Kando, Evangelos Kanoulas, Martin Potthast, Benno Stein, et al. 2023. Perspectives on large language models for relevance judgment. In *Proceedings of the 2023 ACM SIGIR International Conference on Theory of Information Retrieval*. 39–50.

[11] Artur d'Avila Garcez, Sebastian Bader, Howard Bowman, Luis C Lamb, Leo de Penning, BV Illuminoo, Hoifung Poon, and COPPE Gerson Zaverucha. 2022. Neural-symbolic learning and reasoning: A survey and interpretation. *Neuro-Symbolic Artificial Intelligence: The State of the Art* 342, 1 (2022), 327.

[12] Jiafeng Guo, Yixing Fan, Liang Pang, Liu Yang, Qingyao Ai, Hamed Zamani, Chen Wu, W Bruce Croft, and Xueqi Cheng. 2020. A deep look into neural ranking models for information retrieval. *Information Processing & Management* 57, 6 (2020), 102067.

[13] Thomas Hofmann. 1999. Probabilistic latent semantic indexing. In *Proceedings of the 22nd annual international ACM SIGIR conference on Research and development in information retrieval*. 50–57.

[14] Saar Kuzi, Abhishek Narwekar, Anusri Pampari, and ChengXiang Zhai. 2019. Help me search: Leveraging user-system collaboration for query construction to improve accuracy for difficult queries. In *Proceedings of the 42nd International ACM SIGIR Conference on Research and Development in Information Retrieval*. 1221–1224.

[15] David Dolan Lewis. 1992. *Representation and learning in information retrieval.* University of Massachusetts Amherst.

[16] Kelong Mao, Zhicheng Dou, Haonan Chen, Fengran Mo, and Hongjin Qian. 2023. Large Language Models Know Your Contextual Search Intent: A Prompting Framework for Conversational Search. *arXiv preprint arXiv:2303.06573* (2023).

[17] Adam Pauls and Dan Klein. 2011. Faster and smaller n-gram language models. In *Proceedings of the 49th annual meeting of the Association for Computational Linguistics: Human Language Technologies*. 258–267.

[18] Baolin Peng, Chunyuan Li, Pengcheng He, Michel Galley, and Jianfeng Gao. 2023. Instruction tuning with gpt-4. *arXiv preprint arXiv:2304.03277* (2023).

[19] Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, Ilya Sutskever, et al. 2019. Language models are unsupervised multitask learners. *OpenAI blog* 1, 8 (2019), 9.

[20] Alireza Salemi, Sheshera Mysore, Michael Bendersky, and Hamed Zamani. 2023. LaMP: When Large Language Models Meet Personalization. *arXiv preprint arXiv:2304.11406* (2023).

[21] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Proceedings of the 31st International Conference on Neural Information Processing Systems* (Long Beach, California, USA) *(NIPS'17)*. Curran Associates Inc., Red Hook, NY, USA, 6000–6010.

[22] Ryen W White. 2024. Tasks, Copilots, and the Future of Search: A Keynote at SIGIR 2023. In *ACM SIGIR Forum*, Vol. 57. ACM New York, NY, USA, 1–8.

[23] Ke Yang, Jiateng Liu, John Wu, Chaoqi Yang, Yi R Fung, Sha Li, Zixuan Huang, Xu Cao, Xingyao Wang, Yiquan Wang, et al. 2024. If llm is the wizard, then code is the wand: A survey on how code empowers large language models to serve as intelligent agents. *arXiv preprint arXiv:2401.00812* (2024).

[24] Zhilin Yang, Zihang Dai, Yiming Yang, Jaime Carbonell, Russ R Salakhutdinov, and Quoc V Le. 2019. Xlnet: Generalized autoregressive pretraining for language understanding. *Advances in neural information processing systems* 32 (2019).

[25] ChengXiang Zhai et al. 2008. Statistical language models for information retrieval a critical review. *Foundations and Trends® in Information Retrieval* 2, 3 (2008), 137–213.

[26] Yutao Zhu, Huaying Yuan, Shuting Wang, Jiongnan Liu, Wenhan Liu, Chenlong Deng, Zhicheng Dou, and Ji-Rong Wen. 2023. Large language models for information retrieval: A survey. *arXiv preprint arXiv:2308.07107* (2023).

[27] Daniel M Ziegler, Nisan Stiennon, Jeffrey Wu, Tom B Brown, Alec Radford, Dario Amodei, Paul Christiano, and Geoffrey Irving. 2019. Fine-tuning language models from human preferences. *arXiv preprint arXiv:1909.08593* (2019).